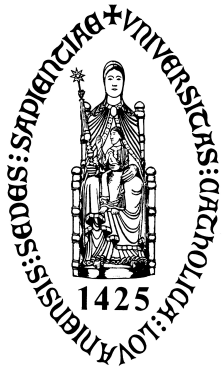
XAVIER GOAS AGUILILLA

A CORPUS-BASED APPROACH TO THE LANGUAGE OF THE PAPYRI



Master's thesis submitted in partial fulfilment of the requirements for the degree of

MASTER OF ARTS IN DE TAAL- EN LETTERKUNDE

Supervisor: dr. T. Van Hal
KU Leuven
Faculty of Arts
Department of Greek Studies

Leuven, 2012

E-MAIL:
xavier.goas@student.kuleuven.be

ABSTRACT

SAMENVATTING

Deze masterpaper bevat **44038** tekens.

PREFACE

The papyri are an invaluable resource for documenting the history and evolution of the Greek language. The recently published *The Language of the Papyri* [Evans and Obbink, 2010] has placed the spotlight firmly on the potential of this field while at the same time pointing out the regrettable lack of recent scholarly work available in it.

Though there is now a near-comprehensive collection of texts available that is well-formatted and can easily be converted into an adequate corpus for linguistic research, comparatively few scholars are interested in exploiting the available resources; and none, to my knowledge, have attempted to do so.

A first necessity is, of course, a broad *status quaestionis* yet in a less traditional sense than one might expect: the focus here lies more upon technical and technological concerns than specialised monographs for which data was gathered manually. Nevertheless, I have for the sake of completeness chosen to include a comprehensive bibliography of the field for clear reference. The term bibliography would be rather less appropriate in describing the array of databases and linguistic tools available to us - it is rather a select catalogue documenting the most relevant items in the instrumentarium in some detail while providing a briefly annotated list of other useful resources.

A second chapter is dedicated

1. a survey of previous studies on the language of the Greek papyri and an analysis of the methodology used therein;
2. a critical evaluation of the methods used by aforementioned studies, with equal attention for both positive and negative aspects of the applied method;
3. the development of a methodology based on this evaluation that is fit for use in corpus-based grammatical and linguistic studies;
4. an analysis and evaluation of the available tools in the field of papyrology for such studies, followed by suggestions for possible improvements to these tools;
5. the study of select grammatical and linguistic questions concerning the language of the Greek papyri using the methodology and toolset developed previously — essentially a practical test of the previous findings.

ACKNOWLEDGEMENTS

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION AND PRELIMINARIES

The study of the language of the papyri has in the past thirty years seen little evolution until the recent appearance of Evans and Obbink's *The Language of the Papyri* [Evans and Obbink, 2010], which has placed the subject in the spotlight again. Twentieth-century scholarship on the topic, though still useful for those interested in the study of the papyri for historical purposes, is either antiquated, limited in scope or incomplete (see *infra*). Despite this, the papyri are useful source material for the history and evolution of the Greek language, as they contain not only official texts but private documents as well, whose linguistic features and peculiarities have the potential to foster new insights into the nature of colloquial Greek.

1.1 THESIS

The following thesis intends to prove that it is possible to generate basic linguistic annotation for a large digitalised corpus of papyri in ancient Greek using readily available tools and techniques with minimal technical overhead. Such a corpus could be a boon to scholars interested in the Greek of the papyri, as it would facilitate, for instance, the creation of linguistically sound grammars and lexica.

1.2 PRELIMINARIES

The following section will provide a background sketch, consisting of a short overview of previous efforts and an elucidation of some key concepts.

1.2.1 The language of the papyri

The papyri began to be studied linguistically not by papyrologists and historians, but rather by Bible scholars and grammarians interested in their relevance in the development began to koinê Greek, particularly that of the New Testament. G. N. Hatzidakis, W. Crönert, K. Dieterich, A. Deissmann, and A. Thumb pioneered the field in the late nineteenth and early twentieth century, spurring a resurgence of scholarship on the topic; an excellent overview of pre-1970s research may be found in Mandilaras [1973] and Gignac [1976 and 1981].

During this period, Mayser begon work on the earliest compendious grammar of the papyri; it limits itself to the Ptolemaic era but

explores it at length and in great detail. The work consists of a part on phonology and morphology, made up of three slimmer volumes, and a part on syntax, encompassing three larger volumes. Its composition seems to have been exhausting: it took Mayser thirty-six years to finish volumes I.2 through II.3, with I.1 only completed in 1970 by Hans Schmoll, at which point the entire series was given a second edition.

When casually browsing through some of its chapters (though casual is hardly the word one would associate with the *Grammatik*) it is remarkable to see that Mayser brings an abundance of material to the table for each grammatical observation he makes, however small it may be. For instance, the section on diminutives essentially consists of pages upon pages of examples categorised by their endings.

This is its great strength as a reference work - whenever one is faced with an unusual grammatical phenomenon in any papyrus, consulting Mayser is bound to clarify the matter; or rather, it was, for the work is now inevitably dated. The volumes published during Mayser's lifetime only include papyri up to their date of publication; only the first tome by Schmoll includes papyri up to 1968. It is still a largely useful resource, but it is in urgent need of refreshment.

After Mayser set the standard for the Ptolemaic papyri, a grammar of the post-Ptolemaic papyri was the new *desideratum* in papyrology. The work had been embarked on by Salonius, Ljungvik, Kapsomenos, and Palmer, only to be interrupted or thwarted by circumstance or lack of resources. Salonius [1927], for instance, only managed to write an introduction on the sources, though he offered valuable comments on the matter of deciding how close to spoken language a piece of writing is. Ljungvik [1932] contains select studies on some points of syntax.

It is in the 1930's that we see attempts to create a grammar of the papyri that would be the equivalent of Mayser for the post-Ptolemaic period. Kapsomenos published a series of critical notes [1938, 1957] on the subject; though he attempted at a work on the scale of the *Grammatik*, he found the resources sorely lacking, as the existing editions of papyrus texts could not form the basis for a systematic grammatical study. The other was Palmer, who had embarked on similar project and had already set out a methodology [1934]; the war interrupted his efforts, and he published what he had already completed, a treatise on the suffixes in word formation [1945].

A new work of some magnitude presents itself two decades later with B. G. Mandilaras' *The verb in the Greek non-literary papyri* [1973]. Though it does not aim to be a grammar of the papyri, it does offer a thorough and satisfactory treatment of the verbal system as manifest in the papyri. Further efforts essentially do not appear until the publication of Gignac's grammar. It is essentially treading in the footsteps of Mayser, only with further methodological refinement and a more limited, though still sufficiently exhaustive, array of examples. The author, for reasons unknown to me, only managed to complete two of the three projected volumes, on phonology and on morphology. The volume on syntax is thus absent, a gap only partly filled by Mandilaras' *The verb in the Greek non-literary papyri*.

