JMorpher: A Finite-State Morphological Parser in Java for Android

Leonel F. de Alencar, Mardonio J.C. França, Katiuscia M. Andrade, Philipp B. Costa, Henrique S. Vasconcelos, and Francinaldo P. Madeira

Universidade Federal do Ceará, Group of Computer Networks, Software Engineering, and Systems (GREat)
Campus do Pici, Bloco 942-A, CEP 60455-760 Fortaleza, Brazil
{leonelararipe,mardoniofranca,katiusciaandrade,philippcosta,henriquevasconcelos}@great.ufc.br,
naldomadeira@gmail.com
http://www.great.ufc.br

Abstract. This paper presents JMorpher, a morphological parsing utility that is implemented in pure Java. It is apparently the first tool of this type that natively runs on Android mobile devices. JMorpher compiles a lexical transducer definition in the AT&T raw text format, of the type generated by Foma and other open source finite-state packages, into an internal Java representation which is drawn upon to parse input strings. Besides the API, JMorpher comprises of a simple graphical interface that allows the user to load a transducer file, type in some text and parse it. Results of an evaluation based on large Portuguese lexical transducers of different complexity degrees are provided. The implementation was shown to be very efficient on a desktop PC. Although, on an Android smartphone, JMorpher’s performance is much lower, it is still suited to the needs of NLP tasks in this environment.

Keywords: NLP. Finite-State Morphology. Morphological Analysis. Morphological Parsing. Lexical Transducer. Android Technology.

1 Introduction

Morphological analysis is a key module in the syntactic parsing pipeline, fulfilling the job of mapping word forms occurring in texts to sets of abstract representations, typically consisting of lemma, POS-tag and subcategorial features \[1,9\]. Independent of sentence structure processing, the output of a morphological parser may be useful in a wide range of applications. Two examples from the growing field of mobile computing include the use of lemma information for dictionary lookup in an e-book reader application and the use of morphosyntactic features to improve performance of classification algorithms for organizing the user’s messages.

Over the last two decades or so, finite-state transducers, due to compact storage and fast processing, have been a preferred implementation of morphological
analyzers. Many industrial-scale lexical transducers were compiled for typologically diverse languages, using a variety of finite-state toolkits [1], [8], [12], [15].

Xerox Finite State Tools (XFST) [1] are one of the most efficient implementations of finite-state morphology, from the linguist’s point of view it is the best documented and is a user-friendly toolkit. XFST has the caveat of being proprietary software, although it is freely distributed for non-commercial research purposes. Since 2010, computational morphology could be carried out with an open source, free software alternative to XFST, the Foma finite compiler and C library [8]. In general Foma is as efficient as XFST, Foma also has the advantage of being almost completely compatible with the XFST syntax, including the Lexc formalism.

As far as we know, Portuguese has no freely available large lexical transducer distributed under a free software/open source license. For the development of free, wide-coverage lexical resources for Portuguese using finite-state techniques, Foma is a very attractive option, especially when targeting lesser powered mobile devices with low storage capacity. As shown in Table 1, a lexical transducer based on the DELAF-PB computational lexicon [13], for example, takes up less than one tenth of the original entries in raw text format. However, in the mobile setting, Foma has the disadvantage of restricted portability. On the Android platform, for example, compiling Foma proved to be a difficult task due to library dependencies. It is worth noting that different mobile processors require different binaries, complicating the distribution of language-aware apps.

Summing up, Foma would be very useful in the computational processing of Portuguese on mobile devices, but, it is seemingly impractical in this environment. In this paper, we present JMorpher, a pure Java solution developed in order to solve this problem. JMorpher requires to no external libraries and is natively able to run on Android. JMorpher compiles finite-state transducers in the AT&T raw text format, of the type generated by Foma, HFST [12] etc., into an internal Java representation. This representation is drawn upon by the lookup algorithm, which outputs the parses for the input strings contained in the surface language of the transducer. JMorpher is not a port of Foma’s flookup utility. Instead, JMorpher’s internal transducer representation as well as its parsing algorithms were developed from scratch, aiming at an efficient Java implementation.

The main motivation for developing JMorpher was that HFST’s Java lookup utility [6], [12], the only tool of this type we know about implemented in this language, is not open source. Their implementation has not been maintained since 2009 and was found to be unstable in simple tests we carried out.

The paper is structured as follows. In the next section, we describe the different types of transducers of Portuguese and test sets used to evaluate JMorpher. In Section 3, we outline the general concept of the package as well as its API and graphical interface. Then, in Section 4, the evaluation results are presented. Finally, the last section summarizes the main characteristics of the tool and points out possible directions for further research.