Development of a Corpus Workbench for the METU Turkish Corpus

Umut Özge, Bilge Say

Middle East Technical University
İnönü Bulvarı, 06531, Ankara, Turkey.
{umut,bsay}@ii.metu.edu.tr

Abstract

We will introduce a corpus workbench designed and implemented for the METU Turkish Corpus. The workbench design introduces a number of useful features and the workbench itself is basically usable with any TEI and XML compliant corpus, provided that it can be indexed in the format required by the workbench.

1. Introduction

In this paper, we give a detailed description of functional and quality aspects of a corpus workbench designed to be used with the METU Turkish Corpus. The METU Turkish Corpus is a 2 million word corpus of post-1990 written Turkish (Say et al., 2002). The corpus consists of 2000-word samples which are annotated in conformance with XCES (XML-based Corpus Encoding Standard) at the typographic-general level (Ide and Priest-Dorman, 2000).[1] A representative selection of samples, named METU-Sabancı Treebank totaling around 10,000 sentences have also been annotated in an XML-conformant manner with morphological and syntactic information (Atalay et al., 2003). The corpus workbench allows graphical browsing of treebank entries in conjunction with corpus queries.

In the next section, we give the main concerns that guided our design. In Section 3., we give a more technical description of the query resolution and viewing mechanisms. We conclude with a summary of the features of the workbench as compared to other workbenches.

2. The Design Rationale of the Workbench

The main purpose of building the workbench was to provide the users with even a minimal computer experience with an easy-to-use and fast tool that would enable them to perform simple search operations over the METU Turkish Corpus. In this respect, our primary design criteria were user-friendliness and speed. Regarding the former, a graphical user interface that enhances query operations, management of results and viewing was developed. As for the latter, we made use of an index mechanism that speeds up the query resolution and retrieval (see Section 3.1.).

Regarding the possible future extensions and modifications to the workbench, flexibility was taken as another major design criteria. To this end, an object oriented approach was pursued in the development, where background query operations and display and dialog components are implemented as separate modules. Again for the sake of flexibility, all the internal and user-saved data are stored as XML files, which provides access to data with any XML processing software.

A certain level of generality was another concern in developing the software. In this respect, indexing and querying systems were designed and implemented in such a way that any XCES annotated corpus is plugable to the workbench. However, the software is not fully internationalized in that a small fraction of the code must be modified in order for the workbench to be properly used with languages with encodings different than Turkish.

Finally considering the variety in platforms used throughout the academic community, Java was chosen as the implementation language. The workbench is tested on Windows XP^{TM} and a few popular Linux systems.

3. System Description

3.1. Query Resolution and Retrieval Mechanism

In order to enhance the speed of query resolution and retrieval, the workbench makes use of an *inverted file index* with a *word level granularity* (Witten et al., 1999). The index associates each type in the corpus with a list of the *occurrences* of its tokens. Occurrences are encoded as triples $\langle D, P, W \rangle$ where:

> D is the disk address of the start of the XCES document the word occurs in.
>
> P is the disk address of the start of the paragraph the word occurs in.
>
> W is the index of the word within the XCES document.

The index consists of two components: a *lexicon file* and an *occurrences file*. In the lexicon file, every type is paired with a disk address pointing to the corresponding entry in the occurrences file which holds the occurrence information of the types listed in the lexicon as lists of triples.

The idea in separating the lexicon and occurrence files is that the lexicon can be loaded to working memory in runtime. Once the lexicon is loaded to a hash map in working memory, terms can easily and efficiently be looked up in this map and a pointer to the occurrence file can be retrieved. This pointer is used to randomly access the occurrences file from which the occurrence information is retrieved. This occurrence information is then used either in resolving complex queries or random accessing the corpus

[1] XCES is based on TEI (Text Encoding Initiative) guidelines (Sperberg-McQueen and Burnard, 1994).

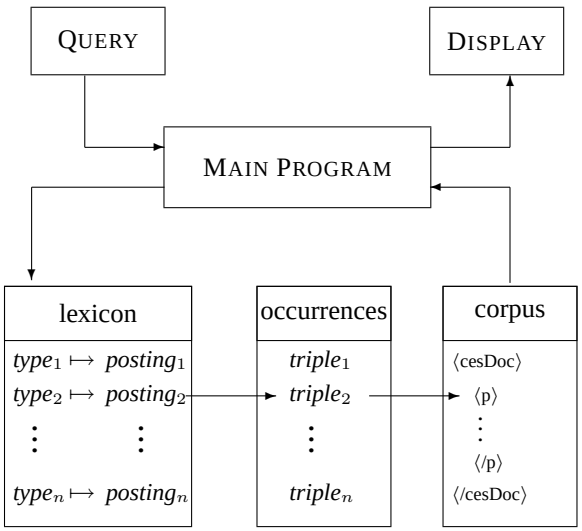

Figure 1: General architecture of the workbench.

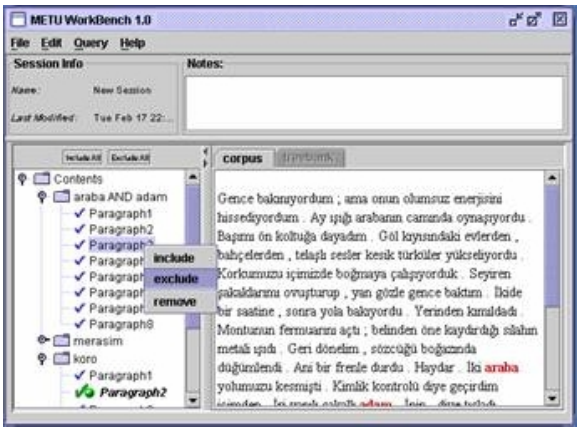

Figure 2: A screenshot of a session.

and retrieving the relevant text and bibliographical information. The system is depicted in Figure 3.1.

