A Current Status of Thai Categorial Grammars and Their Applications

Taneth Ruangrajitpakorn and Thepchai Supnithi
Human Language Technology Laboratory
National Electronics and Computer Technology Center

{taneth.ruangrajitpakorn,thepchai.supnithi}@nectec.or.th

Abstract

This paper presents a current status of Thai resources and tools for CG development. We also proposed a Thai categorial dependency grammar (CDG), an extended version of CG which includes dependency analysis into CG notation. Beside, an idea of how to group a word that has the same functions are presented to gain a certain type of category per word. We also discuss about a difficulty of building treebank and mention a toolkit for assisting on a Thai CGs tree building and a tree format representations. In this paper, we also give a summary of applications related to Thai CGs.

1 Introduction

Recently, CG formalism was applied to several Thai NLP applications such as syntactic information for Thai to English RBMT (Ruangrajitpakorn et al., 2007), a CG treebank (Ruangrajitpakorn et al., 2009), and an automatic CG tagger (Supnithi et al., 2010). CG shows promises to handle Thai syntax expeditiously since it can widely control utilisations of function words which are the main grammatical expression of Thai.

In the previous research, CG was employed as a feature for an English to Thai SMT and it resulted better accuracy in term of BLEU score for 1.05% (Porkaew and Supnithi, 2009). CG was also used in a research of translation of noun phrase from English to Thai using phrase-based SMT with CG reordering rules, and it returned 75% of better and smoother translation from human evaluation (Porkaew et al., 2009).

Though CG has a high potential in immediate constituency analysis for Thai, it sill lacks of a dependency analysis which is also important in syntactical parsing. In this paper, we propose a category dependency grammar which is an upgraded version of CG to express a dependency relation alongside an immediate constituency bracketing. Moreover, some Thai dependency banks such as NAIST dependency bank (Satayamas and Kawtrakul, 2004) have been developed. It is better to be able to interchange data between a Thai CG treebank and a Thai dependency bank in order to increase an amount of data since building treebank from scratch has high cost.

In the point of resources and applications, Thai CG and CDG still have a few number of supported tools. Our CG treebank still contains insufficient data and they are syntactically simple and do not reflect a natural Thai usage. To add complex Thai tree, we found that Thai practical usage such as news domain contains a number of word and very complex.

An example of natural Thai text from news, which contains 25 words including nine underlined function words, is instanced with translation in Figure 1.

สำหรับ|การ|วาง|กำลัง|ของ|คน|เสื้อ|แดง| |ได้|มี|การ|วาง|บังเกอร์|รอบ|พื้นที่|ชุมนุม| |เอา|น้ำมัน|ราด| |รวมทั้ง|ยาง|รถยนต์|ที่|เสื่อม|สภาพ|แล้ว

lit: The red-shirts have put bunkers around the assembly area and poured oil and worn-out tires.

Figure 1. An example of Thai usage in natural language

Proceedings of the 8th Workshop on Asian Language Resources, pages 161–168,
Beijing, China, 21-22 August 2010.

We parsed the example in Figure 1 with CG and our parser returned 1,469 trees. The result is in a large number because many Thai structural issues in a syntactic level cause ambiguity.

The first issue is many Thai words can have multiple functions including employing grammatical usage and representing a meaning. For instance, a word "ที่" /tee/ can be a noun, a relative clause marker, a classifier, a preposition, and an adjective marker. A word "คน" /kon/ can refer to a person, a classifier of human being and it can denote an action. A word "กำลัง" /kumlung/ can serve as an auxiliary verb to express progressive aspect and also refers a meaning as a noun. A function word is a main grammatical representation and it hints an analyser to clarify an overall context structure. Regretfully, it is difficult for system to instantly indicate the Thai function words by focusing on the lexical surface and their surrounding lexicons. This circumstance is stimulates an over generation of many improper trees.

The second issue is a problem of Thai verb utilisations. Thai ordinarily allows to omit either a subject or an object of a verb. Moreover, a Thai intransitive verb is occasionally attached its indirect object without a preposition. Furthermore, Thai adjective allows to perform as a predicate without a marker. With an allowance of verb serialisation, these complexify linguists to design a category into well-crafted category set for verb. Therefore, many Thai verbs contain several syntactic categories to serve their many functions.

