Open CCG Workbench and Visualization Tool

Thepchai Supnithi, Suchinder Singh, Taneth Ruangrajitpakorn,

Prachya Boonkwan, Monthika Boriboon

Human Language Technology
National Electronics and Computer Technology Center
112 Thailand Science Park, Phahonyothin Road, Klong 1,
Klong Luang Pathumthani, 12120, Thailand
+66-2-564-6900 Ext.2236, Fax.: +66-2-564-6772
{ thepchai.supnithi, taneth.ruangrajitpakorn, prachya.boonkwan,
monthika.boriboon }@nectec.or.th, suchindersingh@hotmail.com

Abstract

Combinatorial Category Grammar is (CCG) a lexicalized grammar formalism which is expressed by *syntactic category*, a logical form representation. There are difficulties in representing CCG without any visualization tools. This paper presents a design framework of OpenCCG workbench and visualization tool which enables linguists to develop CCG based lexicons more easily. Our research is aimed to resolve these gaps by developing a user-friendly tool. OpenCCG Workbench, an open source web-based environment, was developed to enable multiple users to visually create and update grammars for using with the OpenCCG library. It was designed to streamline and speed-up the lexicon building process, and to free the linguists from writing XML files which is both cumbersome and error-prone. The system consists of three sub-systems: grammar management system, grammar validator system, and concordance retrieval system. In this paper we will mainly discuss the most important parts, grammar management and validation systems, which are directly related to a CCG lexicon construction. We support users in three levels; Expert linguists who play a role as lexical entry designer, normal linguists who adds or edits lexicons, and guests who requires an acquisition to the lexicon into their applications.

1. Introduction

OpenCCG [OpenCCG], a CCG grammar development toolkit, causes difficulties in speeding up the grammar development process; namely, grammar rule isolation, multiple file referencing, macro-inheritance non-preview ability, and non-visual representation. These difficulties are originated because of the gap between the flexibility of linguists' comprehension and the concreteness of the tool. This paper describes an online lexical resource workbench to facilitate grammar development in OpenCCG.

Our research is aimed to resolve these gaps by developing a user-friendly tool. OpenCCG Workbench, an open source web-based environment, was developed to enable multiple users to visually create and update grammars for using with the OpenCCG library. It was designed to streamline and speed-up the lexicon building process, and to free the linguists from writing XML files which is both cumbersome and error-prone. The system consists of three sub-systems: grammar management system, grammar validator system, and concordance retrieval system. In this paper we will mainly discuss the most important parts, grammar management and validation systems, which are directly related to a CCG lexicon construction.

The rest of this paper is structured as follows. Section 2 explains CCG concept. Section 3 shows the main idea of our CCG workbench and visualization tool. Section 4 illustrates the implementation of CCG workbench and visualization tool. Finally, Section 5 concludes the paper and lists up future work.

2. Combinatorial Category Grammar

Combinatory Categorial Grammar (CCG) [Steedman2000] is a lexicalized grammar formalism originated from Categorial Grammar (CG) [Ajdukiewicz1935, Bar-Hillel1953]. In CG, all grammatical expressions are distinguished by a *syntactic category* identifying them as either a function from arguments of one type to results another (aka. *function*), or as an argument (aka. *primitive category*) [Steedman2000]. CCG is unique among other categorial grammars in its treatment of constituency, long-distant dependency, and binding. According to CCG concept, the core CG was extended with functional operations on adjacent categories — such as functional composition and type-raising operation [Baldridge2003]. CCG is semantic-transparent, since semantics can be compositionally interpreted during derivation.

Combinatory Categorial Grammar (CCG) [Steedman2000] is a lexicalized grammar formalism originated from Categorial Grammar (CG) [Ajdukiewicz1935, Bar-Hillel1953]. In CG, all grammatical expressions are distinguished by a syntactic category identifying them as either a function from arguments of one type to results another (aka. function), or as an argument (aka. *primitive category*) [Steedman2000]. CCG is unique among other categorial grammars in its treatment of constituency, long-distant dependency, and binding. According to CCG concept, the core CG was extended with functional operations on adjacent categories — such as functional composition and type-raising operation [Baldridge2003]. CCG is

semantic-transparent, since semantics can be compositionally interpreted during derivation.

Let us exemplify the CG by the following simplified example.

