Procesamiento del Lenguaje Natural, Revista nº 44, marzo de 2010, pp 153-154 recibido 08-01-10 revisado 17-02-10 aceptado 06-03-10

Parsing Schemata for Practical Text Analysis

Esquemas de Análisis Sintáctico para el Análisis Práctico de Textos

Carlos Gómez-Rodríguez
Universidade da Coruña
Campus de Elviña, s/n, 15071 A Coruña
cgomezr@udc.es

Resumen: Tesis doctoral en Informática realizada por Carlos Gómez-Rodríguez bajo la dirección de los doctores Miguel A. Alonso Pardo (Univ. da Coruña) y Manuel Vilares Ferro (Univ. de Vigo). La defensa de la tesis tuvo lugar el 5 de junio de 2009 ante el tribunal formado por los doctores John A. Carroll (Univ. of Sussex), Giorgio Satta (Univ. degli Studi di Padova), Víctor Jesús Díaz Madrigal (Univ. de Sevilla), Leo Wanner (Univ. Pompeu Fabra) y Jesús Vilares Ferro (Univ. da Coruña). La calificación obtenida fue de *Sobresaliente Cum Laude* por unanimidad, con mención de Doctor Europeo.
Palabras clave: Esquemas de análisis sintáctico, análisis sintáctico, formalismos gramaticales, análisis de constituyentes, análisis de dependencias

Abstract: PhD thesis in Computer Science written by Carlos Gómez-Rodríguez under the supervision of Miguel A. Alonso Pardo (Univ. da Coruña) and Manuel Vilares Ferro (Univ. de Vigo). The author was examined on June 5, 2009 by the following committee: John A. Carroll (Univ. of Sussex), Giorgio Satta (Univ. degli Studi di Padova), Víctor Jesús Díaz Madrigal (Univ. de Sevilla), Leo Wanner (Univ. Pompeu Fabra) and Jesús Vilares Ferro (Univ. da Coruña). The grade obtained was unanimous *Sobresaliente Cum Laude*, with European Doctorate Mention.
Keywords: Parsing schemata, parsing, grammatical formalisms, constituency parsing, dependency parsing

1 Introduction

This dissertation provides several theoretical and practical tools that extend the applicability of Sikkel's theory of parsing schemata (Sikkel, 1997) in several different directions.

First, a compilation technique is defined that can be used to obtain efficient implementations of parsers automatically from their corresponding schemata. This makes it possible to use parsing schemata to prototype and test parsing algorithms, without the need of manually converting the formal representation to an efficient implementation in a programming language.

Second, the range of parsing algorithms that can be defined by means of schemata is extended with the definition of new variants of the formalism that can deal with error-repair parsers and dependency-based parsers.

Apart from these tools themselves, the dissertation also introduces several research results that have been obtained by using them. The compilation technique is used to obtain implementations of different parsers for context-free grammars (CFG) and tree-adjoining grammars (TAG) and perform an empirical analysis of their behaviour with real-sized grammars. The extension of parsing schemata for error-repair parsing is used to define a transformation that can automatically add error-repair capabilities to parsers that do not have them. Finally, the extension of parsing schemata for dependency parsing is used to find formal relationships between several well-known dependency parsers, as well as to define novel algorithms for mildly non-projective dependency structures.

An extended version of the thesis is scheduled for publication by Imperial College Press in Fall 2010 (Gómez-Rodríguez, 2010).

2 Overview

The thesis is structured in five parts. The first part is introductory, containing a first chapter which summarises the main goals and contributions of the thesis, and a second

ISSN 1135-5948

chapter defining the formalism of parsing schemata, that will be used throughout the thesis. The second part presents a compiler for parsing schemata and several empirical studies of constituency-based parsers conducted with it. The third part introduces an extension of parsing schemata that can be used to describe error-repair parsers. The fourth part is devoted to a variant of schemata for dependency-based parsers. Finally, the fifth part summarises conclusions and discusses future work.

A chapter-by-chapter breakdown of the three central parts follows.

2.1 Part II: Compiling Parsing Schemata

Chapter 3 presents a compiler able to automatically transform parsing schemata into efficient Java implementations of their corresponding algorithms. The system analyzes the deduction steps in the input schema in order to determine the best data structures and indexes to use, ensuring that the generated implementations are efficient.

In Chapter 4, the compiler presented in Chapter 3 is used to generate implementations of three well-known CFG and TAG parsing algorithms and compare their empirical performance on several grammars taken from natural language corpora. In particular, TAG parsers are compared with the XTAG grammar, a real-life, wide-coverage feature-based TAG. Note that previous comparative studies of TAG parsers in the literature were done with "toy" grammars, but there were no such studies with wide-coverage grammars.

In Chapter 5, the parsing schema compiler is used to conduct further empirical studies of CFG and TAG parsers. Artificially generated grammars are used to measure the parsers' empirical computational complexity with respect to string and grammar size, as well as the overhead caused by using TAG to parse context-free languages.

2.2 Part III: Parsing Schemata for Error-Repair Parsers

Chapter 6 introduces error-repair parsing schemata: an extension of parsing schemata which can be used to define parsers that can robustly handle sentences with errors or inconsistencies. We show how this formalism can be used to describe existing algorithms and prove their correctness. The framework is then used to develop a generic technique for obtaining efficient robust parsers based on regional error repair, and empirical performance results are provided.

In Chapter 7, the framework of error-repair parsing schemata is used to define a general transformation technique to automatically obtain robust, error-repair parsers from standard non-robust parsers. The technique can be applied to a wide range of parsers, and the schemata thus obtained can be implemented with global, regional or local error-repair techniques, using the compiler of Chapter 3.

2.3 Part IV: Parsing Schemata for Dependency Parsers

In Chapter 8 a variant of parsing schemata is defined that can be used to describe, analyse and compare dependency parsing algorithms. This extension is used to establish clear relations between several existing projective dependency parsers and prove their correctness. Parsing schemata for non-projective dependency parsers are also presented. A variant of the formalism to represent parsers based on link grammar is also shown, including examples of how some existing dependency parsers can be adapted to produce parsers for link grammar.

In Chapter 9, parsing schemata are used to solve the problem of efficiently parsing mildly non-projective dependency structures. Polynomial time parsing algorithms are presented for various mildly non-projective dependency formalisms, including well-nested structures with gap degree bounded by a constant k, and a new class of *mildly ill-nested* structures for gap degree k which includes all the sentences present in a number of dependency treebanks.

References

Gómez-Rodríguez, Carlos. 2010. *Parsing Schemata for Practical Text Analysis.* Mathematics, Computing, Language, and Life: Frontiers in Mathematical Linguistics and Language Theory. Imperial College Press, London, UK.

Sikkel, Klaas. 1997. *Parsing Schemata — A Framework for Specification and Analysis of Parsing Algorithms.* Texts in Theoretical Computer Science — An EATCS Series. Springer-Verlag, Berlin-Heidelberg-New York.