The last issues is a lack of an explicit boundary for a word, a phrase and a sentence in Thai. A Thai word and phrase boundary is implicit and a space is not significantly signified a boundary in the context. In addition, most of modifiers are attached after a core element. This leads to ambiguity of finding an ending of a subject with an attached adjective and relative clause since the verbs in attachment can be serialised and consequently placed with following main verb phrase (which is likely to be serialised either) without a signified indicator.

With these issues, a parser with only syntactic information merely returns a large number of all possible trees. It becomes difficulty and time consuming for linguists to select the correct one among them. Moreover, with many lexical elements, using a statistical parser has a very low chance to generate a correct tree and a manual tree construction is also required as a gold standard. Thus, we recently implemented an assistant toolkit for tree construction and tree representation to reduce linguists' work load and time consumption.

This paper aims to explain the current status of resource and tool for CG and CDG development for Thai language. We also listed open tools and applications that relate to CGs in this paper.

The rest of the paper is organised as follows. Section 2 presents a Thai categorial grammar and its related formalism. Section 3 explains status of CGs resources including syntactic dictionary and treebank. Section 4 shows details of a toolkit which assists linguist to manage and construct CGs derivation tree and tree representations. Section 5 provides information of applications that involve Thai CGs. Lastly, Section 6 concludes this paper and lists future works.

2 Thai Categorial Grammars

2.1 Categorial Grammar

Categorial grammar (Aka. CG or classical categorial grammar) (Ajdukiewicz, 1935; Carpenter, 1992; Buszkowski, 1998) is a formalism in natural language syntax motivated by the principle of constitutionality and organised according to the syntactic elements. The syntactic elements are categorised in terms of their ability to combine with one another to form larger constituents as functions or according to a function-argument relationship.

CG captures the same information by associating a functional type or category with all grammatical entities. Each word is assigned with at least one *syntactic category*, denoted by an argument symbol (such as *np* and *num*) or a functional symbol *X/Y* and *X\Y* that require *Y* from the right and the left respectively to form *X*.

The basic concept is to find the core of the combination and replace the grammatical modifier and complement with set of categories based on the same concept of the rule of fraction cancellation as follow:

$$\mathrm{np} \times \frac{\mathrm{s}}{\mathrm{np}} = \mathrm{s}$$

Upon applying to Thai, we have modified argument set and designed eight arguments shown in Table 1.

From the last version, two arguments were additionally designed. "*ut*" argument was added to denote utterance that is followed after a word "ว่า". The word "ว่า" has a special function to let the word after it perform as an exemplified utterance and ignore its appropriate category as it is signified an example in context. Comparing to "ws" argument, the word "ว่า" is functioned in a different sense which is used to denote a beginner of subordinate clause.

For "X" category, it is used for punctuation or symbol which takes the same categories from the left or right sides and produces the taken category. For instance, "ๆ" is a marker to denote after many types of content word. In details, this symbol signifies plurality while it is after noun but it intensifies a degree of meaning while it is placed after adjective.

Upon CG design, we allowed only binary bracketing of two immediate constituents. To handle serial construction in Thai including serial verb construction, we permitted the exactly same categories which are consequent to be combined. For example, Thai noun phrase 'มติ(np)|คณะรัฐมนตรี(np)' (lit: a consensus of the government) contains two consequent nouns without a joint word to form a noun phrase. Unfortunately, there still remain limits of syntactic parsing in CG that can not handle long dependency and word omission in this state.

argument category	definition	example
np	a noun phrase	ช้าง (elephant), ผม (I, me)
num	a digit and a spelled-out number	หนึ่ง (one), 2 (two)
spnum	a number which is succeeding to classifier	นึง (one), เดียว (one)
pp	a prepositional phrase	ในรถ (in car), บนโต๊ะ (on table)
s	a sentence	ช้างกินกล้วย (an elephant eats a banana)
ws	a specific category for Thai which is assigned to a sentence or a phrase that begins with Thai word ว่า (that : sub-ordinate clause marker).	* ว่าเขาจะมาสาย 'that he will come late' * ว่าจะมาสาย 'that (he) will come late'
ut	an utterance using to exemplify a specific word after a word ว่า	คำ ว่า ดี 'the word "good"'
X	an undefined category that takes the same categories from the left or right sides and produces the taken category.	เด็ก ๆ (plural marker) สะอาด ๆ (intensifier)