Finally, there is the aforementioned *The Language of the Papyri* [Evans and Obbink, 2010], which does not aim to be a work on the same scale as the aforementioned. It is a collection of articles on various topics, the whole of which is meant to illuminate new avenues for future research. A particularly relevant chapter for this thesis is the last one by Porter and O'Donnell [Porter and O'Donnell, 2010], who set out to create a linguistic corpus for a selection of papyri; their tagging approach, however, is manual, and their target corpus limited. The authors also are the creators of `http://www.opentext.org/`, a project aiming for the development of annotated Greek corpora and tools to analyse them; sadly, no progress seems to have been made since 2005.

1.2.2 Corpus linguistics

A[1] corpus or text corpus is a large, structured collection of texts designed for the statistical testing of linguistic hypotheses. The core methodological concepts of this mode of analysis may be found in the concordance, a tool first created by biblical scholars in the Middle Ages as an aid in exegesis. Among literary scholars, the concordance also enjoyed use, although to a lesser degree; the eighteenth century saw the creation of a concordance to Shakespeare.

The development of the concordance into the modern corpus was not primarily driven by the methods of biblical and literary scholars; rather, lexicography and pre-Chomskyan structural linguistics played a crucial role.

Samuel Johnson created his famous comprehensive dictionary of English by means of a manually composed corpus consisting of countless slips of paper detailing contemporary usage. A similar method was used in the 1880s for the Oxford English Dictionary project - a staggering three million slips formed the basis from which the dictionary was compiled.

1950s American structuralist linguistics was the other prong of progress; its heralding of linguistic data as a central given in the study of language supported by the ancient method of searching and indexing ensures its proponents may be called the forerunners of corpus linguistics.

Computer-generated concordances make their appearance in the late 1950s, initially relying on the clunky tools of the day - punch cards. A notable example is the *Index Thomisticus*, a concordance to the works of Thomas of Aquino created by the late Roberto Busa S.J. which only saw completion after thirty years of hard work; the printed version spans 56 volumes and is a testament to the diligence and industry of its author. The 1970s brought strides forward in technology, with the creation of computerised systems to replace catalogue indexing cards, a change that greatly benefited bibliography and archivistics.

It is only in the 1980s and 1990s that are marked the arrival of fully developed corpora in the modern sense of the word; for though the basic concepts of corpus linguistics were already widely used, they could

1 The following section is based *passim* on McCarthy and O'Keeffe [2010].

not be applied on a large scale without the adequate tools. The rise of the desktop computer and the Internet as well as the seemingly ever-rising pace of technological development ensured the accessibility of digital tools. The old tools - punch cards, mainframes, tape recorders and the like - were gladly cast aside in favour of the new data carriers.

The perpetual increase of computing power equally demonstrated the limits of large-scale corpora; while lexicographical projects that had as their purpose to document the greatest number of possible usages could keep increasing the size of their corpora, the size of others went down as they whittled the data down to a specific set of uses of language.

The possible applications of the techniques of corpus linguistics are diverse and numerous; for they allow for a radical enlargement in scope while remaining empirical, and remove arduous manual labour from the equation. Corpus linguistics can be an end to itself; it can, however, assert an important role in broader research. McCarthy and O'Keeffe [2010, p. 7] mention areas such language teaching and learning, discourse analysis, literary stylistics, forensic linguistics, pragmatics, speech technology, sociolinguistics and health communication, among others.

The term 'corpus' has a slightly different usage in classical philology: they designate a structured collection of texts, but that collection is not primarily intended for the testing of linguistic hypotheses. Instead, we have, for instance, the ancient corpus Tibullianum, or modern-day collection, for instance the Corpus Papyrorum Judaicarum, etc. We are primarily interested in the digital techniques used to create linguistic corpora; so let us first take a look at the progress of the digital classics.

1.2.3 The digital classics

Classical philology, despite its status as one of the oldest and most conservative scientific disciplines still in existence today, has in the last fifty years found itself at the front lines of the digital humanities movement. Incipient efforts in the fifties and sixties, mainly stylometric and lexical studies and the development of concordances, demonstrated the relevance of informatics in the classics, an evolution that was at first met with some skepticism, but later fully embraced.

The efforts began with the aforementioned Index Thomisticus, the first computer-based corpus in a classical language; but the first true impetus was the foundation of the Thesaurus Linguae Graecae project in 1972, a monumental project with as its goal the stocking of all Greek texts from the Homeric epics to the fall of Constantinople. Over the years, many functions have been added to this ever more powerful tool; and even in the beginning stages of its development, the TLG garnered praise.

The usefulness of the tool in its current form cannot be overstated: not only does it contain a well-formatted and easily accessible gigantic collection of text editions whose scope and dimensions exceed those of nearly any university library; it also offers all of these texts in a format

that allows for lexical, morphological and proximity searches, as well as including a full version of the Liddell & Scott and Lewis & Short dictionaries. The TLG has become a staple of the digital classics.

Despite this, the TLG is becoming more and more dated as technology progresses. While recent years have seen the rise of Unicode as the standard for encoding ancient Greek, the TLG still uses beta code, a transliteration system designed to only use the ASCII character set, and the texts are stored using an obsolete text-streaming format from 1974, which divides the text in blocks of eight kilobytes and marks the division between segments.

A digitised version of the Liddell-Scott-Jones lexicon has been added to the TLG's web interface, but the texts themselves have not undergone extensive tagging, only lemmatisation. Searching through the database can be done by searching for specific forms of a lemma, or by searching for all forms of a lemma, but this is essentially the limit of the search tool's power; it is not possible to perform a query for all possible lemmata associated with a particular form, i.e. we cannot find all forms which are, for example, an active perfect indicative.

In the wake of the TLG, several notable projects have emerged: Brepols' Library of Latin Texts is trying hard to be for Latin texts what the TLG is for Greek texts; the Packard Humanities Institute has released CD's containing a selection of classical Latin works. In more recent times, the Perseus Project has enjoyed great popularity because of the attractive combination of an excellent selection of classical texts with translations, good accessibility and a set of interesting textual tools, the entire package carrying a very interesting price tag for the average user — it is free to use, and for the greatest part, open source as well.

The databases I have mentioned are quite general in scope; but within the domain of classical philology, other specialised projects exist. Within the field of papyrology, for instance, the digital revolution has taken a firm foothold. Starting with several separate databases, the field has experienced a tendency towards convergence and integration of the available resources, as exemplarised by the papyri.info website, maintained by Columbia University, that integrates the main papyrological databases into a single database.

A great feature of this database is the shell in which all data is wrapped; they are compliant with the EpiDoc standard, a subset of XML based on the TEI standard and developed specifically for epigraphical and papyrological texts. One may access the database's resources through the Papyrological Navigator and suggest corrections and readings through the Papyrological Editor. What's more, all data is freely accessible under the Creative Commons License, crowd-sourced, regularly updated, and can be downloaded for easier searching and tweaking.

In other words, papyri.info has brought the open-source mentality from the computer world into the classics. For our purposes, this open setup is desirable, as the database is not fit for them as it is, but can with some effort be molded into a useful tool.

