# Extraction of Tree Adjoining Grammars from a Treebank for Korean

**Jungyeul Park**
UFR Linguistique
Laboratoire de linguistique formelle
Université Paris VII - Denis Diderot
jungyeul.park@linguist.jussieu.fr

## Abstract

We present the implementation of a system which extracts not only lexicalized grammars but also feature-based lexicalized grammars from Korean *Sejong* Treebank. We report on some practical experiments where we extract TAG grammars and tree schemata. Above all, full-scale syntactic tags and well-formed morphological analysis in *Sejong* Treebank allow us to extract syntactic features. In addition, we modify Treebank for extracting lexicalized grammars and convert lexicalized grammars into tree schemata to resolve limited lexical coverage problem of extracted lexicalized grammars.

## 1 Introduction

An electronic grammar is an interface between the complexity and the diversity of natural language and the regularity and the effectiveness of a language processing, and it is one of the most important elements in the natural language processing. Since traditional manual grammar development is a time-consuming and labor-intensive task, many efforts for automatic and semi-automatic grammar development have been taken during last decades.

Automatic grammar development means that a system extracts a grammar from a Treebank which has an implicit Treebank grammar. The grammar extraction system takes syntactically analyzed sentences as an input and produces a target grammar. The extracted grammar would be same as the Treebank grammar or be different depending on the user's specific purpose. The automatically extracted grammar has the advantage of the coherence of extracted grammars and the rapidity of its development. However, as it always depends on the Treebank which the extraction system uses, its coverage could be limited to the scale of a Treebank. Moreover, the reliable Treebank would be hardly found, especially in public domain.

Semi-automatic grammar development means that a system generates the grammar using the description of the language-specific syntactic (or linguistic) variations and its constraints. A metagrammar in Candito (1999) and a tree description in Xia (2001) are good examples of a semi-automatic grammar development. Even using semi-automatic grammar development, we need the good description of linguistic phenomena for specific language which requires very high level knowledge of linguistics and the semi-automatically generated grammars would easily have an overflow problem.

Since we might extract the grammar automatically without many efforts if a reliable Treebank is provided, in this paper we implement a system which extracts a Lexicalized Tree Adjoining Grammar and a Feature-based Lexicalized Tree Adjoining Grammar from Korean *Sejong* Treebank (SJTree). SJTree contains 32,054 *eojeol*s (the unity of segmentation in the Korean sentence), that is, 2,526 sentences. SJTree uses 43 part-of-speech tags and 55 syntactic tags.

Even though there are many previous works for extracting grammars from a Treebank, extracting syntactic features is tried for the first time. 55 full-scale syntactic tags and well-formed morphological analysis in SJTree allow us to extract syntactic features automatically and to develop FB-LTAG.



*Proceedings of the COLING/ACL 2006 Student Research Workshop*, pages 73–78,
Sydney, July 2006. 

First, we briefly present features structures which are focused on FB-LTAG and other previous works for extracting a grammar from a Treebank. Then, we explain our grammar extraction scheme and report experimental results. Finally, we discuss the conclusion.

## 2 Feature structures and previous works on extracting grammars from a Treebank

A feature structure is a way of representing grammatical information. Formally feature structure consists of a specification of a set of features, each of which is paired with a particular value (Sag *et al.*, 2003). In a unification frame, a feature structure is associated with each node in an elementary tree (Vijay-Shanker and Joshi, 1991). This feature structure contains information about how the node interacts with other nodes in the tree. It consists of a top part, which generally contains information relating to the super-node, and a bottom part, which generally contains information relating to the sub-node (Han *et al.*, 2000).

In FB-LTAG, the feature structure of a new node created by substitution inherits the union of the features of the original nodes. The top feature of new node is the union of the top features ($f_1 \cup f$) of the two original nodes, while the bottom feature of the new node is simply the bottom feature ($g_1$) of the top node of the substituting tree since the substitution node has no bottom feature as shown in Figure 1.

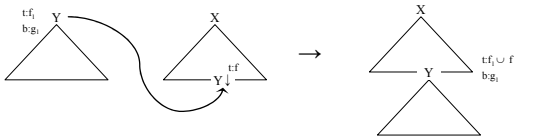



Figure 1. Substitution in FB-LTAG

The node being adjoined into splits and its top feature (f) unifies with the top feature ($f_1$) of the root adjoining node, while its bottom feature (g) unifies with the bottom feature ($g_2$) of the foot adjoining node as shown in Figure 2.

