Generating and Parsing Classical Greek

GREGORY CRANE
Harvard University, USA

Abstract

This paper will describe the development of a rule based system for the analysis of Greek morphology and provide background for a Greek morphological database created by this system. While this paper focuses on classical Greek, a highly inflected Indo-European language, our work was intended from the outset to provide a case study that would illustrate the problems of morphological analysis and the possibilities that a working morphological system might open up for research Much of the impetus for our work on Greek morphological sprang from work in Akkadian and Sumerian, and from the frustrations that scholars inevitably face when they attempt to work with primary material beyond their immediate area of specialization A system for morphological analysis is both a tool for those specializing in the target language and a gateway whereby those from other disciplines may enter more deeply the primary source texts of ancient Greece

Working with a database of 40,000 stems, 13,000 inflections, and 2,500 irregular forms, *Morpheus*, the parser described in this paper has (as of October 1990) been used to analyze roughly 3,000,000 words, including texts in a variety of dialects from the eighth century BC to the second century AD and has been used to create a morphological database. This morphological database plays a major role in binding together 40 megabytes of primary sources within the hypermedia database on ancient Greece developed by the Perseus Project.[1] At the same time, the database exists as a separate entity distinct from the larger Perseus database, and we plan in early 1992 to distribute for linguistic analysis a fuller version of the database, covering those Greek texts in the *Thesaurus Linguae Graecae* from Homer to the fourth century

This paper deals with the reasons why this parser was developed and the problems that we encountered. A parser for Greek morphology was not new even in 1984 when we began, but our parser was designed to control phenomena (such as Greek diacritics and dialects) that earlier work had not addressed. This paper will describe the initial purposes and rationale for devising our parser, and the problems that we encountered in striving for a higher level of precision. Its purpose is to predict problems and highlight possible solutions for those working in Greek and to serve as an example of one approach for those working in other languages.

Why Build a Parser?

Problems of information retrieval stimulated our initial interest in Greek morphology. In the summer of 1982, the *Thesaurus Linguae Graecae* (*TLG*) released its first batch of electronic texts, 80 megabytes of material distributed on magnetic tape, and we at Harvard developed over the following two years a full text retrieval system that ran under Unix. Loosely based on the *Refer* bibliographic system designed by M. E. Lesk, our system (HCCP) could cope with peculiarities of Greek (e.g. diacritics that floated back and forth or assumed different forms in the word, and physical display and printing of accented Greek) and of scholarly texts (e.g. complex and inconsistent systems for 'chapter and verse'). By 1984 we had developed a system built around a reasonably compact set of inverted indices which allowed scholars to search for isolated words, for words/phrases within an arbitrary number of lines or words of one another, and for particular passages (e.g. open a window that displays texts such as 'Thucydides 1.34.2.' or 'Plato *Symposium* 215A'). While development on this retrieval system continued for another two years, we had brought our work to a basic level of functionality rivalled but not properly exceeded by the microcomputer-based systems (such as the *Ibycus SC* and the *Pandora* program written for the Apple Macintosh) for the TLG developed subsequently and now in common use.

HCCP, like both the old mini-computer based and new microcomputer *Ibycus* computer and like the *Pandora* program, allowed scholars to select strings. Thus, the scholar interested in δημοκρατία ('democracy') would ask for δημοκρατι- and the computer would locate δημοκρατία, δημοκρατίας, and δημοκρατίαν. A scholar interested in δῆμος (the word for 'people') would ask for all words beginning with the letters δημ- and would thus only locate relevant forms (e.g. δῆμος, δήμου, δήμῳ) but substantial noise as well (e.g. every form of δημοκρατία). The shorter a word, the noisier searches became, and a small word that overlaps in form with a common word is very difficult to locate: thus, ἄγος, a very interesting noun that describes 'any matter of religious awe', from pollution to sacrifice, has the same stem (αγ-) as the very common verb ἄγω ('to lead'). A scholar must wade through thousands of verbal forms to locate several dozen occurrences of the noun. In some cases, words are so irregular that they defy string searches. Verbs can have many different stems that bear no resemblance to one another (e.g. οἴσω and ἤνεγκον are both forms of φέρω) or they may effectively have no stems at all (the verb 'to go', for example: εἶμι, 'I go', versus ἴμεν, 'we go').

A system that understood Greek morphology would not simply be more convenient but fundamentally more precise, for it would allow scholars to ask many questions more efficiently and some questions that would simply have been impossible. Morphology was particularly important in Greek (as opposed to Western European Languages): a single verb could have roughly 1,000 forms, and, if we consider that any verb may be preceded by up to three distinct prefixes, the number of forms explodes to roughly 5,000,000. We thus began in the early part of 1984 to build morphologically intelligent retrieval tools. Since we initially imagined that these tools would be part of a larger retrieval engine, we

Correspondence: Gregory Crane, Associate Professor of Classics, Harvard University, Cambridge MA 02138, USA

© Oxford University Press 1991

Downloaded from http://llc.oxfordjournals.org/ at University of Alberta on April 26, 2015

worked primarily in the C programming language and did not at the time feel that we could build on a more general purpose system such as *Kimmo*.[2]

