Georgios Ntais

Development of a Stemmer for the Greek Language

February 2006

Department of Computer and Systems Sciences

Master Thesis at Stockholm University / Royal Institute of Technology

Thesis Supervisor: Associate professor Hercules Dalianis

This thesis corresponds to 20 weeks of full-time work

Abstract

In this thesis work, a stemming system for the Greek language is presented. This system takes as input a word and removes its inflexional suffix according to a rule based algorithm. The algorithm follows the known Porter algorithm for the English language and it is developed according to the grammatical rules of the Greek language, as they are described in Triantafyllidis grammar (1941) for the Modern Greek language. An extended documentation of the removal process as well as a short evaluation of the system is showing the algorithm accuracy that works with better performance than other past stemming algorithms for the Greek language giving 92.1 percent correct results. Finally, possible extensions of the proposed system and further evaluation methods are briefly reviewed.

Acknowledgements

I would like to thank my family for their support and encouragement in my studies abroad. I would also like to thank my supervisor and mentor Dr. Hercules Dalianis for his interest and enthusiasm about this thesis work. Finally I will not forget all my supportive friends, who shared their moments with me, during my master studies in Sweden.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

Nowadays many tools are provided for information retrieval. There are some interesting categories of the available information retrieval software. We can find a variety of Internet search engines with advanced search parameters, specialized search engines for retrieving documents in a document collection, data mining and clustering tools as well as other classification tools. During their development we can notice an ongoing specialization on the searching features. These engines are becoming more and more sophisticated trying to cover user's demands to access specific information.

One of the attempts to make the search engines more effective in information retrieval was the usage of word stemming. Lovins (1968) defines a stemming algorithm as "a procedure to reduce all words with the same stem to a common form, usually by stripping each word of its derivational and inflectional suffixes". The main objective of the stemming process is to remove all possible affixes and thus reduce the word to its stem (Dawson 1974). Using Stemming, many contemporary search engines associate words with prefixes and suffixes to their word stem, to make the search broader in the meaning that it can ensure that the greatest number of relevant matches is included in search results. Stemming has also applications in machine translation, document summarisation (Orasan, Pekar & Hasler 2004, Dalianis 2000), and text classification (Gaustad & Bouma 2002).

1.1 Background

Many theories and experiments have been developed to evaluate the efficiency and the stability of the stemming process in information retrieval. Lennon (1981) did an evaluation research about stemming techniques and how these affect the search precision, demonstrating that stemming raises the effectiveness of information retrieval. This was enough to motivate more and more researchers on stemming improvement.

There are several techniques used for word stemming, developed through time. From the first basic Dictionary-Based approach, up to the latest advance Corpus-Based Technique, researchers have been using alternative rules and formations for every language to develop a reliable stemmer with higher precision.

1.1.1 Dictionary-Based Technique

"Historically, stemmers have often been thought of as either dictionary-based or algorithmic" (Porter 2001). Dictionary-based stemmers match every word with a word on a proper digitalized dictionary, correspond each word to its stem (Carlberger et al. 2001). In Krovetz's dictionary experiments (Krovetz 1995), this direct method, seems effective but inadequate to deal with the "unlimited" words and their formation, especially in inflected languages with elevated morphological structure. This was the main reason that led him to evaluate algorithmic stemmers and conclude that "despite the errors they can be seen to make, they still give good practical results". Moreover "dictionary-based stemmers require dictionary maintenance, to keep up with an ever-changing language, and this is actually quite a problem. It is not only that a dictionary created to assist stemming nowadays will probably require major updating in a few years time, but also that a dictionary in use for this purpose today may already be several years out of date".

1.1.2 Rule-Based Technique

This is the widest applied stemming technique, with most representative the algorithm introduced by Porter (1980). With specific rules for the English language, this algorithm removes iteratively suffixes from a given word, reducing it on its stem. Even if the algorithm has its limitations, it is the most commonly accepted for its high precision and recall. Lovin's stemmer (1968) follows the same rule-based technique but it does not apply its rules iteratively and it is more conservative than Porter's algorithm. On this path Paice & Husk (1990) have also worked introducing one more English stemmer with different rules. For Scandinavian languages we have also rule-based stemmers presented on 2001 (Dalianis & Jongejan 2006). Finally, applying the same philosophy, there are implemented rule-based stemmers for:

Romance languages:
- English
- French
- Spanish
- Portuguese
- Italian

Germanic Languages:
- German
- Dutch

Scandinavian Languages:
- Swedish
- Norwegian
- Danish

Other Languages:
- Russian
- Finish

The above stemmers and their algorithms can be found in the web-space of "Tartarus" (http://snowball.tartaurus.org) and they are following the SNOBOL (StriNg Oriented symBOlic Language), a small string processing language designed for creating stemming algorithms for use in Information Retrieval. The stemmers created with SNOBOL named "Snowball".

1.1.3 Light-Stemming Technique

Today there are plenty of rule-based algorithms and stemmers, developed for various languages. Most of the times, for each of them, a different algorithm is used to reach higher precision in the results. So lately we have light-stemmers, referred to the process of stripping off a small set of prefixes and/or suffixes without trying to deal with infixes or recognize patterns and find roots (Sughaiyer & Kharashi 2004).

1.1.4 Corpus-Based Technique

According to Porter (2001) the algorithmic and dictionary-based stemmers are not clearly distinct. An algorithmic stemmer uses lists of words either for suffix removal or exclusion. The more advanced is the algorithm the longer are these lists. On the other hand, a dictionary-based stemmer needs to remove some basic suffixes before starts the look-up process in the extended dictionary. Trying to improve the effectiveness of these stemmers we are driven to the rule-based technique.

This hybrid perspective was applied in many stemming algorithms earlier, with most representatives the corpus-based stemming algorithm of Xu and Croft (1998). The hypothesis of that work is that the word forms that should be conflated will co-occur in documents from the corpus. It starts with a set of rough preliminary stem classes created by another stemmer, perhaps Porter or other that conflates all words starting with the same three letters. It then uses co-occurrence analysis of the words in the preliminary class to find those that do not appear to belong together. Corpus-based stemming was found to provide moderate improvement over existing rule-based stemmers. As they mention in their research "The basic idea behind this work is that we can use co-occurrence analysis of word variants within a particular corpus to ascertain which variants belong together and which do not, when stemmer like Porter's creates the initial word variant (stem) classes".

1.1.5 Stemming in non-English languages

For languages other than English, there are stemmers with different rules applied for each language. A famous stemmer with high percentage of precision and recall is the stemmer for Slovene (Popovic & Wilett 1992) while Savoy (1993) introduced another stemmer for the French language. For Scandinavian languages there is a comparison between CST's (Center for Language Technology) and Euroling's stemmers, showing the evolution of stemming algorithms and the rising demand in the Information Retrieval field (Dalianis & Jongejan 2006).

Building a rule-based stemmer for a new, arbitrary language is time consuming and requires experts with linguistic knowledge in that particular language (Rogati et al. 2003). For a new stemmer in Arabic language there is use of a parallel corpus technique to apply the known English algorithms. A parallel corpus is a collection of sentence pairs with the same meaning but in different languages (Rogati 2003). Usually, entire documents are translated by humans, and the sentence pairs are subsequently aligned by automatic means. A small parallel corpus can be available when native speakers and translators are not, which makes building a stemmer out of such corpus a preferable direction. An overview of the parallel corpus method, used for an Arab stemmer (Rogati et al. 2003), presented in Figure 1.

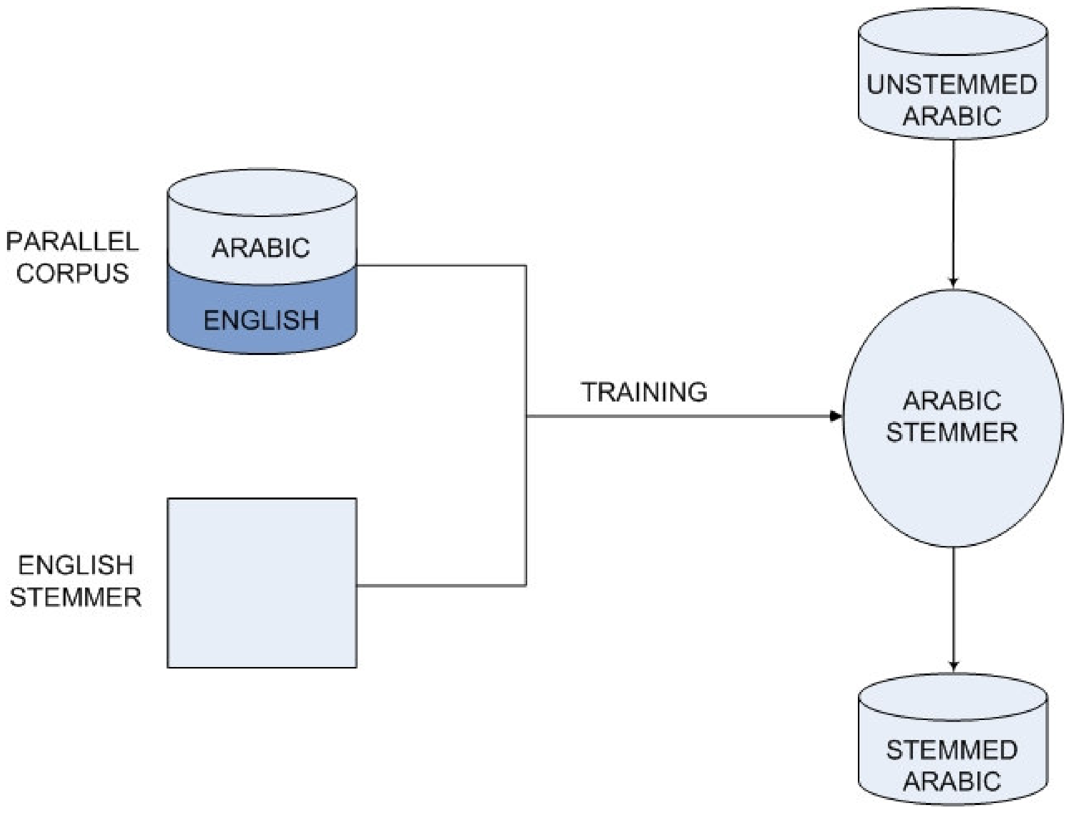


Figure 1: Parallel Corpus Stemmer

A parallel corpus stemmer is language independent and it has successfully been used by other researchers (Yarowsky 2000, Diab & Resnik 2002).

1.1.6 Greek Stemmers

On 2001, Tambouratzis and Carayannis (Tambouratzis & Carayannis 2001) presented a system that performs an automated morphological categorization of Greek words extracted from a corpus, for the Institute for Language and Speech Processing (ILSP) in Greece. The aim of the Automated Morphological Processor (AMP), whose structure is outlined in Figure 2, is to perform the segmentation of a given set of words into stems and endings in an automated manner. The algorithm is using rule-based iterative matching-and-masking approach, which relies on matching parts of different patterns. Here the stemming process is based on an initial set of valid stems and endings. There is also an assumption that each word consists of a stem part and an ending part excluding the compound words.