กิน = *(s\np)/np 'to eat'*
ช้าง = *np 'elephant'*
กล้วย = *np 'banana'*

'elephant'	'to eat'	'banana'
ช้าง	กิน	กล้วย
np	(s\np)/np	np
	s\np	
	s	

Figure 1. Example of CG derivation

The notation *α/β* is a rightward-combining functor over a domain of α into a range of *β*. The notation *α\β* is a leftward-combining functor over *β* into *α*. *α* and *β* are both syntactic categories. In example, three words are defined as syntactic categories. The words ช้าง' elephant' and กล้วย 'banana' are defined as a primitive category "np". The word กิน 'to eat' is defined as "(s\np)/np", a function that takes two arguments which both of them are np's; the first one from the right side, and another one from the left side. The derivation of parsing the sentence ช้างกินกล้วย' elephant eats banana' is illustrated in Figure 2. First, the derivation begins by forward combination from (s\np)/np to np yielding the syntactic category s\np. Then, backward combination from s\np to np is applied yielding the syntactic category s.

3. An architecture on Open CCG Workbench and Visualization tool

.At present, OpenCCG, the only one toolkit for developing CCG grammar, is available. OpenCCG is an open-source natural language processing library written in Java. It provides parsing and realization services. The library makes use of the multi-modal extensions to CCGA grammar for OpenCCG consists of five principle files in XML, those are, grammar.xml, lexicon.xml, morph.xml, type.xml and rules.xml.

To fulfil requirements of the grammar development process, OpenCCG workbench and visualization tool was designed and developed. The system architecture is illustrated in Figure 2. The system provides linguists an online interface to work on and supplies lexical resources for the sentence parser and the sentence realizer of the OpenCCG framework. It includes three sub-systems: grammar management system, grammar validation system, and concordance retrieval system. The *grammar management system*, facilitating linguists to visually manage lexical resources, contains three sub-modules: lexicon editor, type-hierarchy editor, and macro editor. The *grammar validation system*, facilitating linguists to examine their grammar, contains two sub-modules: parser tester and realizer tester. The *concordance retrieval system* facilitates linguists to observe word's concordance in the corpora.

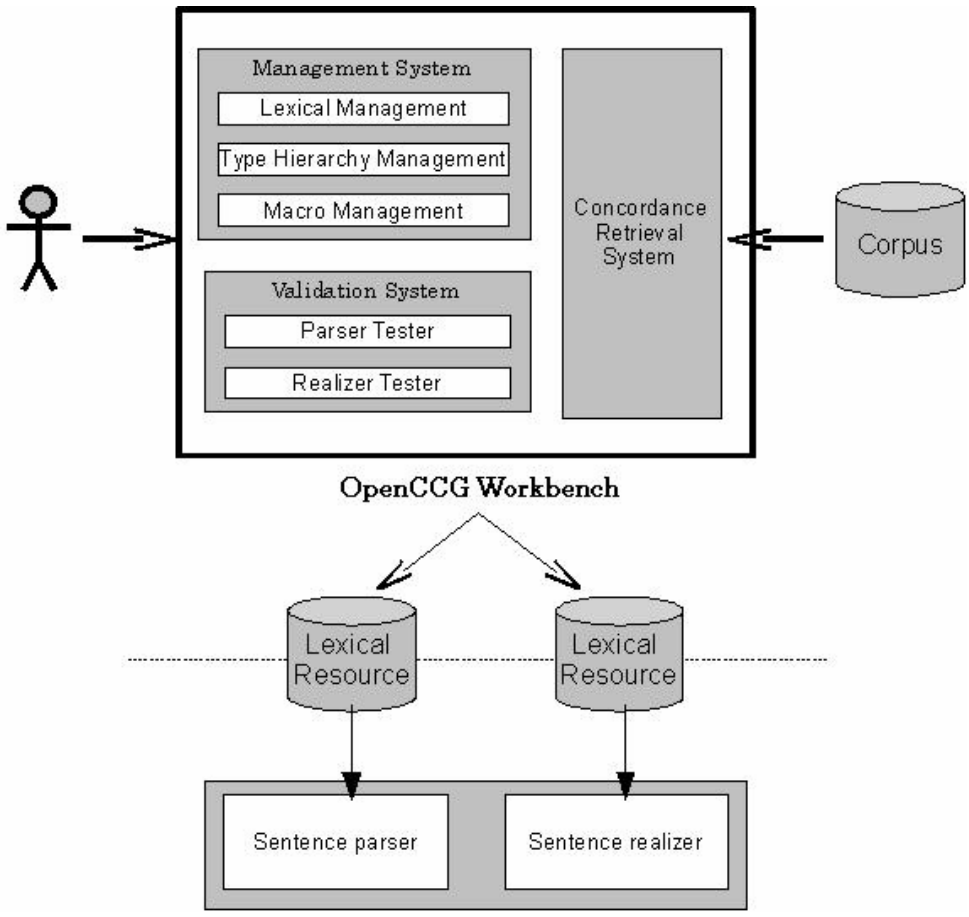

Figure 2 An architecture of CCG workbench and visualization tool

Based on the Open CCG workbench architecture, there are eight use cases for as follows.