Table 1. A list of Thai CDG arguments

2.2 Categorial Dependency Grammar

Categorial dependency grammar (CDG) is an extension of CG. CDG differs from CG in that a dependency direction motivated by Collins (1999) is additionally annotated to each slash notation in syntactic category. The derivation rules of CDG are listed as follow:

$$X/\!<\!Y : d_1 \; Y : d_2 \Rightarrow X : h(d_1) \rightarrow h(d_2)$$
$$X/\!>\!Y : d_1 \; Y : d_2 \Rightarrow X : h(d_1) \leftarrow h(d_2)$$
$$Y : d_1 \; X\backslash\!<\!Y : d_2 \Rightarrow X : h(d_1) \rightarrow h(d_2)$$
$$Y : d_1 \; X\backslash\!>\!Y : d_2 \Rightarrow X : h(d_1) \leftarrow h(d_2)$$

where the notations $h(d_1) \rightarrow h(d_2)$ and $h(d_1) \leftarrow h(d_2)$ mean a dependency linking from the head of the dependency structure d_1 to the head of d_2, and that linking from the head of d_2 to the head of d_1, respectively. Throughout this paper, a constituent type of the syntactic category *c* and the dependency structure *d* is represented by *c:d*.

Let us exemplify a dependency driven derivation of CDG of sentence 'Mary drinks fresh milk' in Figure 2. In Figure 2(a), each pair of constituents is combined to form a larger constituent with its head word. Figure 2(b) shows a dependency structure equivalent to the derivation in Figure 2(a).

Comparing to PF-CCG (Koller and Kuhlmann, 2009), there is different in that their PF-CCG dependency markers are fixed to the direction of slashes while CDG dependency markers are customised based on behaviour of a constituent.

CDG offers an efficient way to represent dependency structures alongside syntactic derivations. Apart from immediate constituency analysis, we can also investigate the correspondence between the syntactic derivations and the dependency structures. It benefits linguists in details a grammar for a specific language be-

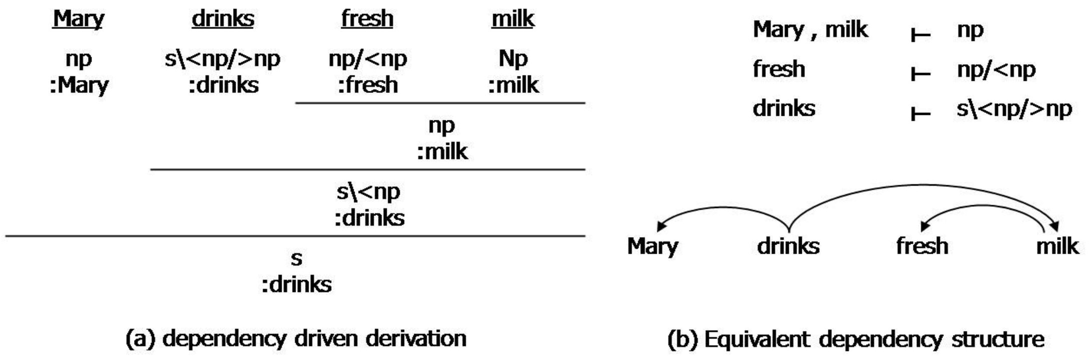

Figure 2. Syntactic derivation of 'Mary drinks fresh milk' based on CDG

cause we can restrain the grammar in lexical level.

In this paper, our Thai CG was applied to CDG. For the case of serial construction, we set the left word as a head of dependency since Thai modifiers and dependents are ordinarily attached on right side.

2.3 Categorial Set

A categorial set is a group of lexicons that exactly contains the same function(s) in terms of their category amount and all their same syntactic categories. With a specific surface, each word certainly is in one categorial set. For example, suppose that we have following words and categories:

word	category	POS
ภูมิทัศน์,ขโมย,ล้อ,เกาะ	⊢ np	noun
ล้อ,เกาะ,ขโมย,กิน,ซื้อ	⊢ s\np/np	verb
ล้อ,เกาะ,ขวบ	⊢ np\np/num	classifier

We can group the given words into five groups based on the concept of categorial set shown in Table 2.

Set-index	Category member	Word member
1	np	ภูมิทัศน์
2	s\np/np	กิน,ซื้อ
3	np s\np/np	ขโมย
4	np s\np/np np\np/num	ล้อ,เกาะ
5	np\np/num	ขวบ

Table 2. An example of categorial set

For current status, we attain 183 categorial sets in total and the maximum amount of category member in a categorial set is 22 categories.