1.2.4 Natural language processing

Natural language processing (henceforth NLP) is a subdiscipline in computer science concerned with the interaction between natural human language and computers. Its history well and truly starts in the fifties, with a basic concept which has played a great role in natural language processing, and computer science in general, the Turing test. This test, put forth by Alan Turing in his seminal paper *Computing Machinery and Intelligence* [Turing, 1950], evaluates whether a machine is intelligent or not by placing a human in conversation with another human and a machine; if the first human cannot tell the other human and the machine apart, the machine passes the test.

Machine translation systems entered development, though progress soon stalled because of technical limitations and because of methodological obstacles: such systems were dependent on complex rulesets written by programmers that allowed for very little flexibility. Because of the slow return on investments made, funding for artificial intelligence in general and machine translation specifically was drastically reduced throughout the late sixties and the seventies.

A resurgence followed: thanks to advances in computational power and the decline of Chomskyan linguistics in natural language processing, which had been the dominant theoretical vantage point in the preceding thirty years, the eighties were marked by the introduction of statistical machine translation, which is fundamentally based on the tenets of corpus linguistics. Modern natural language processing is therefore situated on the crossroads between various fields: artificial intelligence, computer science, statistics, and corpus and computational linguistics. It looks to be an exciting field for the coming years as its techniques are under constant improvement and ever more present in our daily lives.

Most NLP software is designed explicitly with living languages in mind; English, being a world language and the international *lingua franca*, has enjoyed most of the attention, but other major languages have enjoyed some attention, too. Ancient languages, however, are neglected, presumably due to their often high complexity and the extensive study and analysis to which they have been submitted by skilled scholars. Yet most texts have not been integrated in annotated corpora; and though databases such as the Perseus project contain large swathes of morphologically and sometimes syntactically annotated text, the process has been driven largely by manual labour; to give an exhaustive list is not appropriate here, but another such example which is relevant is the PROIEL project [*PROIEL: Pragmatic Resources in Old Indo-European Languages*], which is also a treebank, i.e. a database of syntactically annotated sentences. It contains data for Herodotus and the New Testament.

There is also a corpus which has been tagged using NLP techniques, whose relevancy for this thesis is high and that I have thus described in the next section on methodology, namely the Perseus corpus under the PhiloLogic interface, run by the University of Chicago.

1.3 METHODOLOGY

In this thesis, we have been largely inspired by two articles by H. Dik and R. Whaling, [Dik and Whaling, 2008 and 2009], in which they document their method for semi-automatically tagging the Perseus project's texts under their own framework, PhiloLogic. They start with a database of analysed forms and a series of tagged texts which they use as initial data to train a decision tree tagger, TreeTagger, developed by Helmut Schmid at the University of Stuttgart, a tool which despite being developed in 1995 has aged well as far as performance is concerned. They achieved remarkable accuracy: with refinements to the training data they achieved 96.2% accuracy during tests on the original training data and 91% accuracy on new data, a result which compares quite favorably when compared to TreeTagger's 97% accuracy when used on German newspaper articles considering the high complexity of ancient Greek and the variety of styles of ancient Greek literature.

It occurred to me that this might be a great method for processing the papyri.info database with a relatively small effort for a high payoff; using data from the Perseus and PROIEL projects, it could be possible to train TreeTagger for both morphology and syntax, apply the resulting parameters to the corpus and thus for the greatest part obviate the need for manual tagging. Given the extent of the corpus (about 50,000 texts containg almost 4,500,000 words), achieving even 85% accuracy would reduce the amount of untagged words to 675,000, many of which I would expect to be proper names or morphologically 'erroneous' forms as are often found in the papyri, data which could itself be analysed with regular expressions and then used to improve the training data.[2]

2 As I set out to verify the originality of my thesis, I found that this statistical approach has been used before for textual criticism! *Vide* Mimno and Wallach [2009], an abstract of which may be found at http://people.cs.umass.edu/~wallach/publications/mimno09computational.txt.

2 THE CORPUS

CONTENTS

2.1 GOALS

Our objectives for the modified corpus is threefold. Firstly, we want to morphologically annotate it with reasonable accuracy; secondly, we also want to add syntactical annotation; thirdly, we want to ensure the corpus is compatible with a broad range of tools.

The former two goals are the central enterprise of this thesis. Morphological annotation will pose little theoretical problems, as the morphological features of Greek are not subject to discussion; but for syntactical annotation, a variety of theoretical frameworks are in existence. The fact of the matter is that I have chosen to take the route which is technically easiest: to use the tenets of dependency grammar. The reason is twofold: on one hand, our training data, the New Testament annotated by PROIEL, itself is structured using a dependency model; on the other hand, dependency trees, in contrast to constituent analysis, assigns each word or morph one node, which is easier to 'digest' for a computer program.

Our third goal is to format the corpus in a way that will ensure broad compatibility. This is desirable for several reasons. A first is that a broadly used format offers many options for data treatment; for instance, comma-separated text files can be read by many programs, from the simplest text editor to the most complex software packages, while XML can very easily be integrated in a web interface or parsed and transformed by a variety of programs. A second is that the corpus is to be published on GitHub for collaborative editing, or possibly pulled into the main papyri.info repository. This depends on our first point: the idea of GitHub is that all projects are open source and can be contributed to by others, which is facilitated by using a format with wide adoption.

2.2 DESIGN

2.2.1 Initial inquiries

Before arriving at the approach described below, I attempted several other possible methods for automatic analysis. A first idea was to integrate the XML files in a PhiloLogic setup. PhiloLogic is a tool developed by the ARTFL project and the Digital Library Development center at the University of Chicago and released under the GNU General Public License. Its original purpose was to serve as the full-text search, retrieval and analysis tool for large databases of French literature.

PhiloLogic has support for TEI XML and boasts an impressive array of features: it can search through text and deliver KWIC results filtered by frequency and metadata, as well as collocation tables and word order information. Furthermore, the development web page cites support for bibliographic backends in MySQL databases, out of the box operation, interoperability across Unix-based systems, etc.

The tool seemed promising; especially the built-in support for XML and Unicode drew my attention, along with the built-in features for linguistic analysis. It essentially would have made my work easier and allowed me to invest more time in investigating a few linguistic topics. In the end, however, the software did not fit my needs in a few respects.

Firstly, PhiloLogic requires Apache, Perl and several CPAN modules, as well as the utilities gawk, gdbm, and agrep. Not a huge amount of dependencies, but still less than using Python and the NLTK, which does not require a server or SQL database to run; also a big problem is the out-of the box incompatibility of the required CPAn modules with 64-bit systems. Setting PhiloLogic up was not as easy as advertised; I had to resort to a virtual machine running 32-bit Debian Linux until I was able to discover a method that enabled compatibility with Mac OS X 10.7, my OS of choice.

