Poliqarp
An open source corpus indexer and search engine with syntactic extensions

Daniel Janus
Sentivision Polska Sp. z o.o.
Marynarska 19a, 02-674 Warsaw, Poland
`nathell@korpus.pl`

Adam Przepiórkowski
Insitute of Computer Science
Polish Academy of Sciences
Ordona 21, 01-237 Warsaw, Poland
`adamp@ipipan.waw.pl`

Abstract

This paper presents recent extensions to Poliqarp, an open source tool for indexing and searching morphosyntactically annotated corpora, which turn it into a tool for indexing and searching certain kinds of treebanks, complementary to existing treebank search engines. In particular, the paper discusses the motivation for such a new tool, the extended query syntax of Poliqarp and implementation and efficiency issues.

1 Introduction

The aim of this paper is to present extensions to Poliqarp,[1] an efficient open source indexer and search tool for morphosyntactically annotated XCES-encoded (Ide et al., 2000) corpora, with query syntax based on that of CQP (Christ, 1994), but extending it in interesting ways. Poliqarp has been in constant development since 2003 (Przepiórkowski et al., 2004) and it is currently employed as the search engine of the IPI PAN Corpus of Polish (Przepiórkowski, 2004) and the Lisbon corpus of Portuguese (Barreto et al., 2006), as well as in other projects. Poliqarp has a typical server-client architecture, with various Poliqarp clients developed so far, including GUI clients for a variaty of operating systems (Linux, Windows, MacOS, Solaris) and architectures (big-endian and little-endian), as well as a PHP client. Since March 2006, the 1st stable version of Poliqarp (Janus and Przepiórkowski, 2006) is available under GPL.[2] A version of Poliqarp that implements various statistical extensions is at the beta-testing stage.

Although Poliqarp was designed as a tool for corpora linguistically annotated at word-level only, the extensions described in this paper turn it into an indexing and search tool for certain kinds of treebanks, complementary to existing treebank search engines.

Section 2 briefly introduces the basic query syntax of Poliqarp, section 3 presents extensions of Poliqarp aimed at the processing of treebanks, section 4 discusses implementation and efficiency issues, and section 5 concludes the paper.

2 Query Syntax

In the Poliqarp query language, just as in CQP, regular expressions may be formulated over corpus positions, e.g.: `[pos="adj"]+`, where any non-empty sequence of adjectives is sought, or within values of attributes, e.g.: `[pos="a.*"]`, concerning forms (henceforth: segments) tagged with POSs whose names start with an `a`, e.g., `adj` and `adv`.

Parts of speech and morphosyntactic categories may be queried separately, e.g., the query `[gend=masc]` could be used to search for masculine segments, regardless of the POS or other categories, while the query `[pos="subst|ger"&gend!=masc]` can be used to find nominal and gerundive segments which are not masculine.

A unique feature of Poliqarp is that it may be used for searching corpora containing, in addition to disambiguated interpretations, information about all

[1]**Pol**yinterpretation **I**ndexing **Q**uery **A**nd **R**etrieval **P**rocessor

[2]Cf. `http://poliqarp.sourceforge.net/`.

Proceedings of the ACL 2007 Demo and Poster Sessions, pages 85–88,
Prague, June 2007. ©2007 Association for Computational Linguistics

possible morphosyntactic interpretations given by the morphological analyser. For example, the query `[case~acc]` finds all segments with an accusative interpretation (even if this is not the interpretation selected in a given context), while `[case=acc]` finds segments which were disambiguated to accusative in a given context.

Moreover, Poliqarp does not make the assumption that only one interpretation must be correct for any given segment; some examples of sentences containing an ambiguous segment which cannot be uniquely disambiguated even given unlimited context and all the linguistic and encyclopaedic knowledge are cited in (Przepiórkowski et al., 2004). In such cases, the `=` operator has the existential meaning, i.e., `[case=acc]` finds segments with at least one accusative interpretation marked as correct in the context ("disambiguated"). On the other hand, the operator `==` is universal, i.e., `[case==acc]` finds segments whose all disambiguated interpretations are accusative: segments which were truly uniquely disambiguated to one (accusative) interpretation, or segments which have many interpretations correct in the context, but all of them are accusative.[3] For completeness, the operator `~~` is added, which universally applies to all morphosyntactic interpretations, i.e., `[case~~acc]` finds segments whose all interpretations as given by a morphological analyser (before disambiguation) are accusative.

The most detailed presentation of the original query syntax of Poliqarp is available in (Przepiórkowski, 2004), downloadable from `http://korpus.pl/index.php?page=publications`.

3 Syntactic Extensions

(Przepiórkowski, 2007) argues for the explicit representation of both a syntactic head and a semantic head for each syntactic group identified in a (partially parsed) constituency-based (as opposed to dependency-based) treebank. For example, for the Polish syntactic group *tuzin białych koni*, 'a dozen of white horses', lit. 'dozen-NOM/ACC white-GEN horses-GEN', the syntactic head is *tuzin* 'dozen', while the semantic head is *koni* 'horses'. The segment *koni* is also both the syntactic head and the semantic head of the embedded nominal group *białych koni* 'white horses'. In general, following (Przepiórkowski, 2007), a given segment is a syntactic head of at most one group (e.g., *tuzin* and *koni* in the example above), but it may be a semantic head of a number of groups (e.g., *koni* above is a semantic head of *białych koni* and of *tuzin białych koni*).