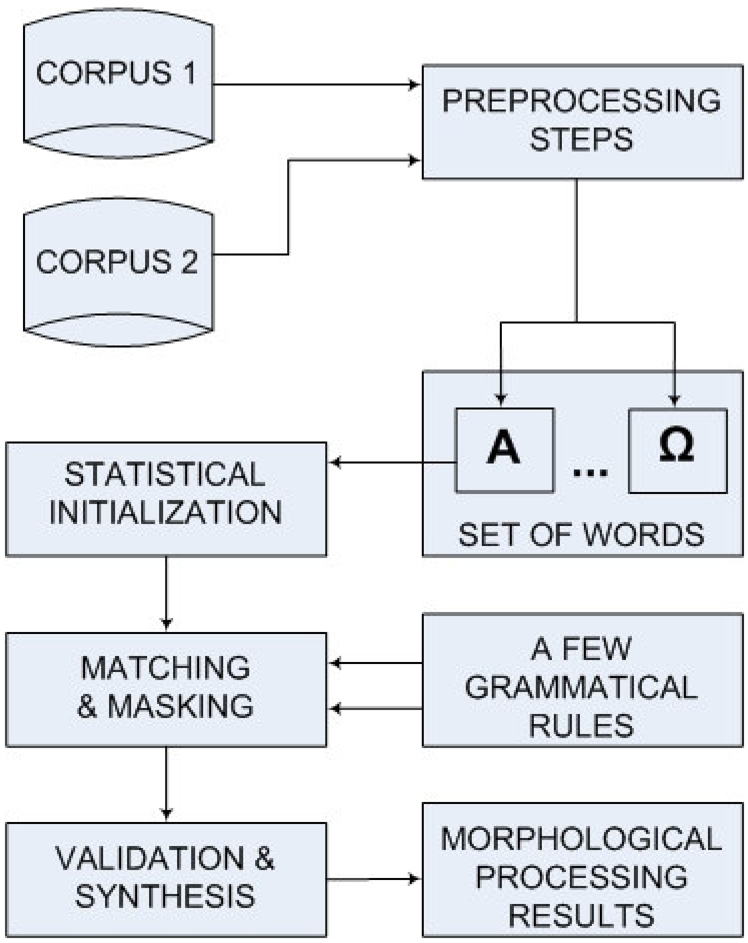


Figure 2: Automated Morphological Processor (AMP) overview

Even if this system is not a pure rule-based stemmer, it performs a successful stemming for Greek words, with the stemming accuracy being approximately 95 per cent. It can distinguish the ending and the stem for a given word and its performance depends on how rich in terms is the linguistic corpus for the stems and for the endings. Even though, the few grammatical rules that follow the matching-and-masking process are not enough to consider as complete stemming algorithm for the Greek language.

The AMP does not stop on the matching-and-masking process. During operation it continues with the synthesis of the stemming results and after 4 steps returns a number of possible solutions. To select the one that represents the correct segmentation, a ranking criterion is employed, using the existing ILSP lexicon for comparison purposes. More information about that system and its evaluation are presented on the relevant research paper (Tambouratzis & Carayannis 2001).

1.1.7 A suffix stripping algorithm

Another, more straightforward work about Greek stemming, took place from Kalamboukis and Nikolaidis in the Research Center of the Athens University of Economics and Business.

On 1995 (Kalamboukis & Nikolaidis 1995) published the first suffix stripping algorithm for the Greek language. That algorithm is designed for information retrieval from Greek texts and deals with inflections and derivations of the Greek language. They use a suffix lists and they have implemented an iterative suffix algorithm with two levels. The first corresponds to the inflectional analysis, and in the second level the derivational suffixes are removed according to their grammatical category. They have formed three different tables of suffixes corresponding to the three main grammatical categories: noun, adjective and verb. The suffixes are checked in accordance with their grammatical category and they are removed according to the specific suffix table in two steps. An overview of that system is given in Figure 3.

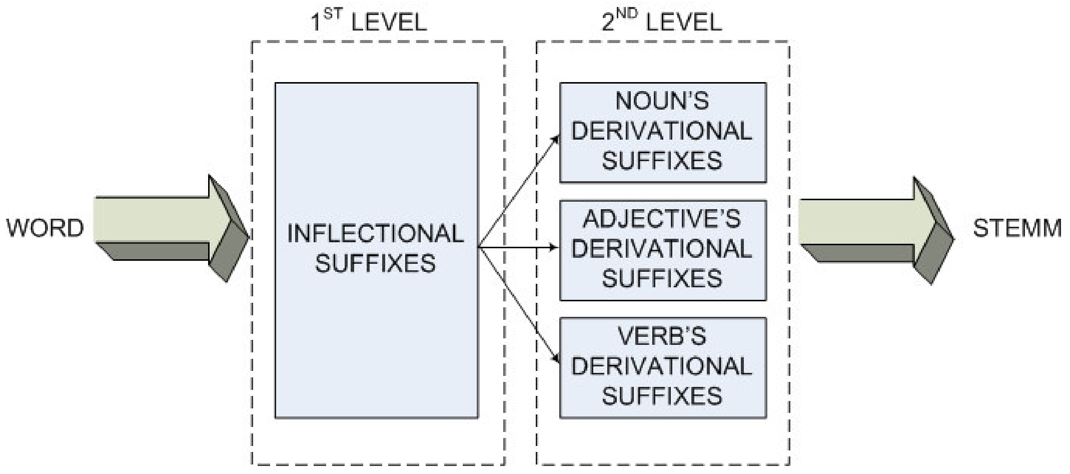

1ST LEVEL
2ND LEVEL
WORD
INFLECTIONAL SUFFIXES
NOUN'S DERIVATIONAL SUFFIXES
ADJECTIVE'S DERIVATIONAL SUFFIXES
VERB'S DERIVATIONAL SUFFIXES
STEMM

Figure 3: "TZK algorithm" overview

The "TZK algorithm", as it is mentioned on the relevant paper, removes totally 65 suffixes in both levels and as they admitted that "they have include only a small set of suffixes because they have reach a stage where the addition of more rules to increase the performance in one area causes a degradation of the performance elsewhere". This main constraint makes the algorithm limited, as in the Greek language there are at least 166 different inflectional suffixes (Triantafyllidis 1941). Furthermore, the algorithm works only with Greek capital letters in order to deal with the diacritical sign (tone-mark) that is placed over a lower case vowel affecting the meaning and the orthography of the word.

Another critical part of the algorithm is that the derivational suffix removal works according to the grammatical category of the word. And as there is no morphological analysis tool for the given word, the suffixes table on the 1st removal level is not enough to distinguish if a word is noun, adjective or verb.

Even though, the algorithm seems to perform an acceptable stemming for Greek texts. According to the first evaluation, in 1995, the algorithm was tested on two document collections on medical and computer science, with 7959 distinct words totally. The errors of the stemmed words were around 10 per cent with satisfactory precision and recall.

In 1999 the same researchers (Kalamboukis & Nikolaidis 1999) did an evaluation of stemming algorithms with Modern Greek using a different approach. Using SMART (Storage Management and Retrieval) system developed at Cornell University. They have added some new and modified existing procedures of SMART in order to make them handle Greek texts, including a "stopword" list of the most frequent Greek words and they tested 3 algorithms in total; the "TZK algorithm" (Kalamboukis & Nikolaidis 1995), the "infl_only" algorithm, which removes only 19 inflectional suffixes and a new modified version of the TZK algorithm. The evaluation of these algorithms showed that "stemming is a clustering process depended on the corpus and therefore to avoid conflation not appropriate we should incorporate corpus-based statistics in order to capture the concept of the terms". That extended evaluation as well as more specific statistical tests was presented on their paper published on 1999.

1.2 Problem

Each natural language has its own characteristics and features. So, it seems quite difficult to follow the same stemming pattern and apply the same stemming rules for all the languages, creating a generic rule-based algorithm. Different prefixes and suffixes, as well as individual exceptions, need special handling and a careful formation of a frame with specific norms, applied on the studied language.

As it mentioned above, there are some stemming methods for Greek texts, presented from the middle of 90's. These methods are parts of more extended work about morphological analysis and information retrieval from various texts and can't be consider as rule-based stemmers; even if the "TZK algorithm" is a set of rules.

According to the research about the Greek stemming, both (Kalamboukis & Nikolaidis 1995) and (Tambouratzis 2001), agree that specific grammatical rules can improve the effectiveness on information retrieval from Greek texts. The development of a Greek stemmer with extended grammatical rules will come to solve the existing problem of the previous limited algorithm.

1.3 Objective

The result of the thesis is an extended stemming algorithm for the Greek language; all these grammatical rules that can effectively remove specific suffixes of a given word. The algorithm is implemented, using JavaScript language, as a web based application and works through a simple web-site.

1.4 Purpose

We introduce this algorithm in order to cover the gap of the existing algorithm and create a more effective Greek stemmer. Based on the previous research, we will create a system that removes effectively the suffixes of the Greek words. An accurate Greek stemmer can be used for various purposes in Information Retrieval and Morphological Analysis.

This system will help the users to obtain more and better hits during searching and retrieving information. The advanced feature of stemming has upgraded the search standards for all of the languages that it has been developed. Furthermore, according to researches and measurements about the Greek stemming, a Greek stemmer on the Web will provide the users with more specific search results as Kalamboukis (1995) mentions in his research about stemming algorithms with Modern Greek.

1.5 Method

For the development of the Greek stemmer we are following the Porter algorithm (Porter 1980) as it seems to be the most reliable. Of course that algorithm is developed for the English language and we can not apply the same rules on a Greek stemmer. But the Greek stemmer tries to follow the simplicity and the directness of Porters'.

The research is based on the previous work about Greek stemming and its effectiveness. The development tool is JavaScript, an open-source script language, freely available on the Web. Finally for the evaluation of the stemmer we used the Greek keyword dictionary, kindly provided by the National Centre of Scientific Research "DEMOCRITOS" (Petasis et. al) and a random word corpus.

1.6 Limitations

To produce better stems one may add a number of constraints to the stemming algorithm. Some assumptions are necessary too, to specify the research. Otherwise we have to deal with a long time consuming process, trying to deal with extended rules like those of the Greek language.

First of all we will use only capital letters as (Kalamboukis & Nikolaidis 1995), trying to solve the problem of the "moving" tone-mark on the stems of the Greek words. Using only capital letters some words may be pronounced in different ways with different meanings each time. Such words however, are only a very small number with no serious affect in stemming effectiveness.

Prefixes in Greek may change the meaning of the word radically and sometimes the semantics. For this reason we have not considered prefix removal in this research. Besides the general prefixes, there are some cases of allomorphy in the Greek language. The verbs starting with consonant, take the letter "ε" as prefix on the past tenses (Triantafyllidis 1941). In these tenses the stem changes formation as well and that's why there are two stems for every verb. So we uphold this distinction and we accept that a verb has a different stem on the past tenses and on the other tenses.

Present/Future Tenses	Past Tenses	Present stem	Past stem
ΔΕΝΩ (I tide)	ΕΔΕΣΑ (I tided)	ΔΕΝ	ΔΕΣ
ΦΕΥΓΩ (I leave)	ΕΦΥΓΑ (I left)	ΦΕΥΓ	ΕΦΥΓ
ΠΑΙΖΩ (I play)	ΕΠΑΙΞΑ (I played)	ΠΑΙΖ	ΕΠΑΙΞ

Table 1: Present and Past stem for the same word

An accurate Greek stemmer should deal with both the inflectional and derivational endings (Kalamboukis 1995). But the Greek language is rich in derivative words. This means that we can have many words coming from the same stem. And if we think that there could be more than 10 derivational endings with around 50 inflectional endings each we have a list of around 500 words belongs in the same family and having the same stem. As we want to apply the Greek stemmer on a search engine, this will not be useful, because we will have too generic results. So we will deal only with inflectional endings.

Extended constrains about the Greek stemming algorithm are following in the second part of the thesis.

2. The Greek Language

2.1 History of Modern Greek

The language that Greeks speak today was not always the same. It is based on the Ancient Greek Language that was established in Athens on the 5th century B.C. As Athens was dominating with its political and spiritual acme, more and more tribes living in Greece adopted the same language. Up to Alexander's the Great era, Ancient Greek Language was spoken by the people living in Greece, Persia, Middle East and Egypt. During this Hellenistic Period, a lot of mixtures took place, adding new elements in the Greek language, especially the oral one. During Byzantine Empire, the Greek language changes again in syntax and grammatical formation trying to make the written formation simple as the oral one.

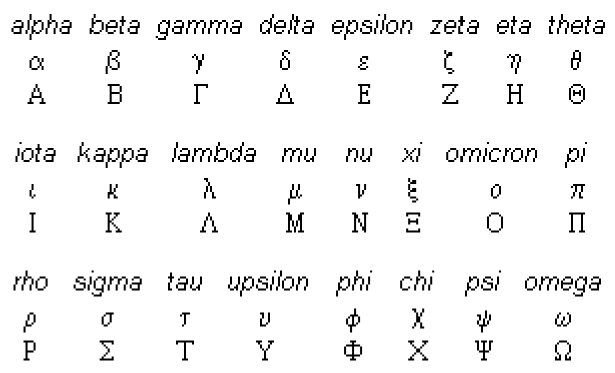
After the Ottoman occupation (1453), the Greek language is almost only oral and it keeps being oral for almost 400 years. People use a kind of dialect totally different from the classical Greek and obviously effected from the Ottomans.