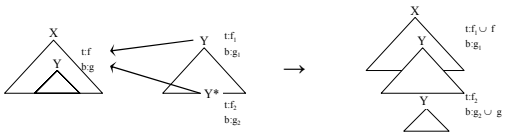



Figure 2. Adjunction in FB-LTAG

Several works for extracting grammars, especially for TAG formalism are proposed. Chen (2001) extracted lexicalized grammars from English Penn Treebank and there are other works based on Chen's procedure such as Johansen (2004) and Nasr (2004) for French and Habash and Rambow (2004) for Arabic. Chiang (2000) used Tree Insertion Grammars, one variation of TAG formalism for his extraction system from English Penn Treebank. Xia *et al*. (2000) developed the uniform method of a grammar extraction for English, Chinese and Korean. Neumann (2003) extracted Lexicalized Tree Grammars from English Penn Treebank for English and from NEGRA Treebank for German. As mentioned above, none of these works tried to extract syntactic features for FB-LTAG.

## 3 Grammar extraction scheme

Before extracting a grammar automatically, we transform the bracket structure sentence in SJTree into a tree data structure. Afterward, using depth-first algorithm for a tree traverse, we determine a head and the type of operations (substitution or adjunction) for children nodes of the given node if the given node is a non-terminal node.

### 3.1 Determination of a head

For the determination of a head, we assume the right-most child node as a head among its sibling nodes in end-focus languages like Korean. For instance, the second NP is marked as a head in [NP NP] composition while the first NP is marked for adjunction operation for the extracted grammar $G_1$ which uses *eojeol*s directly without modification of SJTree (see the section 4 for the detail of extraction experiments). Likewise, in [VP@VV VP@VX] composition where the first VP has a VV (verb) anchor and the last VP has a VX (auxiliary verb) anchor, a principal verb in the first VP could be marked for adjunction operation and an auxiliary verb in the second VP would be a head, that is, the extracted auxiliary verb tree has every argument of whole sentence. This phenomenon could be explained by argument composition. Head nodes of the extracted grammar for a verb *balpyoha.eoss.da* ('announced') in (1) are in bold face in Figure 3 which represents bracketed sentence structure in SJTree

(1) 일본 외무성은 즉각 해명 성명을 발표했다.
*ilbon* *oimuseong.eun*
Japan ministy_of_foreign_affairs.Nom
*jeukgak* *haemyeng*
immediately elucidation
*seongmyeng.eul* *balpyo.ha.eoss.da*
declaration.Acc announce.Pass.Ter
'The ministry of foreign affairs in Japan immediately announced their elucidation.'

(S (NP_SBJ (NP *ilbon*/NNP)
(NP_SBJ *oimuseong*/NNG+*eun*/JX))
(**VP** (AP *jeukgak*/MAG)
(**VP** (NP_OBJ (NP *haemyeng*/NNG)
(NP_OBJ *seonmyeng*/NNG+*eul*/JKO))
(**VP** *balpyo*/NNG+*ha*/XSV+*eoss*/EP+*da*/EF+./SF))))

Figure 3. Bracketed sentence in SJTree for (1)

### 3.2 Distinction between substitution and adjunction operations

Unlike other Treebank corpora such as English Penn Treebank and French Paris 7 Treebank, full-scale syntactic tags in SJTree allow us to easily determine which node would be marked for substitution or adjunction operations. Among 55 syntactic tag in SJTree, nodes labeled with NP (noun phrase), S (sentence), VNP (copular phrase) and VP (verb phrase) which end with _CMP (attribute), _OBJ (object), and _SJB (subject) would be marked for substitution operation, and nodes labeled with the other syntactic tags except a head node would be marked for adjunction operation. In this distinction, some VNP and VP phrases might be marked for substitution operation, which means that VNP and VP phrases are arguments of a head, because SJTree labels VNP and VP instead of NP for the nominalization forms of VNP and VP. In Figure 4, for example, NP_SBJ and NP_OBJ nodes are marked for substitution operation and AP node is marked for adjunction operation.

Children nodes marked for substitution operation are replace by substitution terminal nodes (e.g. NP_SBJ↓) and calls recursively the extraction procedure with its subtree where a root node is the child node itself. Children nodes marked for adjunction operation are removed from the main tree and also calls recursively the extraction procedure with its subtree where we add its parent node of a given child node as a root node and a sibling node as a foot node (e.g. VP*). As defined in the TAG formalism, the foot node has the same label as the root node of the subtree for an adjunction operation.

### 3.3 Reducing trunk

Extracted grammars as explained above are not always "correct" TAG grammar. Since nodes marked for adjunction operation are removed, there remain intermediate nodes in the main tree. In this case, we remove these redundant nodes. Figure 4 shows how to remove the redundant intermediate nodes from the extracted tree for a verb *balpyoha.eoss.da* ('announced') in (1).