1. Grammar management: to schematize the lexical resource; i.e., to design the lexical families, syntactic categories, logical forms, and macros
2. Grammar contribution: to develop the lexical resource with respect to linguists' assignments by adding words and their individual syntax-o-semantic constraints
3. Grammar validation: to examine the lexical resource by expected sentences or ones from word concordance
4. Concordance retrieval: to retrieve word's concordance from existing corpora
5. Grammar importation: to import OpenCCG's XML files into the lexical resource
6. Grammar exportation: to export the lexical resource to OpenCCG's XML files
7. Grammar browsing: to browse the lexical resource
8. System authentication: to authorize users to the system with respect to their access permission

Three levels of users, expert linguists, linguists, and guests are designed to manage the CCG lexical entries. Expert linguists, playing a role as lexical entry designer, can perform all operations. Normal linguists, who add or edit lexicons, can perform operations 2, 3, 4, 6, 7 and 8. Guests, who require an acquisition to the lexicon into their applications, can perform only operations 3, 6, 7 and 8.

4. Implementation

This workbench is implemented as a web application. Since the OpenCCG grammar rules are represented in XML format, Java Server Page (JSP) is used as the server-side mechanism. JSP provides an XML-DOM (XML Document Object Model) library which facilitates reading and manipulating XML documents. For the client side, HTML is applied to implement a powerful, comprehensive, graphical user interface.

The grammar management system facilitates linguists to visually manage lexical resources. It provides the linguists three sub-modules: lexicon editor, type-hierarchy editor, and macro editor. The *lexicon editor* is a visualization tool for editing both syntactic (aka. Syntactic category) and semantics (aka family) lexical information. The *macro editor* is an editor tool for defining macros, constraint feature structures. The *type hierarchy editor* is a graphical tool for managing type hierarchies. Figure 3 represents an example of lexical editor in grammar management system.