Initial work focused on an algorithm that we called the 'big-bang'. We developed a system that could generate the canonical forms of a Greek word and then used a finite state automaton (in fact, a slightly modified version of the Unix utility *fgrep*) to locate the thousand or so forms generated. This prototype system allowed us to locate all the forms of a particular verb in the Attic dialect that predominates in Greek prose and in much Greek poetry. We could ask for φέρω ('to carry') or for a particular combination of pre-verb and verb (e.g. **ἀπο**-φέρω, 'carry away') and locate both obvious and non-obvious forms (e.g. οἴσομεν, ἀφῆνεγκον as well as φέρετε). We could also place restrictions on the search: by generating only 'optative' or 'subjunctive' or 'first person singular' forms we could identify only those forms which could fit into the category that interested us. The big-bang might only locate 90% of the relevant forms of φέρω, but scholars would need to perform as many as a dozen separate string searches and plough through a high percentage of false hits to find as much material. The 'big-bang' was a great improvement and provided scholars with a potential new tool that could supplement their traditional string searches.

Although we chose to extend and build on the 'big-bang' system, rather than to perfect it in its initial form, we should emphasize that a system of this kind is of great potential value. Herbert Weir Smyth's grammar of Classical Greek contains forty pages with the morphological information for the 600 or so most important Greek verbs. The stems and accompanying morphological information from this source could have served as a basic database to drive a 'big-bang' retrieval system. Even though Smyth's summary for a particular verb will normally not include every single irregular form or stem in the TLG, it contains a substantial amount of information. A scholar who could locate those forms of φέρω implied by the Smyth article would be in a far better position than if he/she were searching for mere strings. Anyone contemplating a morphologically intelligent retrieval system for an ancient language should seriously consider the 'big-bang' approach not as a final solution but as an excellent tool that may not be supplanted for many years.

In late 1984, however, we decided to build on the generative work that we did in the 'big-bang' and augmented the parsing algorithm implemented by David Packard in the 1970s. The parser first looks for possible endings and checks to see whether a stem exists that can be attached to that ending. In the case of πέμπετε ('you (pl.) send'), since both -τε and -ε are valid endings, the parser will see whether πεμπετ- or πεμπε- are stems that could be combined with these endings before determining that the word is in fact a combination of the stem πεμπ- and the inflection -ετε. The routine that examines the stem will also check for prefixes of various kind and can recognize temporal augments (e.g. it would segment ἐ-πέμπ-ετε) and preverbs (e.g. **προσ**-ε-πέμπ-ετε).

Like Packard, we built a system that initially ignored diacritics. Thus, varying forms of πιστεύω, 'to trust' such as πιστεῦσαι (an infinitive), πιστεύσαι (an optative), and πίστευσαι (an imperative) we all converted to πιστευσαι. We added, however, a second pass in which we applied our ability to generate accented forms. Whereas *Morph*, Packard's system, could not distinguish between the three forms above, our system would generate all possible combinations of the stem πιστευσ- and -αι known to it. It would then compare the forms that it generated with the original string that it received from the input. The larger the database of stems and inflections, the more forms become possible and the more false matches occur when diacritics are ignored. The use of diacritics allows *Morpheus* to reject approximately 23% of all possible matches, and thus increases the parser's precision by almost a quarter.

A test parser utilizing this algorithm was built in *Franz Lisp* under Unix at Harvard in early 1985. A fuller implementation was created with support from both Harvard University and UCLA during the summers of 1985 and 1986 by Neel Smith and Joshua Kosman, then of the University of California at Berkeley. This system worked effectively for a single dialect (Attic, the dialect of Athens), but its vocabulary was limited and the model as implemented did not account for dialectical variation. There was no way for the parser to recognize that a first declension noun such as ἀδελφή was proper Attic Greek, but that χώρη, though acceptable in the Ionic dialect, was in Attic supplanted by χώρα (in Attic, a long alpha takes the place of an eta after ε, ι, or ρ). We wanted a parser that could recognize that χώρη was well-formed when it appeared in an Ionic author (such as Herodotus) but that would in other contexts reject this form and, in effect, serve as a spelling checker whether for the student composing exercises in Attic Greek or for texts that had been entered on-line and needed to be verified.

Generating and Parsing Classical Greek

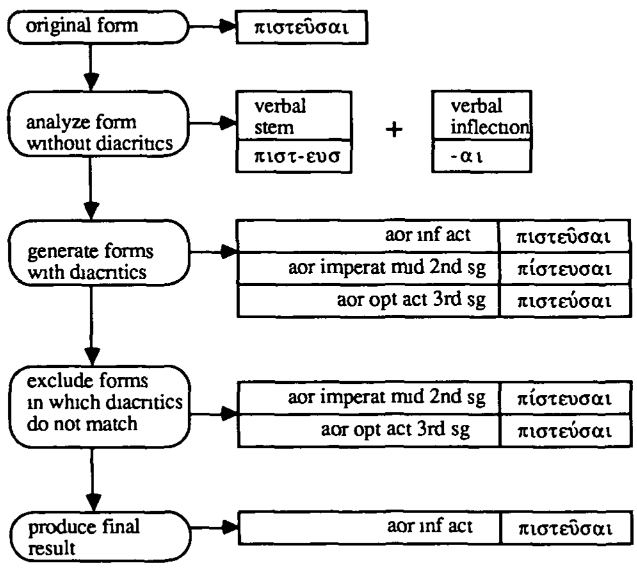

Fig. 1 Simplified view of how *Morpheus* parses a Greek verb *Morpheus* first segments the form into stem and inflection, then generates the various forms that this combination of stem and inflection could take, and compares the generated forms with the original string