Secondly, XML support in PhiloLogic includes standard TEI but has issues with EpiDoc; notably, in my experience, headers were treated as text and EpiDoc's complicated tagset was incorrectly rendered; for instance, original readings and corrections were concatenated in the text browser, an undesirable situation if one wishes to have any hand in our choice of corpus, and a possible source of statistical errors. Developing brand new XSLT stylesheets to convert EpiDoc to a simpler form of TEI markup or to raw text could perhaps have been a feasible solution; but PhiloLogic seemed to behave erratically in general and a first attempt at feeding the system raw text (advertised as one of its capabilities) turned out a spathe of errors.

Thirdly, PhiloLogic's feature set is extensive, but I soon came to realize that it does not offer much more than papyri.info already does; it is essentially a robust concordancing application through a web interface. I wished not only to have a lemmatised corpus searchable by word proximity, but also by word relations, which would shed more light

on the syntax of the Greek of the papyri, as this is arguably the largest gap in scholarship due to the age of Mayser's *Grammatik* and the incomplete state of Gignac's *Grammar*. Furthermore, the tool as available to the public does not integrate the morphological database of Greek developed for Perseus under PhiloLogic — a regrettable point, really, since it was exactly this capacity I wished to exploit in the first place. Unicode search for Greek also seems to be in need of improvement; the search form still requires transcription. Modifying the tool would have been unfeasible given the size of the source code.

Thus I discarded PhiloLogic from my options and set out to find or develop a simpler system.[1] After the technical struggle with PhiloLogic, I decided to focus my efforts on concordancers that could replicate its functions. A quick consultation of Stanford's list of resources [Stanford NLP Group, 2011] for statistical natural language processing and corpus-based computational linguistics directed me to WordSmith Tools, which is regrettably Windows-only and a commercial, closed source program to boot. I set out on a search for open-source alternatives and found a diversity of programs, most of them without the same functionality as WordSmith, and invariably clunky or antiquated — making WordSmith the only option, but even that did not seem viable for my ends.

I continued my search and stumbled across a very interesting piece of software that is enjoying a good amount of popularity as a didactic tool for computational corpus linguistics; the Python NLTK, short for Natural Language Tool Kit, is not a stand-alone application, but rather a set of "open source Python modules for research and development in natural language processing and text analytics bundled with data and documentation" [Natural Language Toolkit, 2012]. These modules implement diverse functionalities useful for natural language processing and allow their easy integration into Python programs. The Python NLTK seemed to offer a more than solid amount of features: corpus readers, tokenizers, stemmers, taggers, chunkers, parsers, classifiers, clusterers, tools for semantic interpretation and metrics were all integrated from the get-go and are easily combined and extended. The NLTK was developed for English, but its extensibility meant it could be applied to ancient Greek with relative ease. A further advantage is the fact that Python is very well-suited to text manipulation and parsing XML, which is an advantage when working with the papyri.info corpus; it is also cross-platform: Python interpreters are available for all major platforms.

Now that I had a solid tool for interpreting corpora, the only issue remaining was to construct a linguistically annotated corpus out of the papyri.info corpus. The extent of the corpus makes extensive manual tagging by a single person unfeasible; thus I set out to look for an automated tagger which could be used, immediately or with some effort, on ancient Greek. A variety of taggers seem to be available,

1 I also considered setting up a mirror of papyri.info on a personal server and modifying the search functionality; this turns out not to be a task for the weak-hearted. The required setup causes substantial overhead for our purposes, too much to be a reasonable solution.

but I settled on TreeTagger, developed by Helmut Schmid, for several reasons:

2.3 TECHNICAL

2.3.1 Requirements

To create the annotated corpus as I have, it is necessary to have the following installed (older versions may work but have not been tested):

- a UNIX-like operating system, i.e. Mac OS X, any variety of Linux or BSD (using Windows should also be possible, since all tools used are portable and cross-platform, but the file location hierarchies in some scripts will not work);
- Python 3.2.3, found at `http://www.python.org/` (some scripts require Python 2.7.3: they are indicated as such at the top of the relevant script), with some additional libraries:
  - the Python Natural Language Toolkit, found at `http://nltk.org/`, which is only compatible with Python 2.5.x to 2.7.x (sparingly used);
  - the numpy library, which contains a vast array of mathematical functionality, found at `http://numpy.scipy.org/`.
- Perl 5.x `http://www.perl.org/`;
- Saxon-HE 9.x, found at `http://saxon.sourceforge.net/`;
- Helmut Schmid's TreeTagger, found at `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/`;
- SQLite 3, found at `http://www.sqlite.org/`.

All the above, excluding Mac OS X, are freely available.

2.3.2 Principles of statistical natural language processing

The crux of the proposed method is the use of the aforementioned TreeTagger, developed by Helmut Schmid, which can lemmatize and tag at the same time, followed by another iteration using the Stanford Parser to analyse sentence dependencies. TreeTagger can be trained for any language, and its efficiency for highly inflected languages such as ancient Greek is due to its combination of two tagging methods, namely n-gram tagging and binary decision tree tagging, the combination of which makes TreeTagger the fastest part-of-speech tagger around. The Stanford Parser is the state of the art in sentence parsing.

The following section, therefore, is intended to provide some theoretical background on statistical natural language processing. I have based myself upon Manning and Schütze's *Foundations of Statistical*

Natural Language Processing [Manning and Schütze, 1999], the current standard reference work, as well as Koshy [2004] and Hopcroft *et al.* [2001] for concepts relating to automata theory, as well as on Bod *et al.* [2004] for the paragraphs on statistics. Furthermore, to illuminate some aspects of TreeTagger's workings, I have also integrated Schmid's articles on tagging methodology as he applied it in TreeTagger [1994 and 1995].

Context-free grammar and pushdown automata

The basis of all statistical natural language processing is the conceptualisation of natural language within the Chomsky hierarchy as a formal language generated by a **stochastic context-free grammar**, that is to say, a context-free grammar whose production rules are augmented by a probability. In computer science, such a language is termed a context-free language and defined as any language that can be recognized by a **nondeterministic pushdown automaton**. A nondeterministic pushdown automaton is understood to be a nondeterministic finite state automaton with access to an infinite stack, a finite state automaton being an automaton which is in a state that can transition into another state when triggered by input. Put more simply, a nondeterministic pushdown automaton possesses the following:

- a **stack**, that is, a string of symbols which functions as the automaton's memory. The automaton only has access to the leftmost symbol in this string, a symbol which is dubbed the top of the stack; the top of the stack can be used to determine which transition the automaton will make next;
- an **input tape** whose symbols are scanned one by one just like a finite-state automaton;
- a **finite state control unit**, which controls the state of the automaton. Normally this is done in response to input, but it is possible in some types of automata to transition into a new state without any input, as is the case for pushdown automata; this type of transition is termed ϵ-transition or λ-transition.