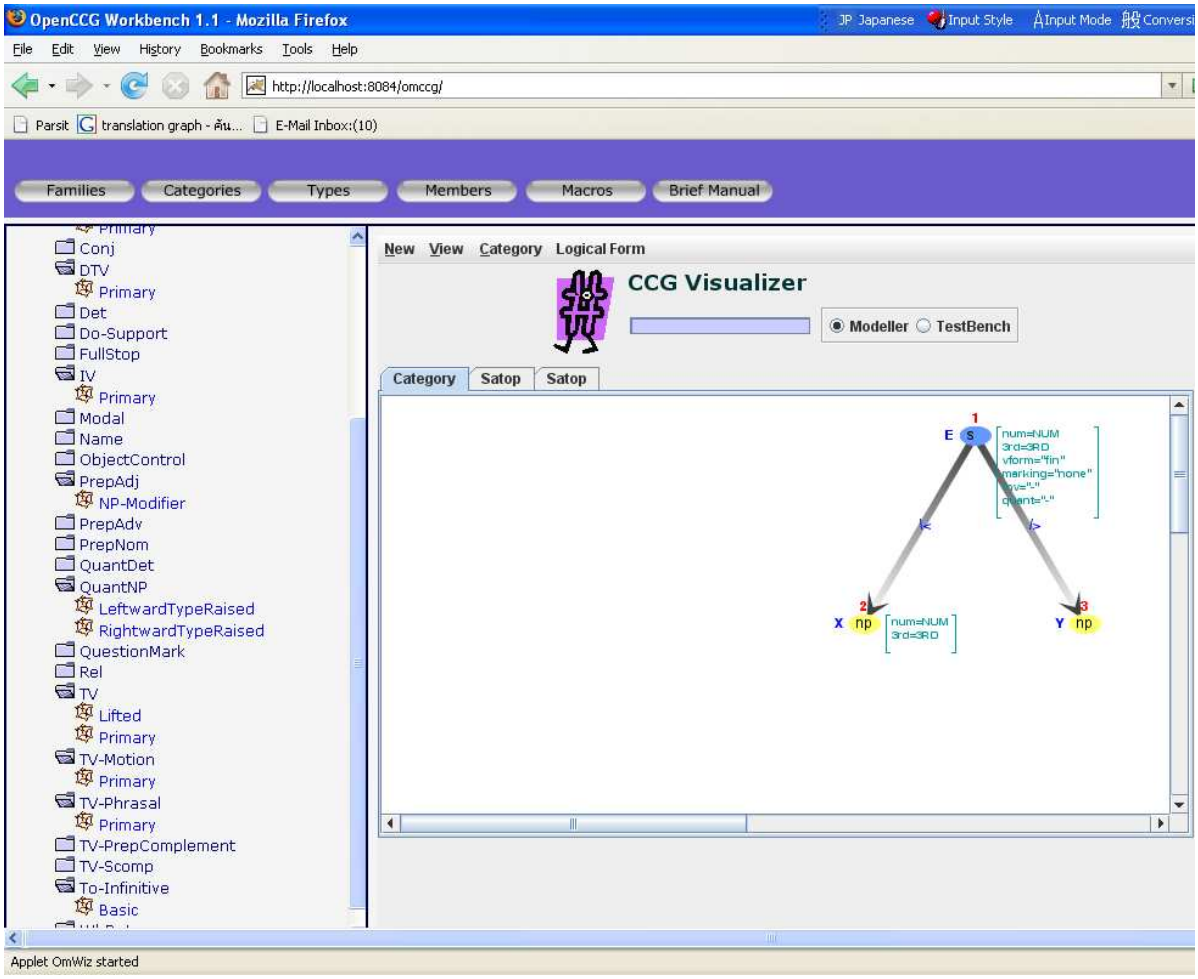

Figure 3 A snapshot of grammar management system in Open CCG Workbench

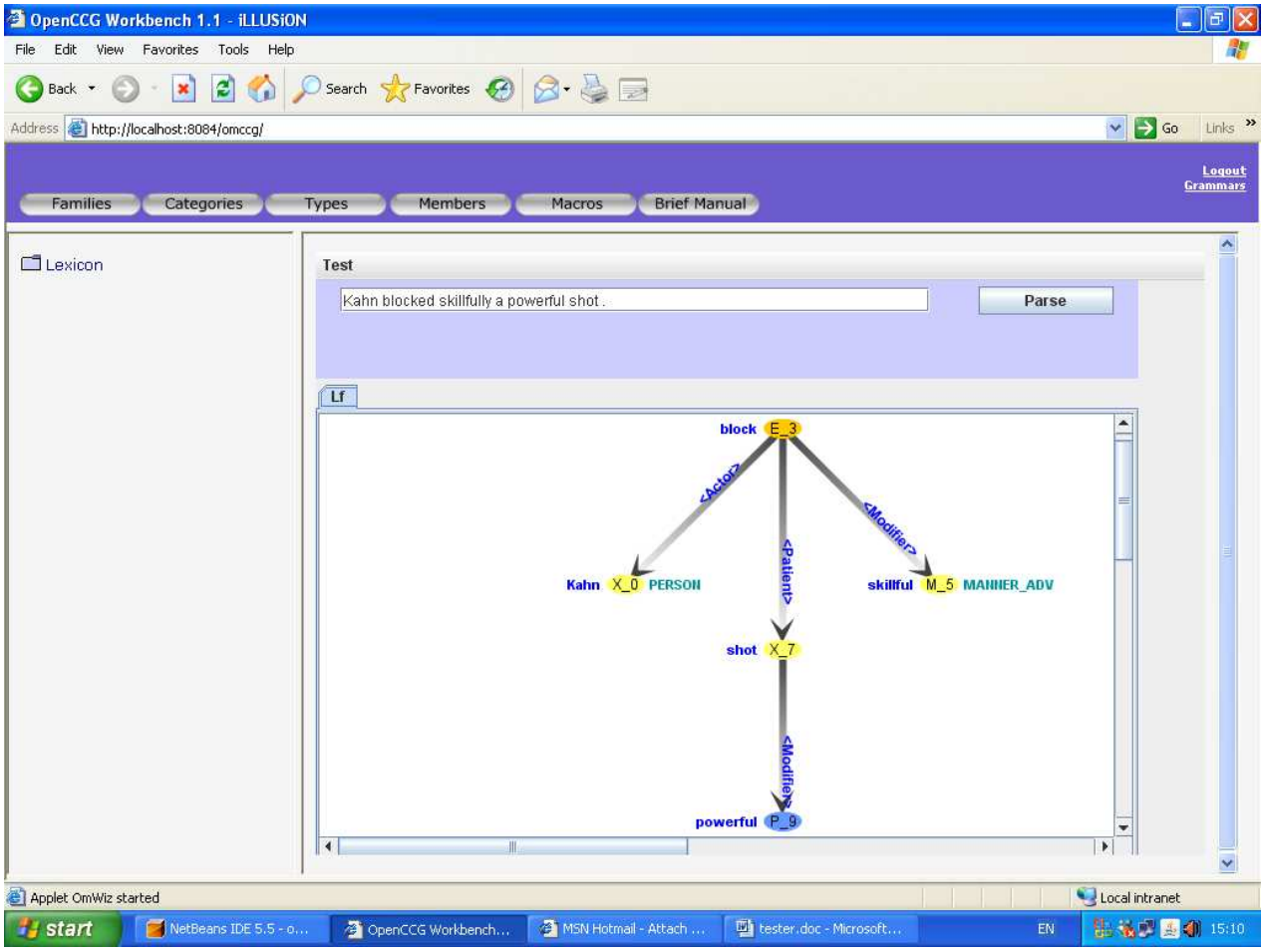

Figure 4 A snapshot of validation system in Open CCG Workbench

We can validate all lexicon components automatically by utilizing various editors using the XML schema specification of the lexicon. However, it is not simple for linguists. The workbench also allows the linguist to ensure that any changes made to the lexicon do not break existing functionality. The validation system facilitates linguists to verify that a given sentence is acceptable under CCG grammar as well as perform regression testing on the test corpus. It provides two sub-modules: parser tester and realizer tester. Parser tester is a graphical tool that describes the correctness of parsing process. Realizer tester is a graphical tool that describes the correctness of realizer process. This adds robustness to the system and allows the linguist to monitor the effects of any changes to the lexicon. Figure 4 represents parser tester in validation system.

5. Conclusion and Future Work

All references within the text should be placed in parentheses containing the author's surname followed by a comma before the date of publication (Martin, 1996). If the sentence already includes the author's name, then it is only necessary to put the date in parentheses: Martin (1996). When several authors are cited, those references should be separated with a semicolon: (Martin, 1996; Chibout & Masson, 1995). When the reference has more than three authors, only cite the name of the first author followed by et al.