Downloaded from http://llc.oxfordjournals.org/ at University of Alberta on April 26, 2015

We anticipated that representing dialectial information would raise issues and problems which we had previously overlooked, and we viewed our work as in large measure an empirical experiment designed to ferret out such problems. We were afraid that the added complexity would make the parsing process exponentially more complex and render our algorithm computationally impractical, but, while complexity did increase, the algorithm held together—indeed, the fact that we could use the same fairly narrow tool to cover several dialects may practically gauge the extent to which these are in fact 'dialects' rather than 'languages'.

Nevertheless, aspects of the system that we had taken for granted became problematic and themselves evolved into sub-projects. We had, for example, begun including our inflections as simple tables that we typed into the system. Two thousand or so inflections cover the Attic dialect fairly well and once these were entered into a database, we were finished. When we began to move beyond Attic prose, however, retrofitting this database became problematic. Different dialects handle phonological changes differently: in Attic, the combination -εο-, for example, contracts to -ου- but in Ionic it can remain as -εο- or become -ευ-, while we find -ιο- in Laconian. We had to locate all the places where an existing -ου- had derived from an -εο- and add the new possibilities, and this was a messy task since not every -ου- in the Attic tables derived from -εο-. Each new dialect might add a number of such philological variants and updating them all by hand invited errors. Likewise, when we added a single ending to a common paradigm, we found that we might need to update more than 130 tables, because the same basic paradigms show up again and again in nominal, adjectival, and participial forms. Furthermore, we could not simply insert the new endings since in some cases these might involve contractions of the -εο- → -ου- kind. Thus, phonological variations and the appearance of new inflections interacted with each other, and each problem rendered the other messier and more complex.

Ultimately, we had to develop a separate, rule-based system that could generate our database of endings automatically from its own knowledge base. Thus, by adding a new phonological rule or a new ending to a single table, the new rule will propagate itself throughout the entire system. This system was based on our generative work for Greek words as a whole, but required a major and unplanned amount of effort. Without such a

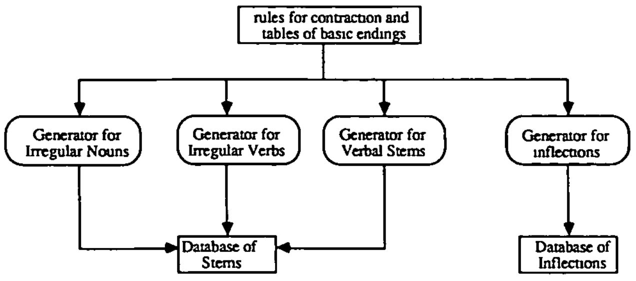
rules for contraction and tables of basic endings
Generator for Irregular Nouns
Generator for Irregular Verbs
Generator for Verbal Stems
Generator for inflections
Database of Stems
Database of Inflections

Fig. 2

system, however, we would simply not be able to handle as many linguistic phenomena. In other words, the more precise the parsing system, the more complex its inner workings and the more 'layered' the system became. An effective parser for one dialect of Greek can be implemented in a relatively short period of time (a few weeks if one follows an established algorithm such as that developed by David Packard), but pushing the process further becomes a more involved problem. Similar sub-projects appeared for generating irregular nominal and verbal forms, and for dealing with semi-regular paradigm classes, and smaller phenomena tied to one dialect or another forced us countless times to revise both our basic data structures and the algorithms used by the parser. In particular, translating the morphological information from a 35,000 word dictionary was an iterative process. New words constantly exhibited peculiarities which forced us to revise our model of Greek morphology. A hierarchical system of generators produce most of the entries in the database for inflections and many of those in the database of nominal and verbal stems. Nevertheless, it should be stressed that the process has not exploded in complexity. We have reached a threshold at which our system, at least when applied to Greek preserved in the relatively consistent literary tradition, has stabilized, where the underlying code normally handles the phenomena that we encounter, and where bugs are relatively infrequent.

Notes

1. Crane, G., Editor-in-Chief. *Perseus 1.0: Interactive Sources and Studies on Ancient Greece* New Haven and London: Yale University Press, 1992. CD-ROM, videodisc and printed documentation.
2. Karttunen, L. (1983). '*Kimmo*: A General Morphological Processor', *Texas Linguistic Forum*, 22: 165–86.

Downloaded from http://llc.oxfordjournals.org/ at University of Alberta on April 26, 2015