A simple example of finite state automaton is a coin-operated turnstile: it has two possible states, locked and unlocked, and two possible inputs, the insertion of a coin or a push. Depending on the state the machine is currently in, each of these inputs will trigger either a change or leave the machine in its current state. For instance, if the turnstile is unlocked, a push will make it turn and lock again; inserting a coin will do nothing. Conversely, when it is locked, a coin will unlock it, while a push will still leave it locked. This type of automaton is deterministic, that is, there is only one next possible state to transition to. A nondeterministic finite state automaton would for example be a vending machine, which functions in a similar way but allows for transitions to various states depending on input; different combinations of coins and button presses will unlock different mechanisms in the machine and let a specific item fall into the bottom compartment.

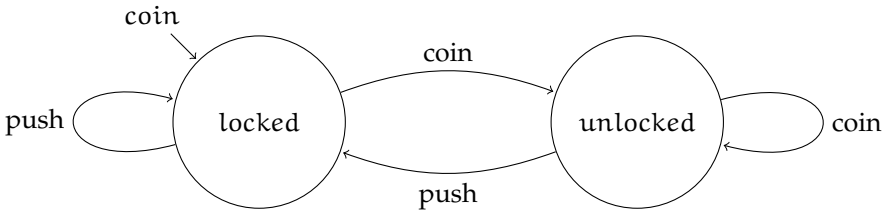


Figure 1: Transition diagram for a deterministic finite state automaton modeling a turnstile. Adapted from Koshy [2004, p. 762].

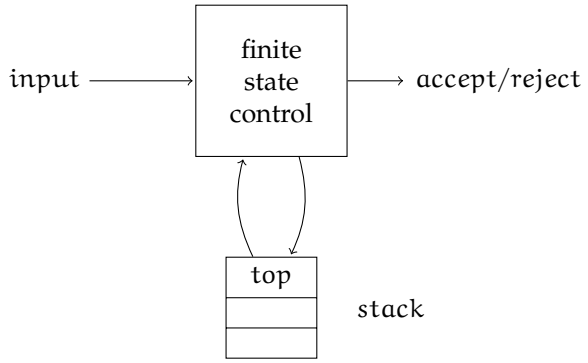


Figure 2: General structure of a pushdown automaton. Adapted from Hopcroft *et al.* [2001, p. 220].

When a stack is added to this setup, the automaton is essentially provided with an infinite memory. Transitions will now be based upon the current state, the input symbol and the symbol at the top of the stack. An ϵ-transition is possible as well, with ϵ replacing the input symbol. Thus, any transition now consists of the following: the consumption of input, the transition to a new state itself, and the replacement of the top of the stack by any symbol (it is even possible for the current symbol to remain in place).

Natural language fits within this paradigm. Given an alphabet $\Sigma = \{a, b, c, d, \dots\}$ or even, in the case of Greek, $\{\alpha, \beta, \gamma, \delta, \dots\}$ the initial state and top of the stack will give rise to a sequence of characters or strings formed from the alphabet, which may be a word, a sentence, an paragraph, ... In other words, context-free grammar provides a simple yet precise method for mathematically describing and studying the rules which govern the construction of natural language from smaller blocks; one can parse generated strings (in themselves context-free languages) and by induction assemble a grammar. It is also possible to feed a string as input to a pushdown automaton applying a context-free grammar; this will show whether the string is acceptable by the rules of the grammar or not.

Stochastics and statistics

As mentioned above, statistical language processing adds a stochastic element to this model of language; not only are the possible transitions analysed, probabilities are also assigned to each of them. Within probabilistics, two interpretations exist: the more widely-known one is **frequentism**, also known as **objectivism**. It is the classical brand of statistics; objectivism views probabilities as realities that can be measured by relative frequencies obtained in experiments. The opposing interpretation is termed **subjectivist** or **Bayesian** (after Thomas Bayes, discoverer of its founding theorem), and views probabilities as degrees of belief or uncertainty. In other words, while frequentism relies on experiment and trial, Bayesianism relies on the observer's judgement. The main interest here lies in Bayesian stochastics, so let's first direct our attention to Bayes' theorem, which is as follows in its basic form:

$$P(H|E) = \frac{P(E \cap H)}{P(E)}. \tag{1}$$

Where $P(H \mid E)$ denotes the probability of a hypothesis H given evidence E. **Conditional probability**, the probability of a given event given some knowledge, is an important notion here. Two kinds of conditional probability exist: *a priori* and *a posteriori* probability, the former denoting the probability before considering additional information, the latter denoting the probability after considering it. The conditional probability of an event A given an event B which has occurred is as follows:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, \tag{2}$$

from which can be derived (using the fact that set intersection is symmetric, *ergo* $A \cap B = B \cap A$) that

$$P(A \cap B) = \quad P(B) \quad P(A|B) = \quad P(A) \quad P(B|A). \tag{3}$$

We can now make a substitution, by which we obtain that

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}. \tag{4}$$

A classic demonstration of Bayes' rule can be given by the solution to the famous Monty Hall problem, a probabilistic puzzle named after the host of the 1970's American talk show *Let's Make a Deal*, which I shall use as an illustration. The problem is set up as follows.

A man is on a quiz program and given a choice between three doors to open, of which one will reveal a car and the others nothing; the man gets to take home whatever is behind the door he picks. The man

picks a door; then the host opens another door, always the one without a car behind it, and asks the man whether he'd like to change doors. The question now is the following: should he switch?

Most people would not switch due to the erroneous belief that the probability of picking any door has not changed from 0.33 for each door, or that the probability has now become 0.5 for each remaining door. But the fact is, he should switch! By eliminating the third door, the host has effectively joined all probabilities beside the one for the car being behind the door the player has originally picked, that is, switching doors at this point will double the probability of winning the car to 0.66.

Bayes' rule is at play here. Given A, B and C, the respective probabilities of the car being behind each door (which are all equal to 0.33), and given I, the initial information, we now have:

$$\mathrm{P}(A|\mathrm{I}) = \qquad \mathrm{P}(A|\mathrm{I}) = \qquad \mathrm{P}(A|\mathrm{I}) = \frac{1}{3}. \tag{5}$$

Now add a factor H, which indicates the opening of the third door C by the host. By Bayes' theorem the following can be stated:

$$\mathrm{P}(A|\mathrm{HI}) = \frac{\mathrm{P}(A|\mathrm{I})\mathrm{P}(\mathrm{H}|A\mathrm{I})}{\mathrm{P}(\mathrm{H}|\mathrm{I})}. \tag{6}$$