The workbench offers two types of queries: *boolean* and *regular expression*. In the *boolean query*, much like in Internet search engines, the user can construct simple or complex queries by using boolean operators "AND" and "OR" and parentheses, or can query successive words by enclosing them in double quotes. In the *regular expression query*, the regular expression entered by the user is searched in the lexicon, and matching types are populated for the user to choose the ones that s/he wants to be retrieved. Both types of queries can be filtered through bibliographic constraints such as author, genre and year. The user is also asked for the unit of retrieval where available options are a paragraph or a 2000-word XCES document. Queries are made through query dialogs enabling the user to specify bibliographic filters and unit of retrieval.

3.2. Viewing and Other Features

In the workbench, user interaction is organized through *sessions*. The session interface has four components (Figure 3.2.). The simplest of these is the label that displays the name and modification information of the session. To the left of this label comes a text component, where the user can take notes about the session. Text from query results can be copied and pasted to this area. Notes are saved with the session, and can be printed as well.

All the queries made within a session and matching results are brought together in a *query tree* (Figure 3.2.), for the user to easily browse through them. With the aid of a context menu, the user can perform the operations of *include*, *exclude* and *remove* on the selected nodes of the query tree. The excluded results or queries are discarded in saving and do not appear in print-outs, whereas removed results or queries are permanently removed from a session.

Upon selection of a node in the query tree, according to whether it is a query or result node, query information or retrieved material is displayed in a tabbed viewing component. This component consists of two tabs. One is a standard HTML viewer used to display the retrieved text where the query terms are highlighted. The other tab is a custom display component that is used to display the syntactic relations and morphological information pertaining to the sentences of a retrieved paragraph (Figure 3.2.). This tab gets active only when a result also has a treebank entry in the corpus. Such treebank-related results are highlighted in the query tree as well.

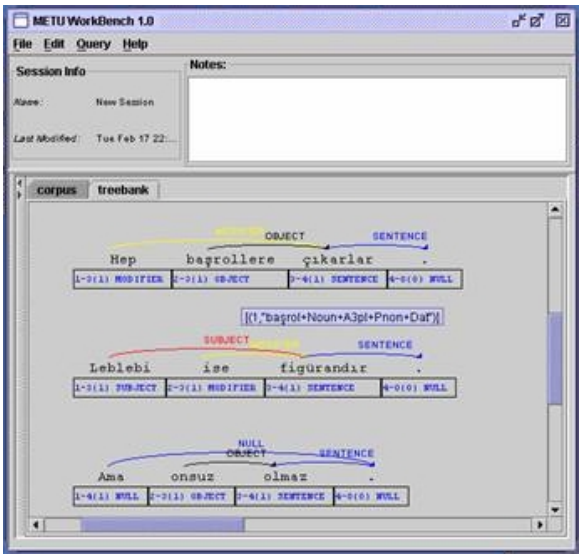

Figure 3: A screenshot of the treebank viewer.

4. Conclusion

The METU Turkish Corpus workbench is similar to currently available corpus workbench and concordancing tools in various respects (SARA (Fresko, 1994), Wordsmith (Scott, 1999), TIGERSearch (König and Lezius, 2001), XKWIC (Lee and Rayson, 2000) etc.) such as standard querying and viewing options. The concept of a session where user-selected queries, user-selected results, and user notes can be treated as a unit for saving and printing is a feature that enhances usability. The indexing mechanism, integratibility of a treebank browser and general platform

independence with Java makes workbench a viable addition to the alternatives available. Further enhancements such as internationalization, integration of a indexing front-end may allow the workbench to develop into a more corpus and language independent tool.

5. Acknowledgments

We thank METU (BAP Project No: 99-06-04-02) and TUBITAK (EEEAG Project No: 199E026) for providing the funding for building the corpus and the treebank respectively, several project members for their suggestions during the development of the workbench and three anonymous reviewers for their comments.

6. References

Atalay, Nart Bedin, Kemal Oflazer, and Bilge Say, 2003. The annotation process in the Turkish treebank. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora - LINC*. Budapest, Hungary.

Fresko, Marc, 1994. SARA (SGML Aware retrieval application). http://www.ukoln.ac.uk/services/papers/bl/rdr6173/sara.html.

Ide, N. and G. Priest-Dorman, 2000. Corpus encoding standard: Document ces 1, version 1.5. http://www.cs.vassar.edu/CES/.

König, E. and W. Lezius, 2001. The TIGER language - a description language for syntax graphs. part 1: User's guidelines. Technical report, IMS, University of Stuttgart.

Lee, David and Paul Rayson, 2000. XKWIC: A powerful concordancer for research. In *Proceedings of TALC 2000*. Graz, Austria.

Say, Bilge, Deniz Zeyrek, Kemal Oflazer, and Umut Özge, 2002. Development of a corpus and a treebank for present-day written Turkish. In *Proceedings of the Eleventh International Conference on Turkish Linguistics*.

Scott, M., 1999. Wordsmith tools version 3.0. Oxford: Oxford University Press. Software Package.

Sperberg-McQueen, C. M. and L. Burnard, 1994. *Guidelines for Electronic Text Ecoding and Interchange (TEI P3)*. Chicago, Oxford: Text Encoding Initiative.

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