After liberation (1821) the Greek nation needs a new formal language, before starts to follow the evolution of the rest Europe. In the early 19th century, there are two dispositions in Greece; the classical who wanted to establish a Greek language similar to the Ancient Greek and other scholars who wanted a simplified version of the Greek language more close to the spoken language of that period. After long time arguments Greece establish as formal language the "Katharevousa", introduced by Adamantios Korais. Katharevousa is something between Ancient and Modern Greek and it was used as formal Greek language up to 1976, even if most of the people were using Modern Greek in most of the cases.

For reasons of simplicity, the Greek parliament accepted on 1976 the Modern Greek language, called "Demotiki", as the official language of Greece, which is the present Greek language.

2.2 The Greek alphabet

The alphabet of the Modern Greek language is the same with the Ancient one and it originates from the Egyptian and Phoenician alphabet. It is presented in the Figure 4.



alpha	*beta*	*gamma*	*delta*	*epsilon*	*zeta*	*eta*	*theta*
α	β	γ	δ	ε	ζ	η	θ
Α	Β	Γ	Δ	Ε	Ζ	Η	Θ
iota	*kappa*	*lambda*	*mu*	*nu*	*xi*	*omicron*	*pi*
ι	κ	λ	μ	ν	ξ	ο	π
Ι	Κ	Λ	Μ	Ν	Ξ	Ο	Π
rho	*sigma*	*tau*	*upsilon*	*phi*	*chi*	*psi*	*omega*
ρ	σ	τ	υ	φ	χ	ψ	ω
Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω

Figure 4: The Greek alphabet

The Greek alphabet contains 24 letters, 7 vowels (A, E, H, I, O, Y and Ω) and 17 consonants (B, Γ, Δ, Z, Θ, K, Λ, M, N, Ξ, Π, P, Σ, T, Φ, X and Ψ).

The character "ς" called "teliko sigma" and it replaces the "σ" only when it is written at the end of a word and for the lower case letters.

Every word with more than 2 syllables, takes a tone-mark called "tonos", over one vowel and under specific rules. In some cases we can have tone-mark over one-syllable words or even words with two tone-marks. This tone-mark is used only for the lower case letters too.

2.3 Definitions

Before a brief explanation of the most important grammatical rules for the Greek language it is useful to define some important terms, common used in this thesis.

Syllable is the piece of word consisted of at least one single vowel or one vowel followed by one or more consonants.

Stem or **Theme** is the static (unchangeable) part on the start of a word.

Derivative is a word created by another word if an affix added on its themes.

Derivational suffix is a suffix of a derivative word

Root word is a word that can not created by another word. It is crated if a suffix added on a root or an initial theme.

Compound is a word created two other words, adding their themes.

Inflectional suffix is the variable (changeable) part on the end of a word; usually we call it just suffix.

In the Table 2 some words in different formations are presented. In correspondence with English Language, the word "ΧΑΡΑΚΤΗΡΑΣ" (character) is the root word. It consists of the stem "ΧΑΡΑΚΤΗΡ" and the suffix "ΑΣ". The same happens on the plural form of the same word where the suffix is "ΕΣ".

For the word "ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ" (characteristic), that has the same stem "ΧΑΡΑΚΤΗΡ", the suffix "ΙΣΤΙΚΟ" considered as derivational and the same appears on the plural form. The last vowel of this word (Ο) is the inflectional suffix.

Consequently, one root word can take many derivational suffixes and change formation or meaning. For each derivational word there are many inflectional suffixes, according to the gender, number, person and tense.

Word	Stem	Derivational Suffix	Suffix
ΧΑΡΑΚΤΗΡΑΣ (Character)	ΧΑΡΑΚΤΗΡ	-	ΑΣ
ΧΑΡΑΚΤΗΡΕΣ (Characters)	ΧΑΡΑΚΤΗΡ	-	ΕΣ
ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ (Characteristic)	ΧΑΡΑΚΤΗΡ	ΙΣΤΙΚΟ	Ο
ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ (Characteristics)	ΧΑΡΑΚΤΗΡ	ΙΣΤΙΚΑ	Α
ΧΑΡΑΚΤΗΡΙΖΩ (I characterize)	ΧΑΡΑΚΤΗΡ	ΙΖΩ	Ω
ΧΑΡΑΚΤΗΡΙΖΕΙΣ (You characterize)	ΧΑΡΑΚΤΗΡ	ΙΖΕΙΣ	ΕΙΣ
ΧΑΡΑΚΤΗΡΙΣΜΟΣ (Characterization)	ΧΑΡΑΚΤΗΡ	ΙΣΜΟΣ	ΟΣ
ΧΑΡΑΚΤΗΡΙΣΜΟΙ (Characterizations)	ΧΑΡΑΚΤΗΡ	ΙΣΜΟΙ	ΟΙ

Table 2: Inflectional and derivational suffixes

2.4 Grammar

The Greek language has ten different types of words: article, noun, adjective, pronoun, verb, participle, adverb, preposition, conjunction and interjection. The article, the noun, the adjective, the pronoun, the verb and the participle are inflectional and they have various types in the language. The noun is declined according to the number (singular, plural). The adjective is declined according to the gender (masculine, feminine, neuter) and the number as above. For each gender and number there are four different types for the nouns and the adjectives (nominative, genitive, accusative, vocative). The verb has number as above, but also tense and voice (passive, active).

Considering the nouns, there are totally 39 different suffixes in all their forms. Adding the adjectives in all their inflections, there are 17 more different suffixes. Counting also all the possible verb inflections, there are 110 more different suffixes. So for the general forms of the main inflectional types of the Greek language there are 166 different suffixes, presented in Figure 5.

ΑΔΕΣ	ΗΣΑΜΕ	ΙΕΣΤΕ	ΟΥΜΕ	ΕΣΑΙ	ΙΟΥΜΑΣΤΕ	ΗΣΟΥΝ	ΟΥΝ
ΑΔΩΝ	ΑΝΕ	ΗΚΑ	ΗΣΟΥΜΕ	ΕΣ	ΙΟΥΝΤΑΙ	ΗΣΩ	ΟΥΝΤΑΙ
ΕΔΕΣ	ΗΚΑΝΕ	ΗΚΕΣ	ΗΘΟΥΜΕ	ΕΤΑΙ	ΙΟΥΝΤΑΝ	Ο	ΟΥΝΤΑΝ
ΕΔΩΝ	ΗΘΗΚΑΝΕ	ΗΚΕ	ΑΤΑ	Ι	Η	ΟΙ	ΟΥΣ
ΟΥΔΕΣ	ΑΓΑΝΕ	ΗΘΗΚΑ	ΑΤΟΣ	ΙΕΜΑΙ	ΗΔΕΣ	ΟΜΑΙ	ΟΥΣΑΝ
ΟΥΔΩΝ	ΟΥΣΑΝΕ	ΗΘΗΚΕΣ	ΑΤΩΝ	ΙΕΜΑΣΤΕ	ΗΔΩΝ	ΟΜΑΣΤΑΝ	ΟΥΣΑΤΕ
ΕΩΣ	ΗΣΑΝΕ	ΗΘΗΚΕ	Α	ΙΕΤΑΙ	ΗΘΕΙ	ΟΜΟΥΝ	Υ
ΕΩΝ	ΙΟΝΤΑΝΕ	ΟΥΣΑ	ΑΓΑΤΕ	ΙΕΣΑΙ	ΗΘΕΙΣ	ΟΜΟΥΝΑ	ΥΣ
ΙΑ	ΙΟΤΑΝΕ	ΟΥΣΕΣ	ΑΓΑΝ	ΙΕΣΑΣΤΕ	ΗΘΕΙΤΕ	ΟΝΤΑΙ	Ω
ΙΟΥ	ΙΟΥΝΤΑΝΕ	ΟΥΣΕ	ΑΕΙ	ΙΟΜΑΣΤΑΝ	ΗΘΗΚΑΤΕ	ΟΝΤΑΝ	ΩΝ
ΙΩΝ	ΟΝΤΑΝΕ	ΑΓΑ	ΑΜΑΙ	ΙΟΜΟΥΝ	ΗΘΗΚΑΝ	ΟΝΤΟΥΣΑΝ	ΟΤΕΡ
ΙΚΟ	ΟΤΑΝΕ	ΑΓΕΣ	ΑΝ	ΙΟΜΟΥΝΑ	ΗΘΟΥΝ	ΟΣ	ΩΤΕΡ
ΙΚΟΥ	ΟΥΝΤΑΝΕ	ΑΓΕ	ΑΣ	ΙΟΝΤΑΝ	ΗΘΩ	ΟΣΑΣΤΑΝ	ΟΤΑΤ
ΙΚΑ	ΕΤΕ	ΗΣΑ	ΑΣΑΙ	ΙΟΝΤΟΥΣΑΝ	ΗΚΑΤΕ	ΟΣΑΣΤΕ	ΩΤΑΤ
ΙΚΩΝ	ΗΣΕΤΕ	ΗΣΕ	ΑΤΑΙ	ΙΟΣΑΣΤΑΝ	ΗΚΑΝ	ΟΣΟΥΝ	ΥΤΕΡ
ΑΜΕ	ΟΝΤΑΣ	ΗΣΟΥ	ΑΩ	ΙΟΣΑΣΤΕ	ΗΣ	ΟΣΟΥΝΑ	ΥΤΑΤ
ΗΚΑΜΕ	ΩΝΤΑΣ	ΗΣΤΕ	Ε	ΙΟΣΟΥΝ	ΗΣΑΝ	ΟΤΑΝ	ΕΣΤΕΡ
ΗΘΗΚΑΜΕ	ΟΜΑΣΤΕ	ΟΥΝΕ	ΕΙ	ΙΟΣΟΥΝΑ	ΗΣΑΤΕ	ΟΥ	ΕΣΤΑΤ
ΑΓΑΜΕ	ΙΟΜΑΣΤΕ	ΗΣΟΥΝΕ	ΕΙΣ	ΙΟΤΑΝ	ΗΣΕΙ	ΟΥΜΑΙ	ΙΑΣ
ΟΥΣΑΜΕ	ΕΣΤΕ	ΗΘΟΥΝΕ	ΕΙΤΕ	ΙΟΥΜΑΙ	ΗΣΕΣ	ΟΥΜΑΣΤΕ	ΙΕΣ
ΙΟΙ	ΙΟΥΣ	ΑΤΕ	ΤΕ	ΗΣΕΙΣ	ΑΣΤΕ		

Figure 5: 166 inflectional suffixes for nouns, adjectives and verbs

2.5 Stem in Greek words

As we mentioned above stem is the static part of a word. Some words in Greek language have two different stems. For the nouns and the adjectives, in their different inflections, the stem increased at one syllable. And this is not an exception in the Greek language. Some examples below show the two different stems for the same root word:

Single Number	Plural Number	Stems	Suffixes
ΚΥΜΑ (wave)	ΚΥΜΑΤΑ (waves)	ΚΥΜ, ΚΥΜΑΤ	Α
ΓΙΑΓΙΑ (grandmother)	ΓΙΑΓΙΑΔΕΣ (grandmothers)	ΓΙΑΓΙ, ΓΙΑΓΙΑΔ	Α, ΕΣ
ΑΛΕΠΟΥ (fox)	ΑΛΕΠΟΥΔΕΣ (foxes)	ΑΛΕΠ, ΑΛΕΠΟΥΔ	ΟΥ, ΕΣ

Table 3: Different stems of nouns

According to the Greek grammar, these two stems correspond, if the extra syllable considered as a part of the suffix. In this case we have the same stem but more suffixes, as it is presented in the Table 4.

Single Number	Plural Number	Stems	Suffixes
ΚΥΜΑ (wave)	ΚΥΜΑΤΑ (waves)	ΚΥΜ	Α, ΑΤΑ
ΓΙΑΓΙΑ (grandmother)	ΓΙΑΓΙΑΔΕΣ (grandmothers)	ΓΙΑΓΙ	Α, ΑΔΕΣ
ΑΛΕΠΟΥ (fox)	ΑΛΕΠΟΥΔΕΣ (foxes)	ΑΛΕΠ	ΟΥ, ΟΥΔΕΣ

Table 4: Corresponding stems

But it is not the same for the verbs. Every verb has two stems for each voice (passive and active); the “present stem” and the “past stem”. The tenses are formed according to these stems. Moreover, the verbs starting with a consonant take the letter “ε” in some of the past tenses. The various formations of the verb “ΔΕΝΩ” (tide) presented on the Table 5.