Given the fact that P(A | I) = 0.33, we will focus on P(H | AI) and P(H | I), the probabilities of respectively the host opening door C with the car at A and the host opening door C given only I. The former is simple: if the car is behind door A, then we must admit that the host can only open two doors, B or C, with the probability being 0.5 for both events; *ergo*,

$$\mathrm{P}(\mathrm{H}|A\mathrm{I}) = \frac{1}{2}. \tag{7}$$

The remaining probability, that of the host opening door C given only info I, is the following:

$$\mathrm{P}(\mathrm{H}|\mathrm{I}) = \mathrm{P}(\mathrm{H}|A\mathrm{I})\,\mathrm{P}(A|\mathrm{I}) + \mathrm{P}(\mathrm{H}|\mathrm{BI})\,\mathrm{P}(\mathrm{B}|\mathrm{I}) + \mathrm{P}(\mathrm{H}|\mathrm{CI})\,\mathrm{P}(\mathrm{C}|\mathrm{I}), \tag{8}$$

which is due to the fact that A, B and C are mutually exclusive and exhaust all possibilities. For P(H | BI), the probability of the host opening C when the car is behind B and we've picked A, we know that if the car is behind door B and we have picked door A, the host has no choice but to open C, making P(H | BI) = 1. P(H | CI), the probability of the host opening door C when the car is behind it and we have picked

A, is obviously 0 due to the rules of the game. Filling in the known probabilities gives us the following:

$$P(H|I) = \frac{1}{2} \cdot \frac{1}{3} + 1 \cdot \frac{1}{3} = \frac{1}{2}. \tag{9}$$

We now have all requisite probabilities to calculate the final probability of winning when we don't switch.

$$P(A|HI) = \frac{\frac{1}{3} \cdot \frac{1}{2}}{\frac{1}{2}} = \frac{1}{3}, \tag{10}$$

which by exclusion gives us the probability of winning when switching:

$$P(B|HI) = \frac{2}{3}. \tag{11}$$

This surprising puzzle sheds some light on how Bayes' theorem works; the effect of applying it is essentially that the order of dependence between events is swapped. It lets us calculate the probability of P(A | HI) in terms of P(H | AI), or to put it more informally, to use the probability of the host opening door C when we have picked A to calculate the probability of the car being behind door A given the host has opened door C.

Our primary interest is in its use for natural language processing. To expound upon this, we first need to introduce the notion of Bayesian belief networks. A good definition is given in Bod *et al.* [2004, p. 80]:

> Bayesian belief networks are data structures that represent probability distributions over a collection of random variables. A network consists of a directed acyclic graph, in which nodes represent random variables (unknown quantities) and the edges between nodes represent causal influences between the variables.

We are particularly interested in one specific type of Bayesian belief network – the hidden Markov model. A Markov model is a stochastic process which has the Markov property; that is to say, its behaviour does not depend on previous process states, only on the current state. Markov models are classified according to two binary axes: autonomous versus controlled and fully versus partially observable. The hidden Markov model is autonomous and partially observable; this makes it an excellent model for analysing natural language, as it shares both these properties. A hidden Markov model is in itself a stochastic finite state automaton, with each state generating an observation and the states themselves being hidden (an important fact; it is here where Bayes' rule comes into play). Using our observations, we can make probabilistic inferences about the next transition that will happen in

the process, thus allowing us up to build up a collection of parameters for state transitions that can be used to predict our next observation.

An additional important factor is that of the order of the model; the order of the model denotes the items the model has in memory. For instance, a first-order model will contain exactly one item in its memory, a second order two, etc... A natural language expression mapped to a model of order 0 will simply give the frequencies for each item in the expression, whereas elevating the order allows the establishment of probabilistic relations between sets of items, bigrams (two items) for first-order models, trigrams (three items) for second-order models and so on. Given sufficient training material, a program analysing such models will be able to reproduce similar language (it has to be said, often with little regard for making sense but with perfect grammaticality) and be able to analyse new sentences as well based upon the behaviour of previously stored models.

Now, there is one problem in the picture, and that is that such programs require **sufficient** training material, and if not enough material is given, inaccuracies will quickly arise. Such a problem easily poses itself in the case of high morphological complexity, where tokens have much smaller frequencies than in morphologically simple languages such as English; since ancient Greek is of considerable morphological complexity, most taggers working with this method will give inaccurate results. On the other hand, TreeTagger implements an extra technique, that of binary decision trees, which allow it to store infrequent analyses in its memory and use them when appropriate.

Our other tool, the Stanford Parser, also relies upon hidden Markov models; its function is to analyse the dependency structure of sentences using unlexicalized probabilistic context-free grammars, the term 'unlexicalized' implying that the use of class words to label words is reduced to only the syntactically most important word classes; i.e. nouns are often not classified. This increases its speed and still retains a high degree of accuracy; it is also widely known, well-supported and very extensible thanks to the fact that it is written in Java, which opens its up to be connected with a variety of other programs and libraries for various purposes, e.g. data visualisation tools, statistical libraries, most other programming languages, ...

I will now give an overview of how I extracted the training data.

2.3.3 Extracting training data

To train TreeTagger and the Stanford Parser, we need annotated data that can be analysed by the programs. We have chosen the following dataset:

1. For TreeTagger (lemmatisation and morphological tagging):
    - as a training file: the Greek New Testament and Herodotus' histories, annotated by the PROIEL project, found at `http://foni.uio.no:3000/`;

- as a lexicon: the Perseus project's million-word morphological database (found at http://www.perseus.tufts.edu/hopper/opensource/download plus all parses from the training file.

2. For the Stanford Parser (dependency parsing):
   - as a training file: the Greek New Testament and Herodotus' histories, annotated by the PROIEL project, supplemented with the Perseus treebanks found at http://nlp.perseus.tufts.edu/syntax/treebank/.

There are a few obstacles to unifying and formatting these different resources; for instance, the PROIEL project and the Perseus project use different formats; the former has published its data in three different XML formats, while the latter has made its databases available in SQL dump format. The PROIEL project has all its Greek text stored in Unicode, while the entire Perseus framework still relies on beta code, which created a few obstacles along the road. Furthermore, the annotation schemes for both databases are different; PROIEL uses ten-characters morphological abbreviations derived from the Perseus system, with the difference that the Perseus system only use nine characters and orders them slightly differently. Most of the conversion work has been automatised by a series of Python scripts, but here and there some manual input is still required.

Morphological data

A first step is the formatting of the PROIEL morphological database and treebank. This is done with an XSL transformation stylesheet which outputs the data in a two raw text files, one to be used as a training corpus, another to serve as a supplement to the Perseus-based lexicon; a Python script is then run which sorts the lexicon file and removes duplicate forms. It then converts the parses in both files to the format used in the Perseus project's databases, this in order to attain a unified training file with minimum complexity (see: tables 1, 2 and 3).

A second step is the extraction of the Perseus morphological data, which is a bit less straightforward. They have published their data in SQL dump format, that is, raw text file which reflect the structure of the relational database in a programmatic way so that it can be inserted into a new database. I wrote a script which created a separate text file for each important relation: lemmata and their ID's, parses and their ID's, and finally a combined file with every inflected form followed by the ID of its parse and lemma. I then converted all Greek text from beta code to Unicode using the Java beta code conversion utility provided by the EpiDoc consortium (the utility may be found at http://sourceforge.net/projects/epidoc/files/transcoder/). Once everything has been converted to Unicode, a Python script creates a SQLite database from the data for later use; using its relational functionality, we then output a final lexicon file formatted for use with TreeTagger and with all Greek text in Unicode.

