Cross-Platform, Cross-Grammar Comparison — Can it be Done?

Ulrich Callmeier and **Stephan Oepen**
Saarland University
Computational Linguistics
{uc | oe}@coli.uni-sb.de

(see 'http://www.coli.uni-sb.de/itsdb/')

Abstract

This software demonstration reviews recent improvements in comparing large-scale unification-based parsing systems, both across different platforms and multiple grammars. Over the past few years significant progress was accomplished in efficient processing with wide-coverage HPSG grammars. A large number of engineering improvements in current systems were achieved through collaboration of multiple research centers and mutual exchange of experience, encoding techniques, algorithms, and pieces of software.

We argue for an approach to grammar and system engineering that makes systematic experimentation and the precise empirical study of system properties a focal point in development. Adapting the profiling metaphor familiar from software engineering to constraint-based grammars and parsers enables developers to maintain an accurate record of system evolution, identify grammar and system deficiencies quickly, and compare to earlier versions, among analytically varied configurations, or between different systems. We demonstrate a suite of integrated software packages facilitating this approach, which are publicly available both separately and together.

The [incr tsdb()] profiling environment (Oepen & Carroll, 2000) integrates empirical assessment and systematic progress evaluation into the development cycle for grammars and processing systems; it enables developers to obtain an accurate snapshot of current system behaviour (a profile) with minimal effort. Profiles can then be analysed and visualized at variable granularity, reflecting various aspects of system competence and performance, and compared to earlier results. Since the [incr tsdb()] package has been integrated with some eight processing platforms by now, it has greatly facilitated cross-fertilization between various research groups and implementations.

PET is a platform for experimentation with processing techniques and the implementation of efficient processors for unification-based grammars (Callmeier, 2000). It synthesizes a range of techniques for efficient processing from earlier systems into a modular C++ implementation, supplying building blocks (such as various unifiers) from which a large number of experimental setups can be configured. A parser built from PET components can be used as a time- and memory-efficient run-time system for grammars developed in the LKB system distributed by CSLI Stanford (Copestake & Flickinger, 2000). In daily grammar development it allows frequent, rapid regression tests.

We emphasize in this demonstration the crucial importance of experimental system comparison, eclectic engineering, and incremental optimization. Only through the careful analysis of a large number of interacting system parameters can one establish reliable points of comparison across different parsers and multiple grammars simultaneously.

References

Callmeier, U. (2000). PET — A platform for experimentation with efficient HPSG processing techniques. *Natural Language Engineering, 6 (1) (Special Issue on Efficient Processing with HPSG)*, 99–108.

Copestake, A., & Flickinger, D. (2000). An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second Linguistic Resources and Evaluation Conference* (pp. 591–600). Athens, Greece.

Oepen, S., & Carroll, J. (2000). Performance profiling for parser engineering. *Natural Language Engineering, 6 (1) (Special Issue on Efficient Processing with HPSG)*, 81–97.