Tenses	Verb	Stems	Suffixes
Present Tenses	ΔΕΝΩ (I tide)	ΔΕΝ	Ω
	ΘΑ ΔΕΝΩ (I will be tiding)	ΔΕΝ	Ω
	ΔΕΝΕ (tide)	ΔΕΝ	Ε
	ΔΕΝΟΝΤΑΣ (tiding)	ΔΕΝ	ΟΝΤΑΣ
Past Tenses	ΕΔΕΣΑ (I tided)	ΔΕΣ	Α
	ΝΑ ΔΕΣΩ (to tide)	ΔΕΣ	Ω
	ΝΑ ΔΕΣΩ (to tide)	ΔΕΣ	Ω
	ΔΕΣΕ (tide)	ΔΕΣ	Ε
	ΔΕΣΕΙ (I have tide)	ΔΕΣ	ΕΙ

Table 5: Different stems of verbs

2.6 Agreements for the Greek Stemmer

Before we continue with the presentation of the stemming algorithm it is important to note the agreements that took place during this stemmer development. As the Greek language is highly inflectional, the suffix removal process could be very time-consuming and the algorithm could be very complicated using long corpus and many exceptions. On the other hand, if we have too general rules we may reduce two semantically different words to the same root (*overstemming*).

The algorithm works for the Greek words in capital letters as we mentioned in the paragraph 1.6 on the limitations of the system. Using characters in upper case we do not have to deal with the diacritical sign (tone-mark) that appears on the lower case characters and it often changes position in a word on its different inflections.

The algorithm removes only inflectional suffixes of a word. According the strict definition of the word “stemming”, both derivational and inflectional suffixes have to be removed. “TZK algorithm” has a rational approach on this point and it tries to deal with both inflectional and derivational suffixes. But as we mentioned above, on the system limitations, there are too many words coming from one stem. For this reason it is more rational to consider as stem a word without remove its derivational suffix. Inflectional suffixes affect much more information retrieval in Greek texts compared with English, as the Greek language is mush more inflectional.

Stemming is taking place for the words that change their suffixes. That is why we tried to develop a stemming algorithm only for inflectional types. And as the main inflectional types in the Greek types are the noun, the adjective and the verb we will deal only with them.

The algorithm distinguishes between the “past” and the “present” stem for the verbs. As the stems in the different tenses are different we can not reduce a verb from a present and a past tense in the same stem. This agreement is not also rational according to the strict definition of the “stemming”. But here we have to deal with a specific grammatical phenomenon for the Greek language. Removing only inflectional suffices we reduce a verb on its stem, either past or present.

3. The Greek Stemmer

3.1 The algorithm

The stemming algorithm focuses on the inflectional suffixes removal for any given inflectional Greek word. Taking under consideration the Greek grammar, we conclude on a list with 166 different suffixes for the 3 main inflectional word types: noun, adjective and verb. As the Greek stemmer can not be a “lemmatizer” and can not distinguish between the types of the words, our approach is straightforward removal of the suffixes.

The main idea is to filter the word through a suffixes list which contains all the possible endings. This list can easily be created after a careful studying on the Greek grammar and includes those 166 suffixes as we mention above. The overview of a system like this presented on the Figure 6.

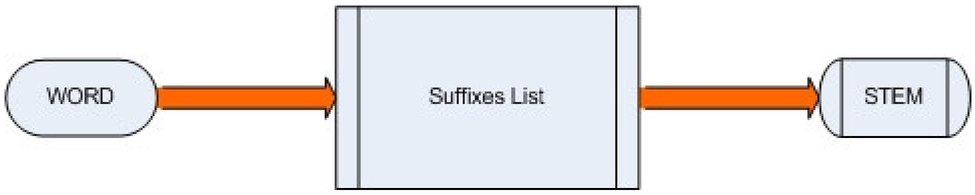


Figure 6: Generic overview

The problem on this system appears if we consider that some of the suffixes in this list can affect words in a wrong way, removing a wrong part of the word as suffix. For example we want to remove the suffixes “A” and “ΑΔΕΣ” for the words “MAM-A” (mother) and “MAM-ΑΔΕΣ” (mothers); so we will have the same stem “MAM” in both cases of plural and singular number. Applying the same general rule on another set of words “ΟΜΑΔ-Α” (team) and “ΟΜΑΔ-ΕΣ” (teams), we reduce these words in different stems, while the algorithm will reduce the plural number word to the stem “ΟΜ” and not “ΟΜΑΔ”.

To deal with this suffixes conflict, that often occurs in the Greek language, we can create a different list for the suffix “ΑΔΕΣ” and capture all or most of the words that are wrong affected from this rule. In the same list we can include the suffix “ΑΔΩΝ”, which is the ending of the same group of words on the genitive form. An algorithm that deals with “exceptions” individual for each suffix or group of suffixes is much more accurate and flexible than an algorithm with centralized rules.

Of course there are suffixes which do not conflict and can easily be removed. For example if we set a rule that removes the suffix “ΑΣ” for any given word, there is no conflict with other words. This general rule covers a big amount on words in the Greek language.

Finally, the rules, generic or specific, are applied each time for the longest possible suffix in the list. So when we have the suffixes “A” and “ATA” in the suffixes list, the word “KYMATA” (waves) will be reduced on the stem “KYM” and not “KYMAT”.

3.2 The rules

Trying to deal with each suffix individually, we have created a decentralized algorithm. The different rules are presented below in pseudo-code:

<u>**Rule-set 1**</u>

```
if (word ends on ΑΔΕΣ|ΑΔΩΝ){
      remove the suffix;
            if (remaining part does not end on ΟΚ|ΜΑΜ|ΜΑΝ…){
                  add “ΑΔ”;
            }
}
```

The rule removes the suffixes *ΑΔΕΣ* and *ΑΔΩΝ* for a group of words.

<u>Example:</u>

*ΓΙΑΓΙ**Α***	*ΓΙΑΓΙ*
*ΓΙΑΓΙ**ΑΔΩΝ***	*ΓΙΑΓΙ*

The rule doesn’t affect the group of words that by chance have similar suffixes.

<u>Example:</u>

*ΟΜΑΔ**Α***	*ΟΜΑΔ*
*ΟΜΑΔ**ΕΣ***	*ΟΜΑΔ*

<u>**Rule-set 2**</u>

```
if (word ends on ΕΔΕΣ|ΕΔΩΝ){
      remove the suffix;
            if (remaining part ends on ΟΠ|ΙΠ|ΕΜΠ…){
                  add “ΕΔ”;
            }
}
```

The rule removes the suffixes *ΕΔΕΣ* and *ΕΔΩΝ* for a group of words.

<u>Example:</u>

*ΚΑΦ**ΕΣ***	*ΚΑΦ*
*ΚΑΦ**ΕΔΩΝ***	*ΚΑΦ*

The rule doesn’t affect the group of words that by chance have similar suffixes.

<u>Example:</u>

*ΓΗΠΕΔ**Ο***	*ΓΗΠΕΔ*
*ΓΗΠΕΔ**ΩΝ***	*ΓΗΠΕΔ*

<u>Rule-set 3</u>

```
if (word ends on ΟΥΔΕΣ|ΟΥΔΩΝ){
      remove the suffix;
            if (remaining part ends on ΑΡΚ|ΚΑΛΙΑΚ|ΛΙΧ…){
                  add “ΟΥΔ”;
            }
}
```

The rule removes the suffixes *ΟΥΔΕΣ* and *ΟΥΔΩΝ* for a group of words.

<u>Example:</u>

*ΠΑΠΠ**ΟΥΣ***	*ΠΑΠΠ*
*ΠΑΠΠ**ΟΥΔΩΝ***	*ΠΑΠΠ*

The rule doesn’t affect the group of words that by chance have similar suffixes.

<u>Example:</u>

*ΑΡΚΟΥΔ**Α***	*ΑΡΚΟΥΔ*
*ΑΡΚΟΥΔ**ΕΣ***	*ΑΡΚΟΥΔ*

<u>Rule-set 4</u>

```
if (word ends on ΕΩΣ|ΕΩΝ){
      remove the suffix;
            if (remaining part is Θ|Δ|ΕΛ|ΓΑΛ…){
                  add “Ε”;
            }
}
```

The rule removes the suffixes *ΕΩΣ* and *ΕΩΝ* for a group of words.

Example:

*ΥΠΟΘΕΣ**Η***	*ΥΠΟΘΕΣ*
*ΥΠΟΘΕΣ**ΕΩΣ***	*ΥΠΟΘΕΣ*

The rule doesn't affect the group of words that by chance have similar suffixes.

Example:

*ΘΕ**ΟΣ***	*ΘΕ*
*ΘΕ**ΩΝ***	*ΘΕ*

Rule-set 5

```
if (word ends on IA|IOY|IΩN){
      remove the suffix;
            if (remaining part ends on vowel){
                  add "I";
            }
}
```

The rule removes the suffixes *IA*, *IOY* and *IΩN* for a group of words.

Example:

*ΠΑΙΔ**Ι***	*ΠΑΙΔ*
*ΠΑΙΔ**ΙΑ***	*ΠΑΙΔ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΤΕΛΕΙ**ΟΣ***	*ΤΕΛΕΙ*
*ΤΕΛΕΙ**ΟΥ***	*ΤΕΛΕΙ*

Rule-set 6

```
if (word ends on IKA|IKO|IKOY|IKΩN){
      remove the suffix;
            if (remaining part is AΛ|AΔ|ENΔ|AMAN …) || (remaining
                  part ends on vowel){
                  add "IK";
            }
}
```

The rule removes the suffixes *IKA*, *IKO*, *IKOY* and *IKΩN* for a group of words.

Example:

*ZHΛIAP**HΣ***	*ZHΛIAP*
*ZHΛIAP**IKO***	*ZHΛIAP*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*AΓPOIK**OΣ***	*AΓPOIK*
*AΓPOIK**OY***	*AΓPOIK*

Rule-set 7

```
if (word is ΑΓΑΜΕ){make the stem “ΑΓΑΜ”;}
      if (word ends on ΑΓΑΜΕ|ΗΣΑΜΕ|ΟΥΣΑΜΕ|ΗΚΑΜΕ|ΗΘΗΚΑΜΕ){
                  remove the suffix;
      }
if (word ends on ΑΜΕ){
      remove the suffix;
            if (remaining part is ΑΝΑΠ|ΑΠΟΘ|ΑΠΟΚ…){
                  add “ΑΜ”;
            }
}
```

The rule removes the suffixes *AΓAME*, *HΣAME*, *OYΣAME*, *HKAME* and *HΘHKAME* for all the words, and the suffix *AME* for a group of words.

Example:

*AΓAΠ**Ω***	*AΓAΠ*
*AΓAΠ**AΓAME***	*AΓAΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ANAΠAM**OΣ***	*ANAΠAM*
*ANAΠAM**E***	*ANAΠAM*

Rule-set 8

```
if (word ends on ΑΓΑΝΕ|ΗΣΑΝΕ|ΟΥΣΑΝΕ|ΙΟΝΤΑΝΕ|ΙΟΤΑΝΕ|
ΙΟΥΝΤΑΝΕ|ΟΝΤΑΝΕ|ΟΤΑΝΕ|ΟΥΝΤΑΝΕ|ΗΚΑΝΕ|ΗΘΗΚΑΝΕ){
      remove the suffix;
            if (remaining part is ΤΡ|ΤΣ){
                  add "ΑΓΑΝ";
            }
}
if (word ends on ΑΝΕ){
      remove the suffix;
            if (remaining part is ΒΕΤΕΡ|ΒΟΥΛΚ…) || (remaining part
            ends on vowel){
                  add "ΑΝ";
            }
}
```

The rule removes the suffixes *ΗΣΑΝΕ*, *ΟΥΣΑΝΕ*, *ΙΟΝΤΑΝΕ*, *ΙΟΤΑΝΕ*, *ΙΟΥΝΤΑΝΕ*, *ΟΝΤΑΝΕ*, *ΟΤΑΝΕ*, *ΟΥΝΤΑΝΕ*, *ΗΚΑΝΕ* and *ΗΘΗΚΑΝΕ* for any word, and the suffixes *ΑΓΑΝΕ* and *ΑΝΕ* for a group of words.