This paper presents OpenCCG workbench and visualization tool, an online lexical resource toolkit to facilitate grammar development in OpenCCG. To support the grammar development process, the system was disintegrated into three sub-systems: *grammar management system* for visually managing lexical resources, *validation system* for examining grammar's coverage, and *concordance retrieval system* for observing word's concordance in corpora. At present, the grammar management system and validation system were developed. The concordance retrieval will be developed separately in the future. Currently, we obtain 28 syntactic categories in total. We have got 19,492 annotating content words and 279 annotating function words. In conclusion, 19,771 Thai words are annotated with completed information [Ruangrajitpakorn 2007].

Our future works are as follows. First, we will integrate a concordance retrieval system into this system. Second, we plan to apply the system to practical corpora. Finally, we planned to analyze the lexical resources developed by linguists and to utilize the lexical resources with our machine translation.

References

[Ajdukiewicz1935] Kazimierz Ajdukiewicz. 1935. Die Syntaktische Konnexität. In Storrs McCall, ed., *Polish Logic 1920-1939*, 207–231. Oxford: Oxford University Press. Translated from *Studia Philosophica*, 1, 1–27.

[Baldridge2003] Jason Baldridge. 2003. Chapter 5: Combinatory Categorial Grammar. Draft 4.0 (August 10, 2003), available from homepages.inf.ed.ac.uk/ jbaldrid/ccg.pdf, 2003.

[Bar-Hillel1953] Yehoshua Bar-Hillel. 1953. A Quasi-Arithmetical Notation for Syntactic Description. *Language*, 29, 47–58.

[Steedman2000] Mark Steedman. 2000. *The Syntactic Process*. The MIT Press, Cambridge Massachusette.

[OpenCCG] http://openccg.sf.net.

[Ruangrajitpakorn2007] Taneth Ruangrajitpakorn, Wasan Na chai, Prachya Bookwan, Monthika Boriboon, Thepchai Supnithi. The Design of Lexical information for Thai to English MT. The 6th Symposium of NLP.