This kind of representation is problematic for general search tools for constituency-based treebanks,[4] such as TIGERSearch (Lezius, 2002),[5] which usually assume that the set of edges within a syntactic representation of a sentence is a tree, in particular, that it has a single root node and that each leaf has (at most) one incoming edge.[6] While the former assumption is not a serious problem (an artificial single root may always be added), the latter is fatal for representations alluded to above, as a single segment may be a semantic head of a number of syntactic groups, i.e., it may have several incoming edges.

The extension of Poliqarp presented here makes it possible to index and search for such (partial) syntactic-semantic treebanks. Specifications of syntactic constructions in the extended Poliqarp query language syntax are similar to specifications of particular segments, but they use a different repertoire of attributes, non-overlapping with the attributes used to specify single segments. Two main attributes to be used for querying for syntactic groups are: `type` and `head`. The attribute `type` specifies the general syntactic type of the group, so `[type=Coordination]` will find coordinated constructions, while `[type="[PN]G"]` will find prepositional and nominal groups.

The syntax of values of the attribute `head` differs from that of the other attributes; its values must be enclosed in a double or a single set of square brackets, as in: `[head=[...][...]]` or `[head=[...]]`. In the first case, the first brackets specify the syntactic head and second brackets specify the semantic

[3] In Polish this may happen, for example, in case of some gerund forms which are homographs of true nouns, where meaning does not make it possible to decide on the nominal / gerundive interpretation of the form.

[4] It seems that it would also be problematic for dependency tools such as Netgraph, cf. (Hajič et al., 2006) and `http://quest.ms.mff.cuni.cz/netgraph/doc/netgraph_manual.html`.

[5] Cf. `http://www.ims.uni-stuttgart.de/projekte/TIGER/`.

[6] In TIGER tools, there is a special mechanism for adding a second edge, e.g., in order to represent control.

head, as in the following query which may be used to find elective constructions of the type *najstarszy z koni* '(the) oldest of horses', which are syntactically headed by the adjective and semantically by the semantic head of the dependent of that adjective: `[head=[pos=adj][pos=noun]]`.

In the second case, the content of the single brackets specifies both the syntactic head and the semantic head and, additionally, makes the requirement that they be the same segment. This means that the queries `[head=[case=gen][case=gen]]` and `[head=[case=gen]]` have a slightly different semantics: the first will find syntactic groups where the two heads may be different or the same, but they must be genitive; the second will find groups with the two heads being necessarily the same genitive segment.

The usefulness of such queries may be illustrated with a query for verbs which co-occur with dative dependents denoting students; the first approximation of such a query may look like this: `[pos=verb][head=[case=dat][base=student]]`. This query will find not only dative nominal groups headed by a form of STUDENT, but also dative numeral groups whose main noun is a form of STUDENT, appropriate dative adjectival elective groups, etc.

As syntactic sugar, the constructs `synh=[...]` and `semh=[...]` can be used to enforce a constraint only on, respectively, syntactic or semantic head of a group.

It may seem that, given the possibility to specify the syntactic head of the construction, the attribute `type` is redundant; in fact, we are not currently aware of cases where the specification `type="PG"` or `type="NG"` could not be replaced by an appropriate reference to the grammatical class (part of speech) of the syntactic head. However, the `type` attribute is useful for finding constructions which are not defined by their heads, for example, *oratio recta* constructions, and it is also useful for dealing with coordinate structures.

4 Implementation Issues

To allow for fast searching, the original Poliqarp uses its own compact binary format for corpora, described in detail in (Janus, 2006) and briefly in (Janus and Przepiórkowski, 2006). Because the number of syntactic groups can easily grow very large and be on par with total number of words in a fully-tagged corpus, the representation of syntactic groups should be space-efficient, yet allow for fast decoding and random access.

The key observation to achieving this goal is that, due to the tree nature of the group set, any two groups can be either mutually disjoint or completely contained in each other. Thus, it is possible to serialize the tree into a list, sorted by the lower bound of a group,[7] such that each group is immediately followed by its direct subgroups.

More precisely, the on-disk representation of a treebank is a bit vector that contains the following data for each group: 1) synchronization bit (see below), usually 0; 2) the difference between the lower bound of the previous group and the lower bound of the one in question, encoded in γ-code;[8] 3) γ-encoded length of current group in segments; 4) γ-encoded number of type of this group (the mapping of numbers to type names is stored in a separate on-disk dictionary in which two type numbers are reserved: 0 for coordinated groups and 1 for conjunctions); 5) if this is a coordinated construct (i.e., $type = 0$) — γ-encoded number of subsequent groups (excluding the current one but including indirect subgroups) that are part of the coordination;[9] or 6) if this is not a coordinated construct (i.e., it is an ordinary group) — offset of syntactic and semantic head of this group, in that order, each represented by a binary number of $\log l$ bits, where l stands for the length of the group.