Example:

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΗΣΑΝΕ***	*ΑΓΑΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΤΡΑΓΑΝ**ΟΣ***	*ΤΡΑΝΑΝ*
*ΤΡΑΓΑΝ**Ε***	*ΤΡΑΝΑΝ*

Rule-set 9

```
if (word ends on ΗΣΕΤΕ){
                  remove the suffix;
}
if (word ends on ΕΤΕ){
      remove the suffix;
            if (remaining part is ΑΒΑΡ|ΒΕΝ|ΕΝΑΡ…) || (remaining
            part ends on ΟΔ|ΑΙΡ|ΦΟΡ…) || (remaining part ends on
            vowel){
                  add "ΕΤ";
            }
}
```

The rule removes the suffix *ΗΣΕΤΕ* for all the words, and the suffix *ΕΤΕ* for a group of words.

Example:

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΗΣΕΤΕ***	*ΑΓΑΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΒΕΝΕΤ**ΟΣ***	*ΒΕΝΕΤ*
*ΒΕΝΕΤ**Ε***	*ΒΕΝΕΤ*

Rule-set 10

```
if (word ends on ΟΝΤΑΣ|ΩΝΤΑΣ){
      remove the suffix;
            if (remaining part is ΑΡΧ){
                  add "ΟΝΤ";
            }
            If (remaining part ends on ΚΡΕ){
                  add "ΩΝΤ";
            }
}
```

The rule removes the suffix ΟΝΤΑΣ and ΩΝΤΑΣ for a group of words.

Example:

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΩΝΤΑΣ***	*ΑΓΑΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΑΡΧΟΝΤ**ΑΣ***	*ΑΡΧΟΝΤ*
*ΚΡΕΩΝΤ**ΑΣ***	*ΚΡΕΩΝΤ*

Rule-set 11

```
if (word ends on ΟΜΑΣΤΕ|ΙΟΜΑΣΤΕ){
      remove the suffix;
            if (remaining part is ΟΝ){
                  add "ΟΜΑΣΤ";
            }

}
```

The rule removes the suffix *ΟΜΑΣΤΕ* and *ΙΟΜΑΣΤΕ* for a group of words.

<u>Example:</u>

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΙΟΜΑΣΤΕ***	*ΑΓΑΠ*

The rule doesn't affect one word that by chance has similar suffix.

<u>Example:</u>

*ΟΝΟΜΑΣΤ**ΟΣ***	*ΟΝΟΜΑΣΤ*
*ΟΝΟΜΑΣΤ**Ε***	*ΟΝΟΜΑΣΤ*

<u>Rule-set 12</u>

```
if (word ends on ΙΕΣΤΕ){
      remove the suffix;
            if (remaining part is Π|ΑΠ|ΣΥΜΠ…){
                  add "ΙΕΣΤ";
            }
}
if (word ends on ΕΣΤΕ){
      remove the suffix;
            if (remaining part is ΑΛ|ΑΡ|ΕΚΤΕΛ…){
                  add "ΕΣΤ";
            }
}
```

The rule removes the suffix *ΙΕΣΤΕ* and *ΕΣΤΕ* for a group of words.

<u>Example:</u>

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΙΕΣΤΕ***	*ΑΓΑΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

<u>Example:</u>

*ΠΙΕΣΤ**ΟΣ***	*ΠΙΕΣΤ*
*ΠΙΕΣΤ**Ε***	*ΠΙΕΣΤ*

Rule-set 13

```
if (word ends on ΗΘΗΚΑ|ΗΘΗΚΕΣ|ΗΘΗΚΕ){
      remove the suffix;
}
if (word ends on ΗΚΑ|ΗΚΕΣ|ΗΚΕ){
      remove the suffix;
            If (remaining part is ΔΙΑΘ|Θ|ΣΥΝΘ…) ||
            (remaining part ends on ΣΦ|ΟΘ|ΠΙΘ…){
                  add "ΗΚ";
            }
}
```

The rule removes the suffix *ΗΘΗΚΑ, ΗΘΗΚΕΣ* and *ΗΘΗΚΕ* for all the words, and the suffixes *ΗΚΑ, ΗΚΕΣ* and *ΗΚΕ* for a group of words.

Example:

*ΧΤΙΖ**Ω***	*ΧΤΙΖ**
*ΧΤΙΣΤ**ΗΚΕ***	*ΧΤΙΣΤ**

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΔΙΑΘΗΚ**Η***	*ΔΙΑΘΗΚ*
*ΔΙΑΘΗΚ**ΕΣ***	*ΔΙΑΘΗΚ*

***Notice the difference on the present and the past stem**

Rule-set 14

```
if (word ends on ΟΥΣΑ|ΟΥΣΕΣ|ΟΥΣΕ){
      remove the suffix;
            if (remaining part is ΦΑΡΜΑΚ|ΧΑΔ|ΜΕΔ…) || (remaining
            part ends on ΠΟΔΑΡ|ΒΛΕΠ…) || (remaining part ends on
            vowel){
                  add "ΟΥΣ";
            }
}
```

The rule removes the suffix *ΟΥΣΑ, ΟΥΣΕΣ* and *ΟΥΣΕ* for a group of words.

Example:

*ΧΤΥΠ**Ω***	*ΧΤΥΠ*
*ΧΤΥΠ**ΟΥΣΕΣ***	*ΧΤΥΠ*

The rule doesn’t affect one group of words that by chance have similar suffixes.

Example:

*ΜΕΔΟΥΣ**Α***	*ΜΕΔΟΥΣ*
*ΜΕΔΟΥΣ**ΕΣ***	*ΒΕΝΕΤ*

Rule-set 15

```
if (word ends on ΑΓΑ|ΑΓΕΣ|ΑΓΕ){
      remove the suffix;
            if ((remaining part is ΑΒΑΣΤ|ΠΟΛΥΦ|ΑΔΗΦ…) ||
            (remaining part ends on ΟΦ|ΠΕΛ|ΧΟΡΤ…)) && !((remaining
            part is ΨΟΦ|ΝΑΥΛΟΧ)
            || (remaining part ends on ΚΟΛΛ)) {
                  add “ΑΓ”;
            }
}
```

The rule removes the suffix *ΑΓΑ, ΑΓΕΣ* and *ΑΓΕ* for a group of words.

Example:

*ΚΟΛΛΑ**Ω***	*ΚΟΛΛ*
*ΚΟΛΛ**ΑΓΕΣ***	*ΚΟΛΛ*

The rule doesn’t affect one group of words that by chance have similar suffixes.

Example:

*ΑΒΑΣΤΑΓ**Ο***	*ΑΒΑΣΤ*
*ΑΒΑΣΤΑΓ**Α***	*ΑΒΑΣΤ*

Rule-set 16

```
if (word ends on ΗΣΕ|ΗΣΟΥ|ΗΣΑ){
      remove the suffix;
            if (remaining part is Ν|ΧΕΡΣΟΝ|ΔΩΔΕΚΑΝ…){
                  add “ΗΣ”;
            }
}
```

The rule removes the suffix *ΗΣΕ, ΗΣΟΥ* and *ΗΣΑ* for a group of words.

Example:

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΗΣΕ***	*ΑΓΑΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΝΗΣ**ΟΣ***	*ΝΗΣ*
*ΝΗΣ**ΟΥ***	*ΝΗΣ*

Rule-set 17

```
if (word ends on ΗΣΤΕ){
      remove the suffix;
            if (remaining part is ΑΣΒ|ΣΒ|ΑΧΡ…){
                  add "ΗΣΤ";
            }
}
```

The rule removes the suffix *ΗΣΤΕ* for a group of words.

Example:

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΗΣΤΕ***	*ΑΓΑΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΣΒΗΣΤ**ΟΣ***	*ΣΒΗΣΤ*
*ΣΒΗΣΤ**Ε***	*ΣΒΗΣΤ*

Rule-set 18

```
if (word ends on ΟΥΝΕ|ΗΣΟΥΝΕ|ΗΘΟΥΝΕ){
      remove the suffix;
            if (remaining part is Ν|Ρ|ΣΠΙ…){
                  add "ΟΥΝ";
            }
}
```

The rule removes the suffixes *ΟΥΝΕ, ΗΣΟΥΝΕ* and *ΗΘΟΥΝΕ* for a group of words.

Example:

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΟΥΝΕ***	*ΑΓΑΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΝΟΥΝ**ΟΣ***	*ΣΒΗΣΤ*
*ΝΟΥΝ**Ε***	*ΣΒΗΣΤ*

Rule-set 19

```
if (word ends on ΟΥΜΕ|ΗΣΟΥΜΕ|ΗΘΟΥΜΕ){
      remove the suffix;
            if (remaining part is ΠΑΡΑΣΟΥΣ|Φ|Χ…){
                  add "ΟΥΜ";
            }
}
```

The rule removes the suffixes *ΟΥΜΕ, ΗΣΟΥΜΕ* and *ΗΘΟΥΜΕ* for a group of words.

Example:

*ΑΓΑΠ**Ω***	*ΑΓΑΠ*
*ΑΓΑΠ**ΟΥΜΕ***	*ΑΓΑΠ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΦΟΥΜ**ΟΣ***	*ΦΟΥΜ*
*ΦΟΥΜ**Ε***	*ΦΟΥΜ*

Rule-set 20

```
if (word ends on ΜΑΤΑ|ΜΑΤΩΝ|ΜΑΤΟΣ){
      remove the suffix;
      add "ΜΑ";
}
```

The rule removes the suffixes *ΑΤΑ, ΑΤΩΝ* and *ΑΤΟΣ* for a group of words.

Example:

*ΚΥΜ**Α***	*ΚΥΜ*
*ΚΥΜ**ΑΤΑ***	*ΚΥΜ*

The rule doesn't affect one group of words that by chance have similar suffixes.

Example:

*ΧΩΡΑΤ**Ο***	*ΧΩΡΑΤ*
*ΧΩΡΑΤ**Α***	*ΧΩΡΑΤ*

Rule-set 21

```
if (word ends on Α|ΑΓΑΤΕ|ΑΓΑΝ…){
      remove the suffix;
}
```

The rule removes the suffixes *Α, ΑΓΑΤΕ, ΑΓΑΝ, ΑΕΙ, ΑΜΑΙ, ΑΝ, ΑΣ, ΑΣΑΙ, ΑΤΑΙ, ΑΩ, Ε, ΕΙ, ΕΙΣ, ΕΙΤΕ, ΕΣΑΙ, ΕΣ, ΕΤΑΙ, Ι, ΙΕΜΑΙ, ΙΕΜΑΣΤΕ, ΙΕΤΑΙ, ΙΕΣΑΙ, ΙΕΣΑΣΤΕ, ΙΟΜΑΣΤΑΝ, ΙΟΜΟΥΝ, ΙΟΜΟΥΝΑ, ΙΟΝΤΑΝ, ΙΟΝΤΟΥΣΑΝ, ΙΟΣΑΣΤΑΝ, ΙΟΣΑΣΤΕ, ΙΟΣΟΥΝ, ΙΟΣΟΥΝΑ, ΙΟΤΑΝ, ΙΟΥΜΑ, ΙΟΥΜΑΣΤΕ, ΙΟΥΝΤΑΙ, ΙΟΥΝΤΑΝ, Η, ΗΔΕΣ, ΗΔΩΝ, ΗΘΕΙ, ΗΘΕΙΣ, ΗΘΕΙΤΕ, ΗΘΗΚΑΤΕ, ΗΘΗΚΑΝ, ΗΘΟΥΝ, ΗΘΩ, ΗΚΑΤΕ, ΗΚΑΝ, ΗΣ, ΗΣΑΝ, ΗΣΑΤΕ, ΗΣΕΙ, ΗΣΕΣ, ΗΣΟΥΝ, ΗΣΩ, Ο, ΟΙ, ΟΜΑΙ, ΟΜΑΣΤΑΝ, ΟΜΟΥΝ, ΟΜΟΥΝΑ, ΟΝΤΑΙ, ΟΝΤΑΝ, ΟΝΤΟΥΣΑΝ, ΟΣ, ΟΣΑΣΤΑΝ, ΟΣΑΣΤΕ, ΟΣΟΥΝ, ΟΣΟΥΝΑ, ΟΤΑΝ, ΟΥ, ΟΥΜΑΙ, ΟΥΜΑΣΤΕ, ΟΥΝ, ΟΥΝΤΑΙ, ΟΥΝΤΑΝ, ΟΥΣ, ΟΥΣΑΝ, ΟΥΣΑΤΕ, Υ, ΥΣ, Ω* and *ΩΝ* for all the words.