3 Categorial Grammars Resources

To apply categorial grammars to Thai NLP, syntactic dictionary and treebank are a mandatory.

3.1 Categorial Grammars Dictionary

For using in other work and researches, we collected all CGs information into one syntactic dictionary. An example of CGs dictionary is shown in Table 3. In a summary, our Thai CGs dictionary currently contain 70,193 lexical entries with 82 categories for both CG and CDG and 183 categorial sets.

Lexicon	CG	CDG	Cset no.
สมุด	np	np	0
เกาะ	np,s\np/np,np\np/num	np,s\<np/>np,np\>np/<num	15
กิน	s\np/np,s\np	s\<np/>np,s\<np	13
ถ้า	s\s/s,s/s/s	s\<s/>s,s/>s/>s	43
พูด	s\np/pp,s\np,s\np/ws	s\<np/>pp,s\<np,s\<np/>ws	19
เขียว	np\np,s\np	np\>np,s\<np	3
วิ่ง	s\np	s\<np	1
กล้าหาญ	np\np,s\np	np\>np,s\<np	3
นอน	s\np	s\<np	1
ขาย	s\np/np,s\np	s\<np/>np,s\<np	13
เสื้อ	np	np	0
ว่า	s\np/np,s\np/ws,np\np/ut	s\<np/>np,s\<np/>ws,np\>np/>ut	136
เพราะ	s\s/s,s/s/s	s\<s/>s,s/>s/>s	43

Table 3. An example of Thai CGs dictionary

3.2 Thai CDGTreebank

Our CG treebank was recently transformed into dependency-driven derivation tree with CDG. An example of derivation tree of sentence |การ|

ล่า|เสือ|เป็น|การ|ผจญภัย| 'lit: Tiger hunting is an adventure' comparing between CG and CDG is illustrated in Figure 3.

```
s
 (np
  (np/(s\np)[การ]
  s\np(
   (s\np)/np[ล่า]
   np[เสือ]
  )
 )
 s\np(
  (s\np)/np[เป็น]
  np(
   np/(s\np)[การ]
   s\np[ผจญภัย]
  )
 )
)
```

(a) CG derivation tree

```
s
 (np
  (np/>(s\<np)[การ]
  s\<np(
   (s\<np)/>np[ล่า]
   np[เสือ]
  )
 )
 s\<np(
  (s\<np)/>np[เป็น]
  np(
   np/>(s\<np)[การ]
   s\<np[ผจญภัย]
  )
 )
)
```

(b) CDG derivation tree

Figure 3. An example of a derivation tree in treebank comparing between CG and CDG

A status of transformed CDG treebank is 30,340 text lines which include 14,744 sentences, 9,651 verb phrases or subject-omitted sentences and 5,945 noun phrases. However, the average word amount of this treebank is 3.41 words per tree which is obviously short.

Upon an attempt to increase a number of trees, we considered that the existing trees are simple and not signify all utilisations of natural Thai text. Therefore, news domain of BEST (Kosawat et al., 2009) corpus was chosen to fulfil such issues because of its practical usage. From our observation, we found that most of data are ranged from 25 to 68 words and the longest line contains 415 words which is extremely long for parser to handle it efficiently.

After a prior experiment, we found that our GLR parser with CDG information resulted 514.62 tree alternatives in average from the range of three to fifteen words per sentence from news domain in BEST. With problems from ambiguous syntax of Thai, to automatically select a correct tree is extremely difficult since several resulted trees are grammatically correct and semantically sound but they are not proper for their context. It becomes difficulty for linguists to select an appropriate one among them. In order to solve that problem, we implemented a toolkit to assist linguists on constructing treebank with such a long and complicated sentence. The manual annotated tree will be used as a gold standard and confidentially apply for statistical parser development.

4 CGs Tree Supported Tool

Building a resource is a laboured work especially a treebank construction. For Thai language which uses several function words to express grammatical function in context, an immediate constituency analysis and a dependency analysis become difficult since many word pair can cause ambiguity and complexity among them. Additionally, a representation of a derivation tree in textual format is excessively complex to be analysed or approved. To reduce a burden of linguists, we developed a toolkit to help a linguists with graphical user-interface in manual tree construction.

4.1 CGs Toolkit

The proposed toolkit supports multi-tasks which are annotating CG tag to a word, bracketing intermediate constituents, generating dependency-driven derivation tree in multiple formats, and visualising graphical tree.