One drawback of this representation is that it does not allow for random access: the γ-code and head offsets have variable length, thus it is not possible to determine which bit one should start with to decode the group sequence for a certain segment. To mitigate this, a synchronization mechanism is employed.

[7] The corpus proper is represented by one large vector of fixed-size structures denoting segments; here, the bounds of a group mean offsets into that vector.

[8] The γ-code is a prefix-free variable-length code that encodes arbitrary integers so that the representation of small numbers takes few bits; see (Witten et al., 1999) for details.

[9] Special treatment of coordination is caused by the fact that, as argued in (Przepiórkowski, 2007), coordinate structures are best treated as multi-headed constructions, with each conjunct bringing its own syntactic and semantic head.

For every k-th segment (k is a constant defined for the corpus, usually 1024), the bit offset of start of the description of the earliest group that intersects this segment is stored as an unsigned little-endian 32-bit integer in a separate file. In the description of this group, the synchronization bit is set to 1, and the lower bound is spelled in full (as an unsigned 32-bit binary integer) so that it is not necessary to know the previous lower bound to start decoding.

This synchronization lines up with the sparse inverted indexing mechanism used by Poliqarp for efficient searching. Poliqarp artificially splits the corpus into fixed-size chunks and remembers which segments occur in which chunks; if the search engine makes random access to the corpus, the accessed segments' offsets are multiplies of the chunk size. It is best, thus, to ascertain that the constant k is also equal to this chunk size.

In a typical scenario with many mostly small groups occurring close to each other, this encoding schema is capable of achieving the ratio of well under two bytes per group and does not incur a significant overhead in corpus size (which is usually in the range of 10–12 bytes times the number of segments for a morphosyntactically but not structurally tagged corpus). This is important, since disk access is the key factor in Poliqarp's performance.

5 Conclusions

In this paper, we presented an extension of Poliqarp, a tool for indexing and searching morphosyntactically annotated corpora, towards the management of syntactically annotated corpora. An interesting feature of thus extended Poliqarp is its ability to deal with treebanks which do not adopt the "at most one incoming edge" assumption and which distinguish between syntactic heads and semantic heads. We also sketched the original and efficient method of indexing such treebanks. The implementation of the extensions currently approaches the alpha stage. By the time of ACL 2007, we expect to release the sources of a relatively stable beta-stage version.

References

Florbela Barreto, António Branco, Eduardo Ferreira, Amália Mendes, Maria Fernanda Nascimento, Filipe Nunes, and João Silva. 2006. Open resources and tools for the shallow processing of Portuguese: The TagShare project. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*.

Oli Christ. 1994. A modular and flexible architecture for an integrated corpus query system. In *COMPLEX'94*, Budapest.

Jan Hajič, Eva Hajičová, Jaroslava Hlaváčová, Václav Klimeš, Jiří Mirovský, Petr Pajas, Jan Štěpánek, Barbara Vidová Hladká, and Zdeněk Žabokrtský, 2006. *PDT 2.0 – Guide*. Charles University, Prague. June 20, 2006.

Nancy Ide, Patrice Bonhomme, and Laurent Romary. 2000. XCES: An XML-based standard for linguistic corpora. In *Proceedings of the Linguistic Resources and Evaluation Conference*, pages 825–830, Athens, Greece.

Daniel Janus and Adam Przepiórkowski. 2006. Poliqarp 1.0: Some technical aspects of a linguistic search engine for large corpora. In Jacek Waliński, Krzysztof Kredens, and Stanisław Goźdź-Roszkowski, editors, *The proceedings of Practical Applications of Linguistic Corpora 2005*, Frankfurt am Main. Peter Lang.

Daniel Janus. 2006. Metody przeszukiwania i obrazowania jego wyników w dużych korpusach tekstów. Master's thesis, Uniwersytet Warszawski, Wydział Matematyki, Informatyki i Mechaniki, Warsaw.

Wolfgang Lezius. 2002. TIGERSearch — ein Suchwerkzeug für Baumbanken. In Stephan Busemann, editor, *Proceedings der 6. Konferenz zur Verarbeitung natürlicher Sprache (KONVENS 2002)*, Saarbrücken.

Adam Przepiórkowski, Zygmunt Krynicki, Łukasz Dębowski, Marcin Woliński, Daniel Janus, and Piotr Bański. 2004. A search tool for corpora with positional tagsets and ambiguities. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004*, pages 1235–1238, Lisbon. ELRA.

Adam Przepiórkowski. 2004. *The IPI PAN Corpus: Preliminary version*. Institute of Computer Science, Polish Academy of Sciences, Warsaw.

Adam Przepiórkowski. 2007. On heads and coordination in valence acquisition. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing (CICLing 2007)*, Lecture Notes in Computer Science, pages 50–61, Berlin. Springer-Verlag.

Ian H. Witten, Alistair Moffat, and Timothy C. Bell. 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 2nd edition.