Rule-set 22

```
if (word ends on ΕΣΤΕΡ|ΕΣΤΑΤ|ΟΤΕΡ|ΟΤΑΤ|ΥΤΕΡ|ΥΤΑΤ|ΩΤΕΡ|ΩΤΑΤ){
      remove the suffix;
}
```

The rule removes the sub-suffixes *ΕΣΤΕΡ, ΕΣΤΑΤ, ΟΤΕΡ, ΟΤΑΤ, ΥΤΕΡ, ΥΤΑΤ, ΩΤΕΡ* and *ΩΤΑΤ* for a group of words.

Example:

*ΠΛΗΣΙ**ΕΣΤΑΤ**(ΟΣ)**	*ΠΛΗΣΙ*
*ΜΕΓΑΛ**ΥΤΕΡ**(Η)**	*ΜΕΓΑΛ*
*ΚΟΝΤ**ΟΤΕΡ**(Ο)**	*ΚΟΝΤ*

***The suffixes *ΟΣ, Η* or *Ο* have been removed on a previous step**

From the 166 inflectional endings we have effectively manage to remove 158 of them, using 22 Rule-sets, as it is presented on the Figure 7.

158 removals								8 missing suffixes
ΑΔΕΣ	ΗΣΑΜΕ	ΙΕΣΤΕ	ΟΥΜΕ	ΕΣΑΙ	ΙΟΥΜΑΣΤΕ	ΗΣΟΥΝ	ΟΥΝ	--ΙΑΣ
ΑΔΩΝ	ΑΝΕ	ΗΚΑ	ΗΣΟΥΜΕ	ΕΣ	ΙΟΥΝΤΑΙ	ΗΣΩ	ΟΥΝΤΑΙ	--ΙΕΣ
ΕΔΕΣ	ΗΚΑΝΕ	ΗΚΕΣ	ΗΘΟΥΜΕ	ΕΤΑΙ	ΙΟΥΝΤΑΝ	Ο	ΟΥΝΤΑΝ	--ΙΟΙ
ΕΔΩΝ	ΗΘΗΚΑΝΕ	ΗΚΕ	ΑΤΑ	Ι	Η	ΟΙ	ΟΥΣ	--ΙΟΥΣ
ΟΥΔΕΣ	ΑΓΑΝΕ	ΗΘΗΚΑ	ΑΤΟΣ	ΙΕΜΑΙ	ΗΔΕΣ	ΟΜΑΙ	ΟΥΣΑΝ	--ΑΤΕ
ΟΥΔΩΝ	ΟΥΣΑΝΕ	ΗΘΗΚΕΣ	ΑΤΩΝ	ΙΕΜΑΣΤΕ	ΗΔΩΝ	ΟΜΑΣΤΑΝ	ΟΥΣΑΤΕ	--ΤΕ
ΕΩΣ	ΗΣΑΝΕ	ΗΘΗΚΕ	Α	ΙΕΤΑΙ	ΗΘΕΙ	ΟΜΟΥΝ	Υ	--ΗΣΕΙΣ
ΕΩΝ	ΙΟΝΤΑΝΕ	ΟΥΣΑ	ΑΓΑΤΕ	ΙΕΣΑΙ	ΗΘΕΙΣ	ΟΜΟΥΝΑ	ΥΣ	--ΑΣΤΕ
ΙΑ	ΙΟΤΑΝΕ	ΟΥΣΕΣ	ΑΓΑΝ	ΙΕΣΑΣΤΕ	ΗΘΕΙΤΕ	ΟΝΤΑΙ	Ω	
ΙΟΥ	ΙΟΥΝΤΑΝΕ	ΟΥΣΕ	ΑΕΙ	ΙΟΜΑΣΤΑΝ	ΗΘΗΚΑΤΕ	ΟΝΤΑΝ	ΩΝ	
ΙΩΝ	ΟΝΤΑΝΕ	ΑΓΑ	ΑΜΑΙ	ΙΟΜΟΥΝ	ΗΘΗΚΑΝ	ΟΝΤΟΥΣΑΝ	ΟΤΕΡ	
ΙΚΟ	ΟΤΑΝΕ	ΑΓΕΣ	ΑΝ	ΙΟΜΟΥΝΑ	ΗΘΟΥΝ	ΟΣ	ΩΤΕΡ	
ΙΚΟΥ	ΟΥΝΤΑΝΕ	ΑΓΕ	ΑΣ	ΙΟΝΤΑΝ	ΗΘΩ	ΟΣΑΣΤΑΝ	ΟΤΑΤ	
ΙΚΑ	ΕΤΕ	ΗΣΑ	ΑΣΑΙ	ΙΟΝΤΟΥΣΑΝ	ΗΚΑΤΕ	ΟΣΑΣΤΕ	ΩΤΑΤ	
ΙΚΩΝ	ΗΣΕΤΕ	ΗΣΕ	ΑΤΑΙ	ΙΟΣΑΣΤΑΝ	ΗΚΑΝ	ΟΣΟΥΝ	ΥΤΕΡ	
ΑΜΕ	ΟΝΤΑΣ	ΗΣΟΥ	ΑΩ	ΙΟΣΑΣΤΕ	ΗΣ	ΟΣΟΥΝΑ	ΥΤΑΤ	
ΗΚΑΜΕ	ΩΝΤΑΣ	ΗΣΤΕ	Ε	ΙΟΣΟΥΝ	ΗΣΑΝ	ΟΤΑΝ	ΕΣΤΕΡ	
ΗΘΗΚΑΜΕ	ΟΜΑΣΤΕ	ΟΥΝΕ	ΕΙ	ΙΟΣΟΥΝΑ	ΗΣΑΤΕ	ΟΥ	ΕΣΤΑΤ	
ΑΓΑΜΕ	ΙΟΜΑΣΤΕ	ΗΣΟΥΝΕ	ΕΙΣ	ΙΟΤΑΝ	ΗΣΕΙ	ΟΥΜΑΙ		
ΟΥΣΑΜΕ	ΕΣΤΕ	ΗΘΟΥΝΕ	ΕΙΤΕ	ΙΟΥΜΑΙ	ΗΣΕΣ	ΟΥΜΑΣΤΕ		

Figure 7: Captured and non-captured suffixes

The reason we can not capture eight of the suffixes is that the wrong affected words from a removal Rule-set are more than the right affected ones. So if we set a removal rule for these endings we need a very long list to exclude the group of words wrongly affected. Considering the time-consumption and the precision of the algorithm we choose to keep these suffixes without removal rules.

The final overview of the system is presented on the Figure 8 below.

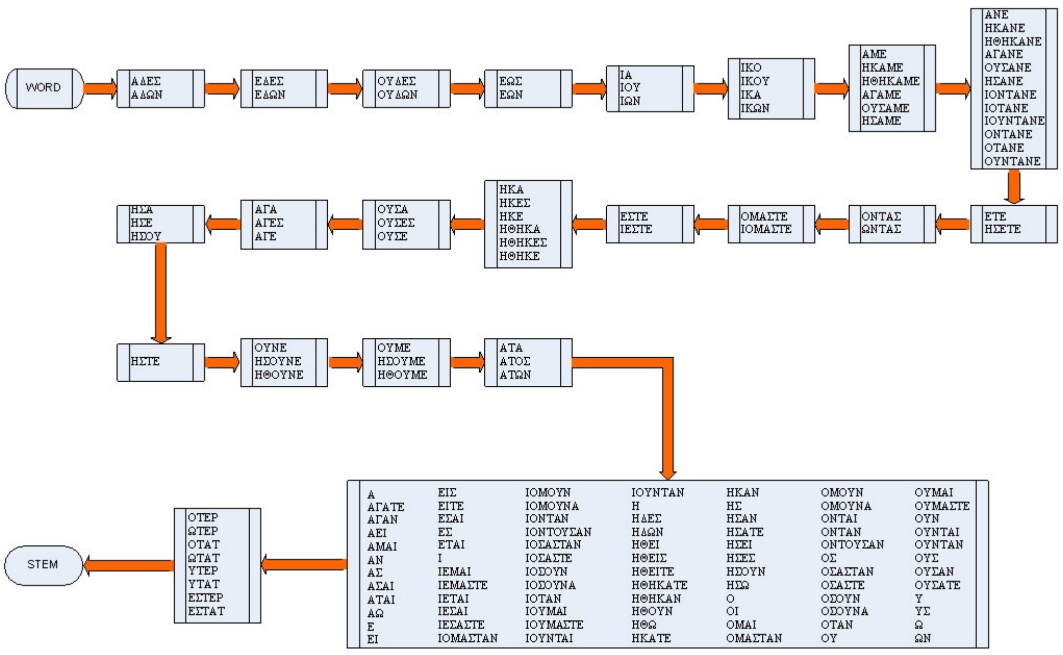

WORD
ΑΔΕΣ ΑΔΩΝ
ΕΔΕΣ ΕΔΩΝ
ΟΥΔΕΣ ΟΥΔΩΝ
ΕΩΣ ΕΩΝ
ΙΑ ΙΟΥ ΙΩΝ
ΙΚΟ ΙΚΟΥ ΙΚΑ ΙΚΩΝ
ΑΜΕ ΗΚΑΜΕ ΗΘΗΚΑΜΕ ΑΓΑΜΕ ΟΥΣΑΜΕ ΗΣΑΜΕ
ΑΝΕ ΗΚΑΝΕ ΗΘΗΚΑΝΕ ΑΓΑΝΕ ΟΥΣΑΝΕ ΗΣΑΝΕ ΙΟΝΤΑΝΕ ΙΟΤΑΝΕ ΙΟΥΝΤΑΝΕ ΟΝΤΑΝΕ ΟΤΑΝΕ ΟΥΝΤΑΝΕ
ΕΤΕ ΗΣΕΤΕ
ΟΝΤΑΣ ΩΝΤΑΣ
ΟΜΑΣΤΕ ΙΟΜΑΣΤΕ
ΕΣΤΕ ΙΕΣΤΕ
ΗΚΑ ΗΚΕΣ ΗΚΕ ΗΘΗΚΑ ΗΘΗΚΕΣ ΗΘΗΚΕ
ΟΥΣΑ ΟΥΣΕΣ ΟΥΣΕ
ΑΓΑ ΑΓΕΣ ΑΓΕ
ΗΣΑ ΗΣΕ ΗΣΟΥ
ΗΣΤΕ
ΟΥΝΕ ΗΣΟΥΝΕ ΗΘΟΥΝΕ
ΟΥΜΕ ΗΣΟΥΜΕ ΗΘΟΥΜΕ
ΑΤΑ ΑΤΟΣ ΑΤΩΝ
Α ΑΓΑΤΕ ΑΓΑΝ ΑΕΙ ΑΜΑΙ ΑΝ ΑΣ ΑΣΑΙ ΑΤΑΙ ΑΩ Ε ΕΙ
ΕΙΣ ΕΙΤΕ ΕΣΑΙ ΕΣ ΕΤΑΙ Ι ΙΕΜΑΙ ΙΕΜΑΣΤΕ ΙΕΤΑΙ ΙΕΣΑΙ ΙΕΣΑΣΤΕ ΙΟΜΑΣΤΑΝ
ΙΟΜΟΥΝ ΙΟΜΟΥΝΑ ΙΟΝΤΑΝ ΙΟΝΤΟΥΣΑΝ ΙΟΣΑΣΤΑΝ ΙΟΣΑΣΤΕ ΙΟΣΟΥΝ ΙΟΣΟΥΝΑ ΙΟΤΑΝ ΙΟΥΜΑΙ ΙΟΥΜΑΣΤΕ ΙΟΥΝΤΑΙ
ΙΟΥΝΤΑΝ Η ΗΔΕΣ ΗΔΩΝ ΗΘΕΙ ΗΘΕΙΣ ΗΘΕΙΤΕ ΗΘΗΚΑΤΕ ΗΘΗΚΑΝ ΗΘΟΥΝ ΗΘΩ ΗΚΑΤΕ
ΗΚΑΝ ΗΣ ΗΣΑΝ ΗΣΑΤΕ ΗΣΕΙ ΗΣΕΣ ΗΣΟΥΝ ΗΣΩ Ο ΟΙ ΟΜΑΙ ΟΜΑΣΤΑΝ
ΟΜΟΥΝ ΟΜΟΥΝΑ ΟΝΤΑΙ ΟΝΤΑΝ ΟΝΤΟΥΣΑΝ ΟΣ ΟΣΑΣΤΑΝ ΟΣΑΣΤΕ ΟΣΟΥΝ ΟΣΟΥΝΑ ΟΤΑΝ ΟΥ
ΟΥΜΑΙ ΟΥΜΑΣΤΕ ΟΥΝ ΟΥΝΤΑΙ ΟΥΝΤΑΝ ΟΥΣ ΟΥΣΑΝ ΟΥΣΑΤΕ Υ ΥΣ Ω ΩΝ
ΟΤΕΡ ΩΤΕΡ ΟΤΑΤ ΩΤΑΤ ΥΤΕΡ ΥΤΑΤ ΕΣΤΕΡ ΕΣΤΑΤ
STEM