4.1.1 Category Annotator

Category annotator supports users to select an appropriate CDG category for each word. The system takes word-segmented input text. It starts with checking possible categories with the given CDG dictionary and lists all of them to each word. Users only select a correct category for each. Unless the word is known or the required category for the word is present, user has to add a new category for the word and the system contiguously updates the dictionary with the given data for further usage.

4.1.2 Dependency-driven Derivation Tree Generator

This system is implemented for manual annotating tree information and dependency relation to a text that is difficult for parser to generate tree such as a text with multiple serial verb constructions, a complex head-dependent relation word pairs, etc. A captured picture of user-interface

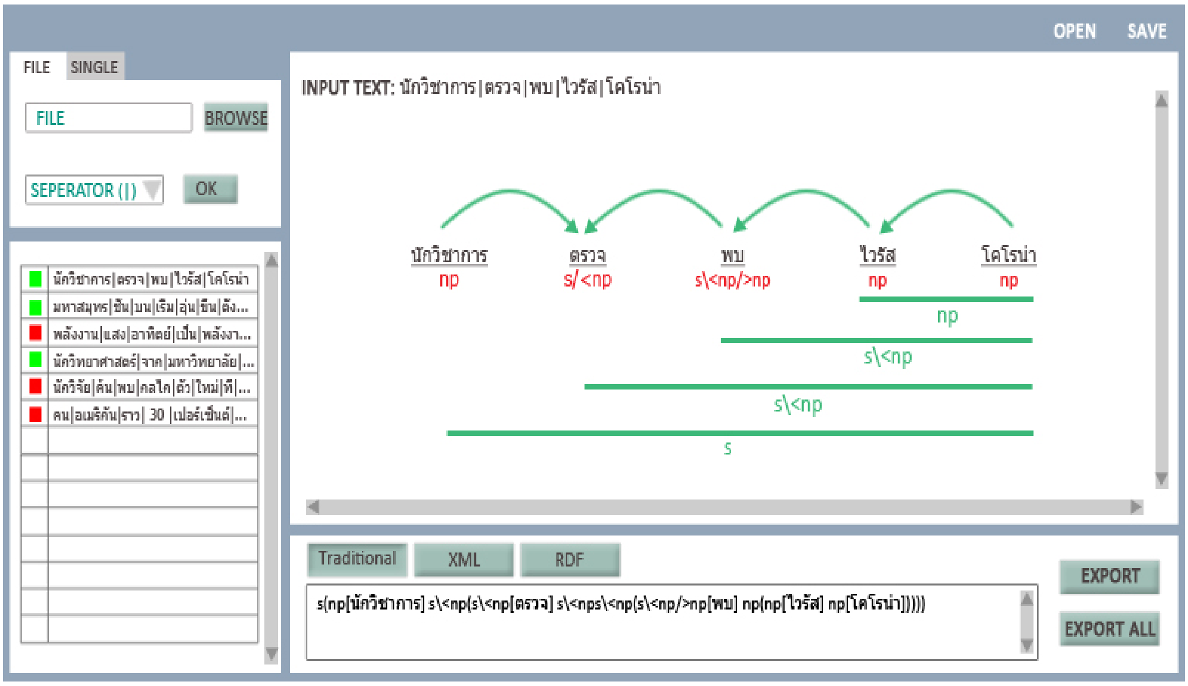

Figure 4. A snapshot of dependency-driven derivation tree generator

working on immediate constituency and dependency annotation is illustrated in Figure 4.

We provide a user-interface for linguists and experts to easily annotate brackets covering. Users begin a process by selecting a pair of words that are a terminal of leaf node. The system apparently shows only categories of the word that can be possibly combined within the bracket for selecting. After choosing categories of those two constituents, the system automatically generates a result category. Next, users will continue the process for other constituents until one top result category is left.

After users finish the bracketing process, dependency relation will be generated from annotated dependency marker within categories without manual assignment.

4.1.3 Tree Visualiser

The system includes a function to create a graphical tree from a file in textual formats. It provides a function to modify a tree by editing a word spelling and its syntactic category and shifting a branch of syntactic tree to another.

4.2 Tree Representation

The CGs toolkit allows users to export a tree output in two representations; traditional textual tree format and XML format.

