Development and Enhancement of a Stemmer for the Greek Language

Georgios Ntais
European Dynamics SA
209, Kifissias av.
15124 Athens, Greece
+30 210 8094500
georgios.ntais@eurodyn.com

Spyridon Saroukos
School of Information Sciences, University of Tampere
Kanslerinrinne 1, Pinni B
Tampere FI-33014, Finland
spyretto@gmail.com

Eleni Berki
School of Information Sciences, University of Tampere,
Kanslerinrinne 1, Pinni B
Tampere FI-33014, Finland
+358 40 190 42 96
eleni.berki@uta.fi

Hercules Dalianis
DSV/Stockholm University
P.O. Box 7003,
164 07 Kista,
Sweden
+46 8 674 75 47
hercules@dsv.su.se

ABSTRACT
Although there are three stemmers published for the Greek language, only the one presented in this paper and called Ntais' stemmer is freely open and available, together with its enhancements and extensions according to Saroukos' algorithm. The primary algorithm (Ntais' algorithm) uses only capital letters and works with better performance than other past stemming algorithms for the Greek language, giving 92.1 percent correct results. Further extensions of the proposed stemming system (e.g. from capital to small letters) and more evaluation methods are presented according to a new and improved algorithm, Saroukos' algorithm. Stemmer performance metrics are further used for evaluating the existing stemming system and algorithm and show how its accuracy and completeness are enhanced. The improvements were possible by providing an alternative implementation in the programming language PHP, which offers more syntactical rules and exceptions. The two versions of the stemming algorithm are tested and compared.

Categories and Subject Descriptors
H.3 INFORMATION STORAGE AND RETRIEVAL. H.3.1 **[Content Analysis and Indexing]**: *Linguistic processing* H.3.3 **[Information Search and Retrieval]:** *Clustering, Information filtering, Search process, Selection process.*

General Terms
Algorithms, Measurement, Documentation, Performance, Design, Experimentation, Languages, Theory.

Keywords
Stemming algorithm, stemmer metrics, Greek language, performance evaluation metrics, Natural Language Processing (NLP), Information Retrieval (IR).

1. INTRODUCTION
Stemming algorithms are used in the fields of Information Retrieval (IR) [1, 2, 3] in general and Natural Language Processing (NLP) [4, 5, 6] to improve precision and recall. We present the construction and evaluation of a stemming system and algorithm for the Greek language according to the Modern Greek grammar [7]. Overall, an accurate Greek stemmer can be used for various purposes in IR and Morphological Analysis. This stemming system can help to obtain more and 'better hit' results during searching and retrieving information. According to research and measurements about Greek language stemming, a Greek stemmer on the Web will provide more specific search results [8]. We also provide a library version of the algorithm written in PHP. By implementing a PHP algorithm, our aim has been to provide a stemmer that can directly be used by the engine of any web application, for any kind of web search or linguistics. Other programming languages such as Javascript or even the more powerful like C and C++ lack this ability [9, 10]. This work, which is available under an Open Source licence, will lead to a more powerful, more complete and more consistent Greek stemmer that can directly be used and modified by others.

We first provide a summary of the Ntais' stemming algorithm [11]. Next we introduce some stemmer performance metrics that will be used during our evaluation in order to compare the output of our original stemmer and its modified version. The design of the existing algorithm and its extensive list of rules are provided through references to online material with free access [11]. We describe the improvements incorporated in the re-designed algorithm along with the new set of rules and exception lists. A detailed account of these can also be found in [12].

PCI '16, November 10-12, 2016, Patras, Greece
 ISBN 978-1-4503-4789-1/16/11…$15.00
DOI: http://dx.doi.org/10.1145/3003733.3003775

2. BACKGROUND RESEARCH
There exist some stemming methods for Greek texts, presented since the mid-90s. These methods are parts of more extended work about morphological analysis and information retrieval from various texts and cannot be considered as rule-based stemmers. Considering past research on Greek language stemming [8, 13, 14], researchers agree that specific grammatical rules can improve the effectiveness on information retrieval from Greek texts. We formulated the following research questions:

RQ1: Which specific grammatical/syntactical rules could comprise a stemming algorithm that will lead to the development of an effective Greek stemmer?