Figure 8: Overview of the system

The key point is in the longest word-list. All the suffixes there are removed without any other specific rule. All the suffixes in the previous word-lists contain rules that exclude some words from the suffix removal.

The system takes as input any given Greek word in upper case letters. Before this word reaches the long word-list, it is tested on all the other Rule-sets. If the suffix matches, it is removed and the word skips the long word list. Then it is tested once more in the last word-list (comparisons removal). If the word was an adjective comparison then the comparison suffix is removed and the stem comes as output. In the case that none of the short word-list Rule-sets capture the suffix, the word is tested on the long word-list and then on the last word list. The part that comes as output is the stem of the given word.

Notice that the Figure 6 is an abstractive model of the Figure 8.

3.3 Implementation of Greek Stemmer

The Greek Stemmer is implemented as a sequential program with a basic web-interface. We chose to have the Greek Stemmer available on the web, so the algorithm is free for testing by everyone.

One of the simplest ways to implement the Greek stemming algorithm is with JavaScript. An open source script language that works fast without complex calculations, and it does not need any further configuration. It supports also the Greek char-set and works proper in any browser. The web interface is also simple according to the snowball prototype (http://snowball.tartarus.org/demo.php). The *stemword()* function in JavaScript is available in the web-rage of the Greek Stemmer (http://dis.dsv.su.se/~x04-gen/stemmer/stemmer.asp).

4. Evaluation

4.1 The Method

In order to evaluate the Greek Stemmer we used two different word-sets. The first one (WS1) is an extract from the Greek keyword dictionary of DEMOCRITOS and contains 703 words in various formations. The second (WS2) is a collection of 177 random pseudo-Greek words not contained in the Greek keyword dictionary.

We consider correct stemmed word any word without inflectional suffix. The stemmed words were evaluated by the supervisor of the thesis Dr. Hercules Dalianis and a sample of the tables with the word-sets and their stems is presented in Appendix A.

4.2 The Results

On the Tables 6 and 7 below, there are summarised the results of the evaluation.

Wordset	Words	Correct Stems	Percentage
WS1	703	652	92,7%
WS2	177	162	91,5%

Table 6: Correct Stems

Table 7 illustrates the distribution of errors for the wrong stemmed words in each word-set. With the term “overstemming” we define the wrong stems that occur because of the exceeding suffix removal. We notice “understemming” in the case that the algorithm stops before reach the correct stem.

Wordset	Errors	Overstemming(%)	Understemming(%)
WS1	51	88,3%	11,7%
WS2	15	93,4%	6,6%

Table 7: Distribution or errors

We can say that the algorithm is working with a quite good precision as the error percentages (7,3% and 8,5%) are considered as acceptable. Based on the Table 7 we can conclude that even if we follow an inflection removal technique, without removing the derivational part of the words, we still have overstemming errors. These errors occur because of the large exception words in the Greek language as well as the high inflectional character of the language.

In similar experiments on error distribution in Kalamboukis & Nikolaidis (1995) research, they had an average of 17,8% overstemming and 69,7% understemming (the rest 12,5% referred as other errors). We mention that the average of the precision in that algorithm was 89,6%.

5. Conclusions and Future Work

5.1 Conclusions

From these first evaluation experiments we can accept that we have reached an appropriate level of suffix removal, according to our initial assumptions. Our goal was to develop and document a new rule based stemmer for the Greek language, which follows the structure of Porter algorithm (and others rule based algorithms); the most effective algorithm regarding precision and recall measurements. Following this technique, we took under consideration the existing research on the Greek information retrieval methodologies and approaches as well as the past research in Greek stemmers, and we initiated a new rule-set for the Greek stemmer.

The deep study of the Greek grammar as well as the analysis of the inflectional types of the Greek language was necessary for this kind of thesis work. We decide to design a new path for the word classification according to their suffixes and not to work with a word classification tool. This approach follows the Swedish stemmer approach (Carlberger et al. 2001) and it was necessary as the past research in stemming for the Greek language is limited.

Analyzing the Greek grammatical rules we decide to follow inflectional suffix removal. This assumption was reasonable if we consider the time limitation of the thesis work and the architecture of our stemmer. Of course every researcher is aiming for the best result in his/her work. And when a new algorithm for word stemming is created, the system specifications should be well defined and documented. All the researchers agree that the stem definition requires in depth suffix removal, both inflectional and derivational. But as we are trying to develop a useful tool for effective information retrieval for the Greek language, we can not skip the specific language rules. Following the strict linguistic definition maybe we can create a complex and unique tool, pointing out the root of any given word. But such a kind of tool is not the object of regard in this work. Our purpose was to develop a "smart" stemmer, working effectively, with high precision and recall, in various texts in search engines. The main assumption to reach this goal was to accept as suffixes only the inflectional endings of the Greek words and work trying to remove them with the most effective way. And with 92,1% precision in our results we can count that we have create a good stemmer, compared to 89,6% of Kalamboukis & Nikolaidis stemmer.

The Greek stemmer was developed in JavaScript programming language giving high flexibility in recall of the system. We tried to follow a simple structure in the algorithm, creating small rule-sets for similar suffixes, which are working as Rule-sets on the input words. As this is a pilot work, one word per time is stemmed and there is no function for extended text stemming. The web-stemmer exists in the web address *http://www.dsv.su.se/~hercules/greek_stemmer.gr.html* and its web interface is illustrated in Appendix B.

5.2 Future Work

The research in this thesis work has lead to a prototype stemmer for the Greek language that seems to work with high precision, according to the first evaluation tests. Like any other pilot software it has its drawbacks and further improvement required regarding the improvement of the algorithm and the efficiency of the stemming process.

The 7,9% of errors is a number that can be reduced introducing more stemming rules and exceptions Rule-sets. But a big step in the future improvement of the Greek stemmer can be a study on how the derivational suffixes affect the Greek words and their stems, and how one can include new derivative rules that do not affect the effectiveness of the stemming process. All the rules described in this work can be a base for this further research and it can support extended stemming rules covering most of the terms in the Greek language.

Moreover, the stemmer has to be tested with large amount of texts to prove its real performance. To succeed this we need to apply the Greek stemmer in a web search engine, which retrieves information from Greek texts. Then we can have a complete view of the stemming system and the returned results after every search request. In this case we can do extended evaluation tests, we can measure the precision and recall in various texts and we can estimate the errors distribution in the stemming results.

Finally we believe that this thesis work contribute in the stemming research and offer a pilot tool for the Greek language that can be used free on the web by everyone.

References

Al-Sughaiyer I. and Al-Kharashi I. (2004): *Arabic Morphological Analysis Techniques: A Comprehensive Survey*. Journal of the American Society for Information Science and Technology, Vol 55, Issue 3, pp. 189 - 213.

Carlberger J., Dalianis H., Hassel M. and Knutsson O. (2001): *Improving Precision in Information Retrieval for Swedish using Stemming*. NODALIDA '01 - 13th Nordic Conference on Computational Linguistics, May 21-22 2001, Uppsala, Sweden.

Dalianis H. (2000): *SweSum - A Text Summarizer for Swedish.* TRITA-NA-P0015, IPLab-174, NADA, KTH, http://www.dsv.su.se/~hercules/papers/Textsumsummary.html

Dalianis, H. and Jongejan B. (2006): Hand-crafted versus machine-learned inflectional rules: The SiteSeeker stemmer and CST's lemmatiser, to be presented at the International Conference on Language Resources and Evaluation, LREC 2006, May 22-28, Genoa, Italy

Dawson J.L. (1974): *Suffix removal and word connation*. Bulletin of the Association for Literary and Linguistic Computing, No. 2, pp. 33-46.

Diab M. and Resnik Ph. (2002): *An unsupervised method for word sense tagging using parallel corpora*. In Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL-02).

Gaustad T. and Bouma G. (2002): *Accurate Stemming of Dutch for Text Classification*. In Computational Linguistics in the Netherlands 2001, pp. 104-117.

Kalamboukis T. Z. and Nikolaidis S. (1999): *An Evaluation of Stemming Algorithms with Modern Greek*. Proceedings of the 7th Hellenic Conference on Informatics, pp. 61- 70.

Kalamboukis T. Z. and Nikolaidis S. (1995): *Suffix stripping with Modern Greek.* Program, 29, pp. 313-321

Krovetz R. (1995): *Word sense disambiguation for large text databases*. PhD Thesis. Department of Computer Science, University of Massachusetts Amherst.

Lennon M., Pierce D.S., Tarry B.D. and Willett P. (1981): *An evaluation of some conflation algorithms for information retrieval*. Journal of information Science, 3, pp.177-183.

Lovins J.B. (1968): *Development of a stemming algorithm*. Mechanical Translation and Computational Linguistics, Vol. 11 No. 1/2, pp. 22-31.

Orasan C., Pekar V., and Hasler L. (2004): *A comparison of summarisation methods based on term specificity estimation*. In Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC-04). May, Lisbon, Portugal pp.1037-1041.

Paice C. and Husk G. (1990): *Another Stemmer*, ACM SIGIR Forum 24(3): p 566

Petasis G. , Karkaletsis V.,. Farmakiotou D, Androutsopoulos I., and C.D. Spyropoulos. ÒA Greek Morphological Lexicon and its Exploitation by Natural Language Processing Applications. Lecture Notes on Computer Science (LNCS), vol.2563, "Advances in Informatics - Post-proceedings of the 8th Panhellenic Conference in Informatics". Vol. editors: Yannis Manolopoulos, Skevos Evripidou, Antonis Kakas, pp. 401-419, 2003.

Porter M. (1980): *An algorithm for suffix stripping*. Program, 14(3), pp. 130-137.

Porter M. (October 2001)*: http://snowball.tartarus.org*

Popovic M. and Willett P. (1992): *The effectiveness of stemming for natural-language access to Slovene textual data*. Journal of the American Society for Information Science, 43(5), pp. 384-390.

Rogati M., McCarley S. and Yiming Y. (2003): *Unsupervised Learning of Arabic Stemming using a Parallel Corpus*. Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, pp. 391-398.

Savoy J. (1993): *Stemming of French words based on grammatical categories*. Journal of the American Society for Information Science, Vol. 44, No 1, pp. 1-10.

Tambouratzis G. and Carayannis G. (2001): *Automatic Corpora-based Stemming in Greek*. Literary and Linguistic Computing, Vol. 16, No. 4.

Triantafyllidis M. (1941): *Modern Greek Grammar*. ”Institute M Triantalyllidis”.