Throughout all tree format examples, we exemplify a Thai sentence 'นักวิชาการ ตรวจ พบ ไวรัส โคโรน่า' (lit: an expert discovers corona virus.) with the following categories:

Word	CDG category
นักวิชาการ (expert) ไวรัส (virus) โคโรน่า (corona)	├ np
ตรวจ (diagnose)	├ s\<np
พบ (discover)	├ s\<np/>np

4.2.1 Traditional Textual Tree Format

A traditional textual tree format represents a terminal (*w*) with its category (*c*) in form of *c[w]*. The brackets are enclosed two constituents split by space with parentheses and the result category (c_r) is placed before the open parenthesis in format c_r*(c[w] c[w])*. Figure 5 shows an example of a traditional textual tree format.

s(np[นักวิชาการ] s\<np(s\<np[ตรวจ] s\<np(s\<np(s\<np/>np[พบ] np(np[ไวรัส] np[โคโรน่า])))

Figure 5. An example of a traditional textual tree format of 'นักวิชาการ ตรวจ พบ ไวรัส โคโรน่า'

4.2.2 XML Tree Format

For XML tree format, we design three tag sets, i.e., *word* tag, *tree* tag and *input* tag. The *word* tag bounds a terminal to mark a word. In a start-tag of *word* tag, there are two attributes which are *cat* to assign a category in a value and text to assign a given text in a value. For *tree* tag, it marks a combination of either *word* tags or *tree* tags to form another result category. It contains two previous attributes with an additional attribute, i.e., a *head* attribute to fill in a notation that which word has a head-outward relation value where '0' value indicates head from left constituent and '1' value indicates head from right constituent. The input tag shows a boundary of all input and it has attributes to show line number, raw input text and status of tree building process. Figure 6 illustrates an XML tree representation.

5 Thai CGs Related Applications

Several applications related to Thai CGs or used Thai CGs as their syntactic information have been implemented recently. Below is a summary of their methodology and result.

5.1 CG AutoTagger for Thai

To reduce an amount of trees generated from a parser with all possible categories, an automatic syntactic category tagger (Supnithi et al., 2010) was developed to disambiguate unappropriated combinations of impossible categories. The system was developed based on CRF and Statistical Alignment Model based on information theory (SAM) algorithm. The accuracy 89.25% in word level was acquired. This system also has a function to predict a syntactic category for an unknown word and 79.67% of unknown word are predicted correctly.

5.2 Chunker

With a problem of a long sentence in Thai, chunker was implemented to group a consequent of words to larger unit in order to reduce a difficulty on parsing too many lexical elements. CRD method with syntactic information from CG and categorial set was applied in the system to chunk a text into noun phrase, verb phrase, prepositional phrase, and adverbial phrase. Moreover, the system also attempts to handle a compound word that has a form like sentence. The result was impressive as it improved 74.17% of accuracy on sentence level chunking and 58.65% on sentence-form like compound noun.

5.3 GLR parser for Thai CG and CDG

Our implemented LALR parser (Aho and Johnson, 1974) was improved to GLR parser for syntactically parse Thai text. This parser was developed to return all possible trees form input to show a baseline that covers all syntactic possibilities. For our GLR parser, a grammar rule is not manually determined, but it is automatically produced by any given syntactic notations aligned with lexicons in a dictionary. Hence, this GLR parser has a coverage including CG and CDG formalism parsing. Furthermore, our GLR parser accepts a sentence, a noun phrase, a verb phrase and prepositional phrase. However, the parser does not only return the best first tree, but also all parsable trees to gather all ambiguous trees since Thai language tends to be ambiguous because of lacking explicit sentence, phrase and word boundary. This parser includes a pre-process to handle named-entities, numerical expression and time expression.

```
- <input id="103" text="นักวิชาการ|ตรวจ|พบ|ไวรัส|โคโรน่า" status="approved">
  - <tree cat="s" head="1" text="นักวิชาการ|ตรวจ|พบ|ไวรัส|โคโรน่า">
      <word cat="np" text="นักวิชาการ" />
    - <tree cat="s\{np" head="0" text="ตรวจ|พบ|ไวรัส|โคโรน่า">
        <word cat="s\{np" text="ตรวจ" />
      - <tree cat="s\{np" head="0" text="พบ|ไวรัส|โคโรน่า">
          <word cat="s\{np/}np" text="พบ" />
        - <tree cat="np" head="0" text="ไวรัส|โคโรน่า">
            <word cat="np" text="ไวรัส" />
            <word cat="np" text="โคโรน่า" />
          </tree>
        </tree>
      </tree>
    </tree>
  </input>
```