The third step is unification and final formatting; we merge both lexica into a single text file, sort the contents and remove all duplicates, after which another script is run which aligns all possible parses on one line, in this way:

```
1 ἀγαπᾶτε        2ppia----i      ἀγαπάω,V-
2 ἀγαπᾶτε        2ppma----i      ἀγαπάω,V-
3 ἀγαπᾶτε        2ppsa----i      ἀγαπάω,V-
```

which is transformed in:

```
1 ἀγαπᾶτε        2ppia----i ἀγαπάω,V-  2ppma----i ἀγαπάω,V-  2
    ppsa----i ἀγαπάω,V-
```

Dependency data

2.3.4 Accuracy testing

field	category	possible values
first field	lemma type	a - adjective c - conjunction d - adverb e - exclamation g - particle l - article m - numerals n - noun p - pronoun r - preposition t - participle v - verb x - miscellanea
second field	person	1 - first person 2 - second person 3 - third person
third field	number	d - dual p - plural s - singular
fourth field	tense	g - gerund p - participle a - aorist f - future i - imperfect l - pluperfect p - present r - perfect t - future perfect
fifth field	mood	i - indicative m - imperative n - infinitive o - optative s - subjunctive
sixth field	diathesis	a - active e - energetic m - medial p - passive
seventh field	gender	f - feminine m - masculine n - neuter o, p and q - unclear
eighth field	case	a - accusative d - dative g - genitive n - nominative v - vocative
ninth field	degree of comparison	c - comparative s - superlative

Table 1: The Perseus ennealiteral morphological abbreviation system.

field	category	possible values
first field	person	1 - first person 2 - second person 3 - third person
second field	number	d - dual p - plural s - singular
third field	tense	a - aorist f - future i - imperfect l - pluperfect p - present r - perfect t - future perfect
fourth field	mood	i - indicative m - imperative n - infinitive o - optative s - subjunctive
fifth field	diathesis	a - active e - energetic m - medial p - passive
sixth field	gender	f - feminine m - masculine n - neuter
seventh field	case	a - accusative d - dative g - genitive n - nominative v - vocative
eighth field	degree of comparison	c - comparative s - superlative
ninth field	placeholder column	-
tenth field	inflectibility	i - inflected n - not inflected

Table 2: The PROIEL decaliteral morphological abbreviation system.

field	value	Perseus first field equivalent
A-	adjective	→ a
C-	paratactic conjunctions	→ c
Df	adverbs	→ d
Dq	adverbial response particles (where, how, etc.)	→ g
Du	adverbial question particles (where, how, etc.)	→ g
F-	Hebrew loan words	→ x
G-	hypotactic conjunctions	→ c
I-	illocutive particles	→ g
Ma	cardinal numerals	→ m
Mo	ordinal numerals	→ m
Nb	nouns (in general)	→ n
Ne	nouns (proper names)	→ n
Pc	pronouns (reciprocative)	→ p
Pd	pronouns (demonstrative)	→ p
Pi	pronouns (interrogative)	→ p
Pk	pronouns (reflexive)	→ p
Pp	pronouns (personal)	→ p
Pr	pronouns (relative)	→ p
Ps	pronouns (possessive)	→ p
Px	pronouns (quantitative, i.e. some, all, none, same, other)	→ x
R-	prepositions	→ r
S-	article	→ l
V-	verb	→ v

Table 3: The PROIEL biliteral lemmatic abbreviation system.

3 THE TOOL

CONTENTS

3.1 GOALS

1. The tool must be able to read EpiDoc XML and a subset of additional tags added for linguistic annotation. It should be, essentially, a text-based replica of the Papyrological Navigator as found at `papyri.info`.

2. search

3. statistics

4. syntactic trees

3.2 DESIGN

The tool works from the command line; while this type of interface has the obvious disadvantage of being a type interface most people cannot work with right off the bat, command line programs generally have an edge over programs with a graphical user interface in some aspects. They are smaller, faster, more efficient and flexible, and in Unix environments can easily be integrated in a larger workflow; it is possible, for instance, to take the output of a program and pipe it immediately into a text file or into another program for processing, or to write a script which automates the use of the program.

3.3 TECHNICAL

The tool is written in Python 3.

lxml for XML display? Whoosh for indexing and search numpy for statistics CairoPlot for SVG syntactic trees

3.4 INTERFACE

The basic command for launching the script is `python tjufy` from within a terminal or simply

```
tjufy
```

from within the Python interpreter (from here on I will use this shorter notation). Running it without options like this will display a help dialog; the same goes for running

```
python tjufy -h
```

or

```
python tjufy --help
```

4 APPLICATIONS

This chapter is meant as a brief evocation of the potential of a fully annotated corpus of the papyri for further research.

4.1 COLLABORATIVE EDITING

The IDP project is heading into crowdsourcing territory at full steam, and excluding our own work from this movement would be inserting a shrill note into this symphony. All data is placed on GitHub at https://github.com/sinopeus/tjufy, freely accessible and editable for all.

This opens promising avenues of inquiry that do not have a direct relation to this thesis. For instance, it can in the future be integrated into SoSOL and absorbed into the larger codebase for the IDP project if it does not create too much overhead for the current developers (the technical back end of the IDP seems to be labyrinthine and adding new layers of complexity might be off-putting). It could even grow into a separate project which itself could be linked to papyri.info as the HGV, APIS and Trismegistos currently are, by a common system of indexation.

Making the code publically available to all also has the advantage of the public eye inspecting the texts; using solely automatic analysis is bound to deliver an inaccurate result, however small that inaccuracy may be, as creating a NLP engine perfectly capable of understanding language would be the equivalent of creating a perfect artificial intelligence. Therefore, considering the size of the corpus, one must rely upon the intelligence of the community. In the same way open source software is often among the best of software due to public inspection, the potential for a crowdsourced corpus is immense.

4.2 CORPUS-BASED GRAMMARS AND LEXICA

Expanding our method to other texts might bring the benefit of comprehensive corpus-based grammars and lexica, which can integrate available data on the fly and create a self-updating and reliable web of grammatical knowledge. Instead of focusing mainly upon a few choice authors or laboriously trudging through the huge wealth of ancient Greek literature to linearly create lexica and grammars, all of it could be harnessed at once in a quantitatively precise and easily visualisable way.

http://www.digitalhumanities.org/dhq/vol/003/1/000033/000033.html

4.3 HISTORICAL AND VARIATIONAL LINGUISTICS

The language of the papyri has an important role to play in the historical linguistics of Greek; once a full annotation has been achieved, it could be possible to implement the same methods used for synchronic language processing to map language changes in a statistical way; it could be possible to estimate the transition probabilities for diachronic grammatical evolutions, which has the potential to create a picture of the evolution of Greek that would be both comprehensive and precise. It even has potential on a comparative level; given the long history and meandering evolutionary trajectory of the Greek language, one could observe from the data catalysts for language evolution in one direction or the other and apply that comparatively.