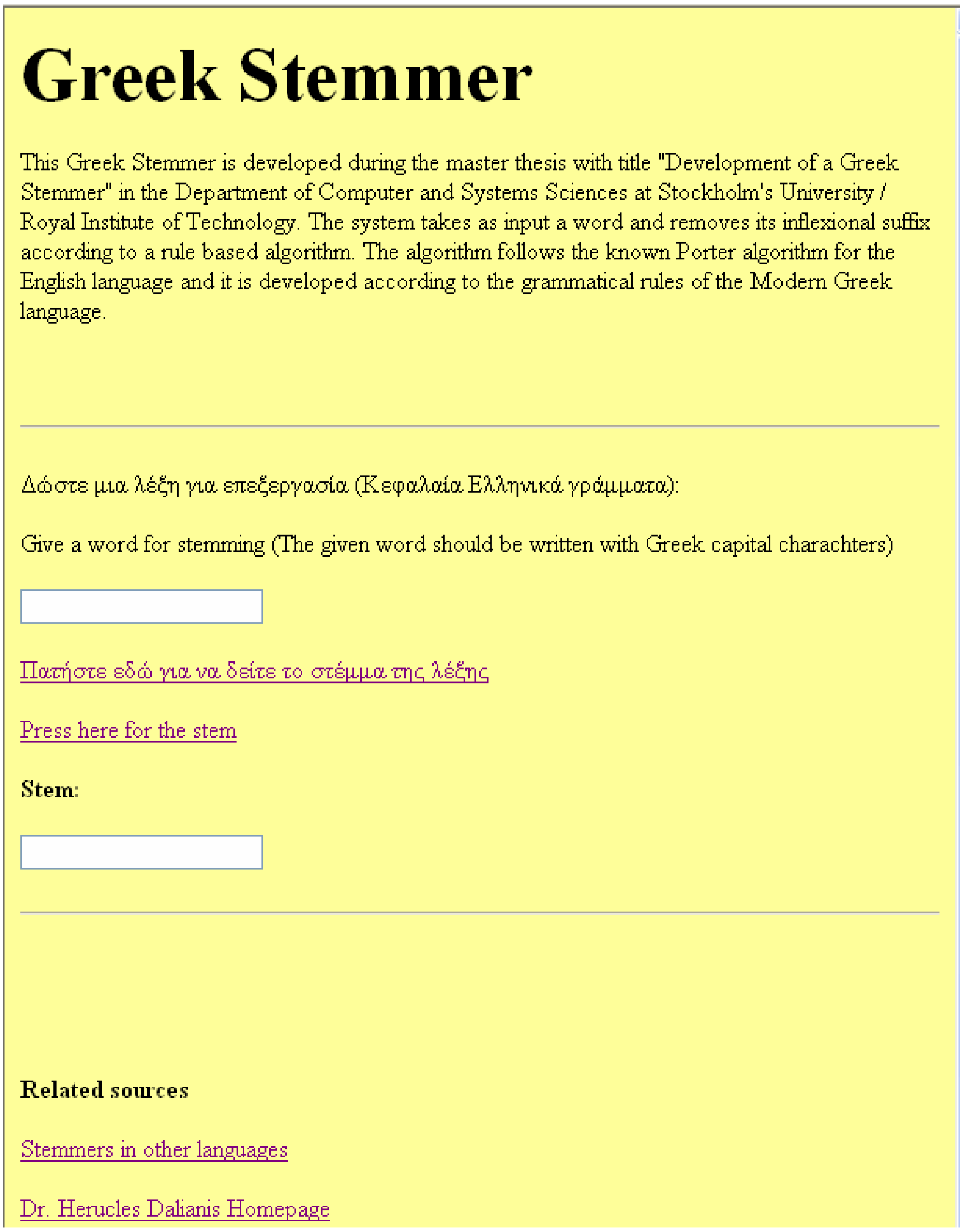
Xu J. and Croft W. B. (1998): *Corpus-based stemming using cooccurrence of word variants*. ACM Transactions on Information Systems, 16 (1), pp. 61-81.

Yarowsky D. (2000): *Hierarchical Decision Lists for Word Sense Disambiguation*. Computer and the Humanities, 34, pp. 179–186.

APPENDIX A: Evaluation Results

Word-set 1		Word-set 2	
ΑΥΤΟΚΙΝΗΣΗ	ΑΥΤΟΚΙΝΗΣ	ΥΔΡΟΘΕΡΑΠΕΙΑ	ΥΔΡΟΘΕΡΑΠΕΙ
ΑΥΤΟΚΙΝΗΣΗΣ	ΑΥΤΟΚΙΝΗΣ	ΥΔΡΟΘΕΡΑΠΕΙΑΣ	ΥΔΡΟΘΕΡΑΠΕΙ
ΑΥΤΟΚΙΝΗΤΑ	ΑΥΤΟΚΙΝΗΤ	ΥΔΡΟΘΕΡΑΠΕΙΕΣ	ΥΔΡΟΘΕΡΑΠΕΙ
ΑΥΤΟΚΙΝΗΤΕ	ΑΥΤΟΚΙΝΗΤ	ΥΔΡΟΘΕΡΑΠΕΙΩΝ	ΥΔΡΟΘΕΡΑΠΕΙ
ΑΥΤΟΚΙΝΗΤΕΣ	ΑΥΤΟΚΙΝΗΤ	ΠΑΙΔΟΠΟΔΗΛΑΤΑ	ΠΑΙΔΟΠΟΔΗΛΑΤ
ΑΥΤΟΚΙΝΗΤΗ	ΑΥΤΟΚΙΝΗΤ	ΠΑΙΔΟΠΟΔΗΛΑΤΟ	ΠΑΙΔΟΠΟΔΗΛΑΤ
ΑΥΤΟΚΙΝΗΤΗ	ΑΥΤΟΚΙΝΗΤ	ΠΑΙΔΟΠΟΔΗΛΑΤΟΥ	ΠΑΙΔΟΠΟΔΗΛΑΤ
ΑΥΤΟΚΙΝΗΤΟ	ΑΥΤΟΚΙΝΗΤ	ΠΑΙΔΟΠΟΔΗΛΑΤΩΝ	ΠΑΙΔΟΠΟΔΗΛΑΤ
ΑΥΤΟΚΙΝΗΤΟΙ	ΑΥΤΟΚΙΝΗΤ	ΠΑΙΔΟΠΟΔΗΛΑΤΟΥ	ΠΑΙΔΟΠΟΔΗΛΑΤ
ΑΥΤΟΚΙΝΗΤΟΣ	ΑΥΤΟΚΙΝΗΤ	ΠΑΙΔΟΠΟΔΗΛΑΤΩΝ	ΠΑΙΔΟΠΟΔΗΛΑΤ
ΑΥΤΟΚΙΝΗΤΟΥ	ΑΥΤΟΚΙΝΗΤ	ΒΡΟΧΟΧΟΡΕΥΑ	ΒΡΟΧΟΧΟΡΕΥ
ΑΥΤΟΚΙΝΗΤΟΥΣ	ΑΥΤΟΚΙΝΗΤ	ΒΡΟΧΟΧΟΡΕΥΑΝ	ΒΡΟΧΟΧΟΡΕΥ
ΑΥΤΟΚΙΝΗΤΩΝ	ΑΥΤΟΚΙΝΗΤ	ΒΡΟΧΟΧΟΡΕΥΕ	ΒΡΟΧΟΧΟΡΕΥ
ΧΟΡΔΙΖΑΜΕ	ΧΟΡΔΙΖ	ΒΡΟΧΟΧΟΡΕΥΕΣ	ΒΡΟΧΟΧΟΡΕΥ
ΧΟΡΔΙΖΑΝΕ	ΧΟΡΔΙΖ	ΒΡΟΧΟΧΟΡΕΨΑ	ΒΡΟΧΟΧΟΡΕΨ
ΧΟΡΔΙΖΑΤΕ	ΧΟΡΔΙΖ	ΒΡΟΧΟΧΟΡΕΨΑΝ	ΒΡΟΧΟΧΟΡΕΨ
ΧΟΡΔΙΖΕΙ	ΧΟΡΔΙΖ	ΒΡΟΧΟΧΟΡΕΨΕ	ΒΡΟΧΟΧΟΡΕΨ
ΧΟΡΔΙΖΕΙΣ	ΧΟΡΔΙΖ	ΒΡΟΧΟΧΟΡΕΨΕΣ	ΒΡΟΧΟΧΟΡΕΨ
ΧΟΡΔΙΖΕΤΕ	ΧΟΡΔΙΖ	ΕΠΙΤΡΟΠΟ	ΕΠΙΤΡΟΠ
ΧΟΡΔΙΖΟΜΕ	ΧΟΡΔΙΖΟΜ	ΕΠΙΤΡΟΠΟΙ	ΕΠΙΤΡΟΠ
ΧΟΡΔΙΖΟΝΤΑΣ	ΧΟΡΔΙΖ	ΕΠΙΤΡΟΠΟΣ	ΕΠΙΤΡΟΠ
ΧΟΡΔΙΖΟΥΜΕ	ΧΟΡΔΙΖ	ΕΠΙΤΡΟΠΟΥ	ΕΠΙΤΡΟΠ
ΧΟΡΔΙΖΟΥΝ	ΧΟΡΔΙΖ	ΕΠΙΤΡΟΠΟΥΣ	ΕΠΙΤΡΟΠ
ΧΟΡΔΙΖΟΥΝΕ	ΧΟΡΔΙΖ	ΕΠΙΤΡΟΠΩΝ	ΕΠΙΤΡΟΠ
ΧΟΡΔΙΖΩ	ΧΟΡΔΙΖ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΑ	ΒΟΡΕΙΟΑΣΙΑΤ
ΧΟΡΔΙΞΕΙ	ΧΟΡΔΙΞ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΕ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΧΟΡΔΙΣΑΜΕ	ΧΟΡΔΙΣ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΕΣ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΧΟΡΔΙΣΑΝ	ΧΟΡΔΙΣ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΗ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΜΕΛΛΟΝ	ΜΕΛΛΟΝ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΗΣ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΜΕΛΛΟΝΤΑ	ΜΕΛΛΟΝΤ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΟΙ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΜΕΛΛΟΝΤ	ΜΕΛΛΟΝΤ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΟΥ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΜΕΛΛΟΝΤΑΣ	ΜΕΛΛ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΟΥΣ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΜΕΛΛΟΝΤΕΣ	ΜΕΛΛΟΝΤ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΟ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΜΕΛΛΟΝΤΟΣ	ΜΕΛΛΟΝΤ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΟΣ	ΒΟΡΕΙΟΑΣΙΑΤΙΚ
ΑΡΕΙΕ	ΑΡΕΙ	ΒΟΡΕΙΟΑΣΙΑΤΙΚΩΝ	ΒΟΡΕΙΟΑΣΙΑΤ
ΑΡΕΙΟ	ΑΡΕΙ	ΥΠΟΜΕΛΗ	ΥΠΟΜΕΛ
ΑΡΕΙΟΙ	ΑΡΕΙ	ΥΠΟΜΕΛΟΣ	ΥΠΟΜΕΛ
ΑΡΕΙΟΣ	ΑΡΕΙ	ΥΠΟΜΕΛΟΥΣ	ΥΠΟΜΕΛ
ΑΡΕΙΟΥ	ΑΡΕΙ	ΥΠΟΜΕΛΩΝ	ΥΠΟΜΕΛ

APPENDIX B: User Interface



Greek Stemmer

This Greek Stemmer is developed during the master thesis with title "Development of a Greek Stemmer" in the Department of Computer and Systems Sciences at Stockholm's University / Royal Institute of Technology. The system takes as input a word and removes its inflexional suffix according to a rule based algorithm. The algorithm follows the known Porter algorithm for the English language and it is developed according to the grammatical rules of the Modern Greek language.

Δώστε μια λέξη για επεξεργασία (Κεφαλαία Ελληνικά γράμματα):

Give a word for stemming (The given word should be written with Greek capital charachters)

Πατήστε εδώ για να δείτε το στέμμα της λέξης

Press here for the stem

Stem:

Related sources

Stemmers in other languages

Dr. Hercules Dalianis Homepage

The Greek stemmer is available in the web-address
http://www.dsv.su.se/~hercules/greek_stemmer.gr.html