Figure 6. An example of XML tree format of 'นักวิชาการ ตรวจ พบ ไวรัส โคโรน่า'

6 Conclusion and Future Work

In this paper, we update our Thai CG information and a status of its resources. We also propose CDG for Thai, an extended version of CG. CDG offers an efficient way to represent dependency structures with syntactic derivations. It benefits linguists in terms of they can restrain Thai grammar in lexical level. With CDG dependency-driven derivation tree, both bracketing information and dependency relation are annotated to every lexical units. In the current state, we transformed our CG dictionary and CG treebank into CDG formalism.

With an attempt to increase an amount of our treebank with a complex text, CDG tree toolkit was developed for linguists to manual managing a derivation tree. This toolkit includes a CDG category tagger tool, dependency-driven derivation tree generator, and tree visualiser. This toolkit can generate an output in two formats which are traditional textual tree and XML tree. The XML tree format is an option for standardised format or further usage such as applying tree for ontology.

We also summarised CGs related works and their accuracy. They included an automatic CG tagger and a Thai phrase chunker.

In the future, we plan to increase an amount of CGs derivation trees of complex sentence and practical language. Moreover, we will implement a system to transform an existing Thai dependency bank to CDG format to gain more number of trees. We also plan to include semantic meaning into derivation tree and represent such trees in an RDF format. In addition, statistical parser will be implemented based on the CDG derivation trees.

References

Ajdukiewicz Kazimierz. 1935. Die Syntaktische Konnexitat, Polish Logic.

Aho Alfred, and Johnson Stephen. 1974. LR Parsing, Proceedings of Computing Surveys, Vol. 6, No. 2.

Bar-Hillel Yehoshua. 1953. A quasi-arithmetical notation for syntactic description. 29(1): 47-58.

Carpenter Bob. 1992. Categorial Grammars, Lexical Rules,and the English Predicative, In R. Levine, ed., Formal Grammar: Theory and Implementation. OUP.

Collins Micheal. 1999. Head-Driven Statistical Models for Natural Language Parsing. Ph.D. Thesis, University of Pennsylvania.

Koller Alexander, and Kuhlmann Marco. 2009. Dependency trees and the strong generative capacity of ccg, Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: 460-468.

Kosawat Krit, Boriboon Monthika, Chootrakool Patcharika, Chotimongkol Ananlada, Klaithin Supon, Kongyoung Sarawoot, Kriengket Kanyanut, Phaholphinyo Sitthaa, Purodakananda Sumonmas, Thanakulwarapas Tipraporn, and Wutiwiwatchai Chai. 2009. BEST 2009: Thai Word Segmentation Software Contest. The 8th International Symposium on Natural Language Processing: 83-88.

Porkaew Peerachet, Ruangrajitpakorn Taneth, Trakultaweekoon Kanokorn, and Supnithi Thepchai.. 2009. Translation of Noun Phrase from English to Thai using Phrase-based SMT with CCG Reordering Rules, Proceedings of the 11th conference of the Pacific Association for Computational Linguistics (PACLING).

Porkaew Peerachet, and Supnithi Thepchai. 2009. Factored Translation Model in English-to-Thai Translation, Proceedings of the 8th International Symposium on Natural Language Processing.

Ruangrajitpakorn Taneth, Na Chai Wasan , Boonkwan Prachya, Boriboon Monthika, Supnithi Thepchai. 2007. The Design of Lexical Information for Thai to English MT, Proceedings of the 7th International Symposium on Natural Language Processing.

Ruangrajitpakorn Taneth, Trakultaweekoon Kanokorn, and Supnithi Thepchai. 2009. A Syntactic Resource for Thai: CG Treebank, Proceedings of the 7th Workshop on Asian Language Resources, (ACL-IJCNLP): 96–102.

Satayamas Vee, and Kawtrakul Asanee . 2004. Wide-Coverage Grammar Extraction from Thai Treebank. Proceedings of Papillon 2004 Workshops on Multilingual Lexical Databases, Grenoble, France.

Supnithi Thepchai, Ruangrajitpakorn Taneth, Trakultaweekoon Kanokorn, and Porkaew Peerachet. 2010. AutoTagTCG : A Framework for Automatic Thai CG Tagging, Proceedings of the 7th international conference on Language Resources and Evaluation (LREC).