RQ2: Up to which point the addition of more grammatical/syntactical rules and exceptions improves the precision of the stemming algorithm used in the Greek stemmer?

The resulted initial stemming algorithm is herein implemented using JavaScript language as a web based application and works through a simple web-site [15], as well as its successor [16].

3. DESIGN AND ENHANCEMENT

3.1 Towards Ntais' Stemming Algorithm

For the design of the Greek stemmer we followed the Porter's algorithm [17] as the literature review on stemmer technology (Table 1) revealed that Porter's algorithm was probably the most reliable one. That algorithm was developed for the English language. The Greek stemmer structure follows the simplicity and the directness of Porter's rules. Figure 1 illustrates the generic overview and straightforward structure of our approach.

Figure 1. Stemming Approach

The research and decision about this was also based on the previous work about Greek stemming and its effectiveness. The development tool was JavaScript, an open-source script language, freely available on the Web. For the evaluation of the stemmer we used the Greek keyword dictionary, kindly provided by the National Centre of Scientific Research "DEMOCRITOS" [18] and a random word corpus.

First, for the sake of simplicity, we only used capital letters as past research [8, 13] indicated that it would be very difficult to try to solve the problem of the "moving" tone-mark on the stems of the Greek words. 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 [7]. In these tenses the stem changes formation as well and this is why there are two stems for every verb. Therefore, we uphold this distinction and we accept that a verb has a different stem in the past tenses from the other tenses. Table 1 provides a summary of the overall information and comparison on stemming technology.

Table 1. Stemming Algorithms – Summarised Information

Author (year)	Language/Web Availability	Execution Steps	Weaknesses & Limitations
Lovins (1968) [19]	EN / Yes	2	Aggressive with short stems and words
Porter (1980) [17]	EN / Yes	5	Quite aggressive and produces over-stemming [20]
TZK (1995) [8]	GR / No	2	Does not handle all suffixes
AMP (2001) [14]	GR / No	4	Unable to handle compound words
Ntais (2006) [11]	GR / Yes	29	Outperforms TZK and AMP [11]. New and untested; handles capital letters; under-stemming errors[12]
CST (2009) [6]	GR and others / No	Prefix/ Infix/Suffix	Relatively new; no criticisms so far

3.2 Towards Saroukos' Stemming Algorithm

One of the aims of this work was to test the original stemmer in combination with a search engine, and Google's search engine was a candidate. A web interface that would feed Google with modified, stemmed queries and unmodified ones could easily be built. The results of both modified and unmodified queries could then be compared. Unfortunately the application of a stemmer in a web search engine was beyond the time limitations of this project. It was also unclear whether Google is already utilizing any kind of stemming techniques for Greek. In a previous web search engine evaluation [4], it was pointed out that Google returns a different number of results for different variations of the word "Athens" (Αθήνα: Athens, Αθήνας: of Athens, Αθηνών: of (the city of) Athens). The difference in results can only imply that no stemming is used. Despite that, there are reports e.g. from Google in 2003 [21] that some form of stemming is being conducted although it is unclear how extensively. In addition, Paice [22] suggests that evaluating a stemmer solely in terms of IR is incomplete since IR is only one field that stemming can be applied and "*...gives no insight into the specific causes of errors*".

In order to evaluate both Ntais' algorithm and its revised stemming algorithm, we executed both in batch mode against a collection of more than half a million Greek words. Both algorithms stemmed the input words from the text, and formed groups of words that had the same stem. The reader can peruse some of these execution step tests in [12].

3.3 Alternative Stemmer Designs and Using Stemmer Performance Metrics

Before proceeding to this work we examined alternative stemmer techniques, e.g. computational process metamodelling and formal language theory [23, 24] and other for redesign, which were rejected mainly because of their application domain incompatibility. A detailed account of the reasons for rejection can be found in [12]. For this decision we additionally re-considered the summarized information behind each and every stemming approach available in Table 1. In order to evaluate the existing stemmer and measure its effectiveness, we used the Frake's Metrics [25] for stemmer strength and Error metrics [26]. Since the strength of a stemmer can affect the precision and recall in queries, Frakes defines a set of metrics that help to compare algorithms by having the algorithms stem the same texts and compare the results of the following metrics: *i) The mean number of words per conflation class; ii) index compression factor; iii) the number of words and stems that differ*, and *iv) the median and mean modified Hamming distance.* There are two clearly distinct error metrics categories concerning stemmers, *under-stemming* and *over-stemming*. [12, 25, 26].

