Creating language resources and applications using finite-state morphological grammars

Mans Hulden
The University of Helsinki
Iñaki Alegria
The University of the Basque Country

Overview

This tutorial will be concerned with how to use recent freely available (open source) resources to construct morphological analyzers and generators as well as applications that depend on these. The particular focus will be on building robust morphological processing systems and immediately usable derivative applications with the aid of two open-source finite-state toolkits, *foma* and *HFST*. Relevant applications that can be derived from these include, for instance, lemmatizers, spell checkers and grammar correctors. A well-constructed morphological analyzer for a language is both a valuable stand-alone product that can be used for a number of purposes in applications and research and often an essential component in larger NLP systems.

The purpose of the tutorial is to provide an overview of

- current finite-state machine compilers intended for linguistic modeling, and how to start writing a morphological grammar/analyzer/parser for a language
- the types of problems finite-state methods are particularly suitable for (in addition to morphology and lemmatization)
- practical concerns in finite-state programming (by showing smaller grammars in detail)
- the methods and resources available for launching other applications and creating new resources from finite-state grammars
- where to locate further documentation and grammar examples that may be needed in the course of a morphological project.

While the focus of the tutorial is computational morphological analysis, the techniques presented are useful also for linguistic research, phonology in particular. The same techniques that are used for constructing morphological analyzers can also be put to use in modeling phonological (and morphological) generalizations in a variety of theoretical frameworks, as well as testing and comparing the predictive power of different phonological hypotheses within both rule and constraint-based approaches.

Content & Schedule

The tutorial is divided into two parts. In the first part, we familiarize the participants with finite-state programming and give an overview of the design steps in constructing a morphological analyzer. The second part focuses on the variety of applications that can be derived from existing

morphological analyzers, as well as integration of a morphological component into other language processing projects.

Part 1: Finite-state programming

In this part we introduce the formalisms of *foma* and *HFST*, two freely available toolkits for the construction of unweighted and weighted automata and transducers. We give special attention to current techniques of morphological modeling with finite-state methods. This includes creating a lexicon component (probabilistic and non-probabilistic) using the *lexc* lexicon specification language, and creating transducers that model phonological and morphological alternation.

Part 2: Creating new applications

In this part we give an overview of how morphological analyzers created with the current tools can be used to create new language resources. We include treatment of how to export a morphological analyzer as a spell checking component into OpenOffice, Mozilla (Firefox), and other applications. We also touch upon design concerns and the integration of morphological analyzers and guessers into systems that deal with syntactic processing.

Speakers

Mans Hulden is a researcher in language technology at the University of Helsinki as part of the EU FP7 project CLARA (Common Language Resources and their Applications), and works with parsing technologies and grammar models. He is the author of the open-source finite-state toolkit *foma*, designed for producing large-scale morphological analyzers and generators. As a member of the Helsinki Finite State Research Group he is also involved with the development of the HFST toolkit for finite-state language processing. He received his PhD in Computational Linguistics in 2009 from the University of Arizona.

Iñaki Alegria is a member of the IXA group at the University of the Basque Country and has developed a number of NLP tools for the Basque language. The IXA group consists of 24 members working on a variety of practical and theoretical topics in developing language technology for the Basque language, including computational morphologies, spell checkers, information retrieval applications, lemmatizers, machine translation applications (Spanish-Basque), and corpora construction. He has a PhD in Computer Science (1995) from the University of the Basque country.