One might also win valuable insight into language diversity in Egypt; using the paraliterary data already available from the Trismegistos, linguistic phenomena and evolutions could be visualised on a map and give insight into the diatopic, diastratic and diaphasic variation of Egyptian koinê, much in the way of modern dialect survey maps but directly linked to the original texts.

4.4 TEXTUAL CRITICISM

Textual criticism, too, could benefit from improved access to linguistic data; dubious *passus* could be disambiguated by comparing them to similar instances in papyri from the same period and adapting constructions and words from them. This technique is harnessed by Mimno and Wallach [2009], who use the techniques of statistical NLP solely for these specific critical problems. Though textual criticism will for the foreseeable future still necessitate trained papyrologists, the need for a very in-depth knowledge of the corpus of papyri can be greatly reduced by calling upon data from other parts of the corpus to present a series of statistically possible solutions for textual issues.

4.5 NAMED ENTITY RECOGNITION

Named entity recognition is a subdiscipline in natural language processing which is concerned with the automatic extraction and localisation of all kinds of names from texts. It has been used extensively in literary texts with a view to discern the importance of certain characters throughout the text. The KU Leuven's long-standing Prosopographia Ptolemaica project, which aims to be a repository of all inhabitants of

Egypt between 300 and 30 B.C., could easily benefit from these techniques. The abundant manual labour that has gone into the project could be fed as training data to and then supplemented by a named entity recognition engine that could also categorise personal names by any criteria and establish contextual relations between them. To take a very rudimentary example, the name 'Alexander' could be retrieved in all texts and a cluster of related names generated, so that related individuals may be placed in a web of relations; or one could ask, by combining the already present linguistic annotation, to display all adjectives which accompany the name 'Alexander'.

It could even go further than this and also include other particular names, such as places, distances, monetary units, weights, and so on. Historians could create a comprehensive overview of, for instance, the inflation of Egyptian currency, or map out trade connections using a search for all mentions of currency, weight and places which are in proximity to each other.

5 CONCLUSION

BIBLIOGRAPHY

Bod, Rens, Jennifer Hay and Stefanie Jannedy

2004 *Probabilistic Linguistics* (eds.), Cambridge, MA and London: MIT Press. (Cited on pp. 13, 17.)

Dik, Helma and Richard Whaling

2008 "Bootstrapping Classical Greek Morphology", in *Digital Humanities*. (Cited on p. 7.)

2009 "Implementing Greek Morphology", in *Digital Humanities*. (Cited on p. 7.)

Evans, Trevor V. and Dirk D. Obbink

2010 *The Language of the Papyri* (eds.), Oxford University Press. (Cited on pp. v, 1, 3.)

Gignac, Francis Thomas

1976 *A Grammar of the Greek Papyri of the Roman and Byzantine Periods. I. Phonology.* Milano: Goliardica. (Cited on p. 1.)

1981 *A Grammar of the Greek Papyri of the Roman and Byzantine Periods. II. Morphology.* Milano: Goliardica. (Cited on p. 1.)

Haug, Dag Trygve Truslew *et al.*

PROIEL: Pragmatic Resources in Old Indo-European Languages, http://foni.uio.no:3000/, info at http://www.hf.uio.no/ifikk/proiel/. (Cited on p. 6.)

Hopcroft, John E., Rajeev Motwani and Jeffrey D. Ullman

2001 *Introduction to automata theory, languages, and computation*, 2nd ed., Boston: Addison-Wesley. (Cited on pp. 13, 14.)

Kapsomenos, Stylianos G.

1938 *Voruntersuchungen zu einer Grammatik der Papyri der nachchristlichen Zeit*, Münchener Beiträge zur Papyrusforschung und antiken Rechtsgeschichte, München: C. H. Beck. (Cited on p. 2.)

1957 "Ἔρευναι εἰς τὴν γλῶσσαν τῶν ἑλληνικῶν παπύρων. Σείρα Πρώτη", *EEThess*, vii, pp. 225–372. (Cited on p. 2.)

Koshy, Thomas

2004 *Discrete Mathematics with Applications*, Amsterdam, Boston: Elsevier Academic Press. (Cited on pp. 13, 14.)

Ljungvik, Herman

1932 *Beiträge zur Syntax der spätgriechischen Volkssprache.* Vol. 27, Skrifter utgivna av K. Humanistiska Vetenskaps-Samfundet i Uppsala, 3, Uppsala & Leipzig: Humanistiska Vetenskaps-Samfundet. (Cited on p. 2.)

Mandilaras, Basileios G.

1973 *The verb in the Greek non-literary papyri*, Athens: Hellenic Ministry of Culture and Sciences. (Cited on pp. 1, 2.)

Manning, Christopher D. and Hinrich Schütze

1999 *Foundations of Statistical Natural Language Processing*, Cambridge, MA and London: MIT Press. (Cited on p. 13.)

McCarthy, Michael and Anne O'Keeffe

2010 "What are corpora and how have they evolved?", in *The Routledge Handbook of Corpus Linguistics*, ed. by Anne O'Keeffe and Michael McCarthy, London and New York: Routledge. (Cited on pp. 3, 4.)

Mimno, David and Hanna Wallach

2009 "Computational Papyrology (presentation)", in *Media in Transition 6: Stone and Papyrus, Storage and Transmission*, Cambridge, MA. (Cited on pp. 7, 28.)

Natural Language Toolkit

2012 *Natural Language Toolkit*, http://www.nltk.org/. (Cited on p. 11.)

Palmer, Leonard Robert

1934 "Prolegomena to a Grammar of the post-Ptolemaic Papyri", *J. Th. S.*, xxxv, pp. 170–5. (Cited on p. 2.)

1945 *A Grammar of the post-Ptolemaic Papyri: Accidence and Word-Formation*, London: Oxford University Press. (Cited on p. 2.)

Porter, S. E. and M. B. O'Donnell

2010 "Building and Examining Linguistic Phenomena in a Corpus of Representative Papyri", in *The Language of the Papyri*, Oxford University Press, pp. 287–311. (Cited on p. 3.)

Salonius, A. H.

1927 *Zur Sprache der griechischen Papyrusbriefe*, vol. i, Die Quellen, Akademische Buchhandlung. (Cited on p. 2.)

Schmid, Helmut

1994 "Probabilistic Part-of-Speech Tagging Using Decision Trees", in *Proceedings of International Conference on New Methods in Language Processing*. (Cited on p. 13.)

1995 "Improvements in Part-of-Speech Tagging with an Application to German", in *Proceedings of the ACL SIGDAT-Workshop*. (Cited on p. 13.)

Stanford NLP Group

2011 *Statistical natural language processing and corpus-based computational linguistics: An annotated list of resources*, http://www-nlp.stanford.edu/links/statnlp.html. (Cited on p. 11.)

Turing, Alan Mathison

1950 "Computing Machinery and Intelligence", *Mind*, 59, 236, pp. 433–460. (Cited on p. 6.)