One other approach to stemming is to use lemmatization that means to remove the inflection, (suffix) and create the base or normal form of the word the so called lemma. This has been carried out for Greek, among other languages, including both prefix, (infix for some languages), and suffix removal for Greek obtaining 90% accuracy. The lemmatization algorithm has been trained on examples. [6]. Other approaches to text stemming regarding performance and other challenges in Greek and other languages can be found in [27, 28, 29].

4. PERFORMANCE AND IMPROVEMENT

4.1 Evaluation of Ntais' Algorithm

We first ported the algorithm in PHP. We implemented a set of helper applications that will be using directly the algorithm and will keep statistics about the returned stems. The operating system used for the evaluation was Gentoo Linux but because of the portability of PHP and our style of coding the source code is portable and can be used in any platform that PHP is ported to. Our evaluation commenced by executing our port of Ntais'

stemmer against a list of Greek words. We set the application in a manner that words with common stems would be grouped into conflation classes and then the output be would be directed to a text file. This text file was examined manually for under-stemming and over-stemming errors. We used modified Hamming Distances in order to find similar stems. We concluded that two stems with a modified Hamming Distance of four or less can be possibly merged into one, indicating an under-stemming or over-stemming error of the stemmer that generated them.

4.2 New Features and Properties

4.2.1 The Introduction of Stop-Word Elimination

Stop-word removal is one of the most commonly used techniques in IR [1, 2, 3, 4]. We use stop-word elimination in order to improve the performance of the stemming algorithm. The stop-word list mainly contains words of length of at most four letters. In our modified algorithm, stop-word elimination is the first step of execution, a step that not only produces better results but also improves the running time of the algorithm. The initial algorithm had solved this problem by processing only words of 4 letters or more. Although this approach left just a few words of 3 that could be stemmed unprocessed, we decided to add a stop-word list of more than 500 words, in order to increase precision.

4.2.2 The Addition of More Grammatical Rules

We created a set of helper applications that directly use our implementations of both Ntais' and our modified algorithm. One of these applications uses as input a list of words and creates conflation classes according to the stems returned by the stemmers. These classes were manually checked for over-stemming and under-stemming errors in a manner similar to previous literature [11, 26]. According to the results, more suffixes were added in order to deal with under-stemming. As pointed out in [11], the introduction of more rules for additional suffixes raises stemming errors due to over-stemming. In dealing with this, we added more exceptions in order to deal with over-stemming and keep precision at an acceptable level.

4.2.3 The Introduction of Lower Case Letters

The initial stemming algorithm of Ntais only accepts as input words in upper case letters, as mentioned in earlier sections and in [11]. Our improved algorithm, Saroukos' algorithm [12] is capable of handling words given in any case, upper, lower or combinations of both. The main body of the algorithm remains unchanged and all rules are still in capital letters. Before returning the stem of the given word, a final alteration of the stem occurs as the algorithm consults the case of each letter on the stem, and alters the case of a letter if needed. In our implementation the problem of the "moving" tone-mark still remains [8, 11, 12, 13], but we decided to treat both upper case and lower case words.

4.3 The Final Saroukos' Algorithm

After careful examination of the output of the original stemmer, we tried to incorporate as many modifications as possible. However, an addition of a rule that corrects some errors may create other errors, unless an appropriate exception list is also created. Nevertheless, we added more rules in order to correct wrong patterns that kept appearing in the output. One striking example was the omission of any rules for suffixes that appear in Past Continuous (***ΙΖΑ, ΙΖΕΣ, ΙΖΕ, ΙΖΑΜΕ, ΙΖΑΤΕ, ΙΖΑΝ***) and past tenses in general. This detailed work appears in [12, Table 8].

5. FINAL ALGORITHM'S EVALUATION

After the improvement modifications of Ntais' algorithm [11], we evaluated the revised algorithm, Saroukos' algorithm [12]. We used the same word list and the same applications we created for testing the initial Ntais' algorithm. We made sure that the statistics produced can be comparable. Although the two stemmers leave unchanged roughly the same number of words, our modified version produces fewer and bigger conflation classes by altering more letters in every word, on average. Table 2 presents summary statistics gathered after executing both Ntais' and Saroukos' algorithms against a list of 574,621 Greek words.

Table 2. Comparison of the Original and Revised Algorithms

	Original (Ntais)	**Revised (Saroukos)**
Mean number of words per conflation class	4.055	5.664
Index compression factor	75.34%	82.34%
Ratio of unchanged to total words	2%	2%
Mean modified Humming Distance	2.441	2.916
Median Modified Humming Distance	2	2
Correct Stems (sample of 12468 words)	10.885 (87.3%)	11.669 (93.52%)
Distribution of Stemming errors per algorithm		
Understemming Errors	88.44%	23.67%
Overstemming Errors	11.56%	76.33%
Number of different stems generated by the two stemmers (sample of 574.621 words)	35.885 (6.24%)	

The majority of the errors of the initial algorithm had to do with under-stemming (88.44%). The new algorithm produces more over-stemming errors (76.33%) despite the fact that the total number of errors is reduced. The two stemmers produced 35,885 different stems for the same list of words. The number of execution steps was increased from 29 in the original algorithm of Ntais to 42 in Saroukos' algorithm. 10 of the new execution steps have to do with the 72 newly added stems, while the remaining 3 deal with stop-word removal and lower to upper and upper to lower case treatment. Although the number of the steps was increased by approximately 44%, the new algorithm by Saroukos now executes 23.17 steps on average. The reason is that while the original algorithm always executes all of its 29 steps, the modified algorithm returns the correct stem and then exits earlier if the remaining rules are not going to modify the word any further.

6. CONCLUSIONS AND FUTURE

We gradually constructed and incrementally tested and improved the only Greek language stemmer currently available and open to everyone. Our new and improved stemming algorithm returns more correct results than its predecessor. The under-stemming and over-stemming errors are less. The new algorithm is more complete since it supports most of the grammatical tenses and stems correctly suffixes (like diminutives and other) not included before. Due to the PHP implementation language, our implementation can be used by any web or non web application for stemming of Greek words. Adding more suffixes is attainable but the effort required for each additional suffix increases geometrically. The initial algorithm already deals with the majority of suffixes found in the Greek grammar. Our latest algorithm, like its predecessor, is not dealing with the moving tone-mark issue. The stemmer can be enhanced by adding more suffixes and exceptions. In addition to the 158 suffixes of the initial algorithm, we added rules for 72 more.

7. REFERENCES

[1] Baeza-Yates, R. and Ribeiro-Neto, B. 1999. *Modern Information Retrieval.* Addison Wesley, New York.

[2] van Rijsbergen, C.J. 1979. *Information Retrieval* Butterworths, London.

[3] Risvik, K. M, Mikolajewski T. and Boros P. Q. 2003 Segmentation for Web Search. In *Proceedings of the 12th International World Wide Web Conference*, 52.

[4] Lazarinis, F. 2005. Do search engines understand Greek or users requests "sound Greek" to them ? In: M. Beigbeder and W.G. Yee (eds) *Open Source Web Information Retrieval* Workshop in conjunction with IEEE/WIC/ACM International Conference on Web Intelligence & Intelligent Agent Technology. Compiegne, France, 19 Sep. 43-6.

[5] Lazarinis, F. 2007. Lemmatization and stopword elimination in Greek web searching. *Proceedings of the 2007 Euro American Conference on Telematics and information Systems EATIS '07*, ACM, New York, NY.

[6] Jongejan, B. and Dalianis, H. 2009. Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike. *Proceeding of the ACL-2009, Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics* and *the 4th International Joint Conference on NLP of the Asian Federation of NLP*, Singapore, pp. 145-153.

[7] Triantafyllidis, M. 1941. *Modern Greek Grammar*. Institute of M Triantalyllidis, Greece.

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

[9] Lerdorf, R & Tatroe, K. 2002. *Programming PHP*. O'Reilly, Sebastopol, CA.

[10] Flanagan D. 2004. Javascript: The Definitive Guide. O'Reilly, Sebastopol, CA.

[11] Ntais, G. 2006. MSc Thesis. *Development of a stemmer for the Greek language*. MSc Thesis at Stockholm University / Royal Institute of Technology. (https://people.dsv.su.se/~hercules/papers/Ntais_greek_stemmer_thesis_final.pdf).

[12] Saroukos, S. 2008. *Enhancing a Greek Language Stemmer Efficiency and Accuracy Improvements* MSc Thesis, University of Tampere. (http://uta32-kk.lib.helsinki.fi/bitstream/handle/10024/80480/gradu03463.pdf?sequence=1).

[13] 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.

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

[15] Ntais, G. 2008. Online Greek Stemmer. Web Interface implemented in Javascript. (Last Retrieved: 14/05/2016). http://people.dsv.su.se/~hercules/greek_stemmer.gr.html

[16] Saroukos, S. 2010. Online Greek Stemmer. Web Interface implemented in PHP. (http://saroukos.com/stemmer/)

[17] Porter, M. 1980. An algorithm for suffix stripping. *Program*, 14(3), 130-137.

[18] Petasis, G., Karkaletsis, V., Farmakiotou, D., Androutsopoulos, I. and Spyropoulos, C.D. 2003. A Greek Morphological Lexicon and its Exploitation by NLP Applications. LNCS, vol.2563, Eds: Y. Manolopoulos *et al. Advances in Informatics - Post-proceedings of the 8th Panhellenic Conference in Informatics*, pp. 401-419.

[19] Lovins, JB. 1968. Development of a stemming algorithm. *Mechanical Translation & Comp. Linguistics*, 11, 1. 22-31.

[20] Carlberger, J., Dalianis, H., Hassel, M. & Knutsson, O. 2001. Improving Precision in Information Retrieval for Swedish using Stemming. In *Proc. of NODALIDA 01-13th Nordic Conf. on Computational Linguistics*, May 21-22, Uppsala.

[21] [Google, 2003] Google Starts Auto Stemming Searches. http://www.searchengineshowdown.com/blog/2003/11/google_starts_auto_stemming_se.sh (Retrieved 04/05/2008).

[22] Paice, D. C. 1994. An Evaluation Method for Stemming Algorithms. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 42-50.

[23] Berki, E. 2001. ***Establishing a scientific discipline for capturing the entropy of systems process models: CDM-FILTERS - A Computational and Dynamic Metamodel as a Flexible and Integrated Language for the Testing, Expression and Re-engineering of Systems.*** Ph.D. Thesis, University of North London.

[24] Lewis, R. H. and Papadimitriou, H. C. 1998. *Elements of the Theory of Computation*. Prentice Hall, Upper Saddle River.

[25] Frakes W. B. 2003. Strength and Similarity of Affix Removal Stemming Algorithms. *ACM SIGIR Forum*, 37, 1 26 – 30.

[26] Alvares, R. V, Garcia, A. C. B. and Inhaúma, F. 2005. STEMBR: A Stemming Algorithm for the Brazilian Portuguese Language. *Proc. of the12th Portuguese Conf on Artificial Intelligence*, EPIA 2005, LNAI 3808, 693-701.

[27] Adam, G., Asimakis, K., Bouras, C. and Poulopoulos, V. 2010. An efficient mechanism for stemming and tagging: the case of Greek language. In: *Proceedings of the 14th international conference on knowledge-based and intelligent information and engineering systems*, pp 389–397.

[28] Singh, J. and Vishal, G. 2016. Text Stemming: Approaches, Applications, and Challenges. *ACM Comput. Surveys*. 49(3).

[29] Moral, C., de Antonio, A., Imbert, R. & Ramírez, J. 2014. A survey of stemming algorithms in information retrieval/ *Information Research*, 19(1).