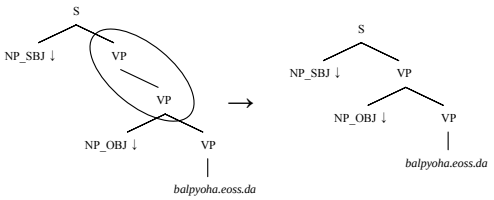



Figure 4. Removing redundant intermediate nodes from extracted trees

### 3.4 Extracting features

55 full-scale syntactic tags and morphological analysis in SJTree allow us to extract syntactic features automatically and to develop FB-LTAG. Automatically extracted FB-LTAG grammars eventually use reduced tagset because FB-LTAG grammars contain their syntactic information in features structures. For example, NP_SBJ syntactic tag in LTAG is changed into NP and a syntactic feature <case=subject> is added. Therefore, we use actually 13 reduced tagset for FB-LTAG grammars. From full-scale syntactic tags which end with _SBJ (subject), _OBJ (object) and _CMP (attribute), we extract <case> features which describe argument structures in the sentence.

Alongside <case> features, we also extract <mode> and <tense> from morphological analyses in SJTree. Since however morphological analyses for verbal and adjectival endings in SJTree are simply divided into EP, EF and EC which mean non-final endings, final endings and conjunctive endings, respectively, <mode> and <tense> features are not extracted directly from SJTree. In this paper, we analyze 7 non-final endings (EP) and 77 final endings (EF) used in SJTree to extract automatically <mode> and <tense> features. In general, EF carries <mode> inflections, and EP carries <tense> inflections. Conjunctive endings (EC) are not concerned with <mode> and <tense> features and we only extract <ec> features with its string value. <ef> and <ep> features are also extracted

with their string values. Some of non-final endings like *si* are extracted as <hor> features which have honorary meaning. In extracted FB-LTAG grammars, we present their lexical heads in a bare infinitive with morphological features such as <ep>, <ef> and <ec> which make correspond with its inflected forms.

<det> is another automatically extractable feature in SJTree and it is extracted from both syntactic tag and morphological analysis unlike other extracted features. For example, while <det=-> is extracted from dependant nouns which always need modifiers (extracted by morphological analyses), <det=+> is extracted from _MOD phrases (extracted by syntactic tags). From syntactic tag DP which contains MMs (determinative or demonstrative), <det=+> is also extracted[1].

The actual procedure of feature extraction is implemented by 2 phases. In the first phase, we convert syntactic tags and morphological analysis into feature structure as explained above. In the second phase, we complete feature structure onto nodes of dorsal spine. For example, we put the same feature of VV bottom onto VV top, VP top/bottom and S bottom because nodes in dorsal spine share certain number of feature of VV bottom. The initial tree for a verb *balpyoha.eoss.da* is completed like Figure 5 for a FB-LTAG (see Park (2006) for details).

[1] Korean does not need features <person> as in English and <gender > or <number> as in French. Han *et al*. (2000) proposed several features for Korean FBLTAG which we do not use in this paper, such as <adv-pp>, <top> and < aux-pp> for nouns and <clause-type> for predicates. While postpositions are separated from *eojeol* during our grammar extraction procedure, Han *el al*. considered them as "one" inflectional morphology of noun phrase *eojeol*. As we will explain the reason why we separate postpositions from *eojeol* in the section 4, the separation of postpositions would be much efficient for the lexical coverage of extracted grammars. In Han *et al*. <adv-pp> simply contains string value of adverbial postpositions. <aux-pp> adds semantic meaning of auxiliary postpositions such as *only*, *also* etc. which we can not extract automatically from SJTree or other Korean Treebank corpora because syntactically annotated Treebank corpora generally do not contain such semantic information. <top> marks the presence or absence of a topic marker in Korean like *neun*, however topic markers are annotated like a subject in SJTree which means that only <case=subject> is extracted for topic markers. <clause-type> indicates the type of the clause which has its values such as main, coord(inative), subordi(native), adnom(inal), nominal, aux-connect. Since the distinction of the type of the clause is very vague except main clause in Korea, we do not adopt this feature. Instead <ef> is extracted if a clause type is a main clause and <ec> is extracted for other type.

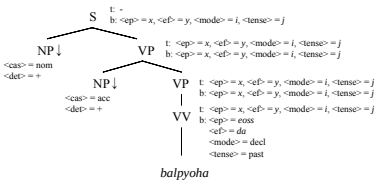



Figure 5. Extracted FB-LTAG grammar for *balpyoha.eoss.da* ('announced')

## 4 Extraction experiments and results

### 4.1 Extraction of lexicalized trees

In this paper, we extract not only lexicalized trees without modification of a Treebank, but also extract grammars with modifications of a Treebank using some constraints to improve the lexical coverage in extracted grammars.

- $G_1$: Using *eojeol*s directly without modification of SJTree.
- $G_2$: Separating symbols and postpositions from *eojeol*s. Separated symbols are extracted and divided into α and β trees based on their types. Every separated postposition is α tree. Complex postpositions consisted of two or more postpositions are extracted like one α tree[2]. Finally, converting NP β trees into α trees and removing syntactic tag in NP α trees.

Figure 6 and 7 show extracted lexicalized grammars $G_1$ and $G_2$ from (1) respectively. Theoretically extracting order is followed by word order in the sentence.

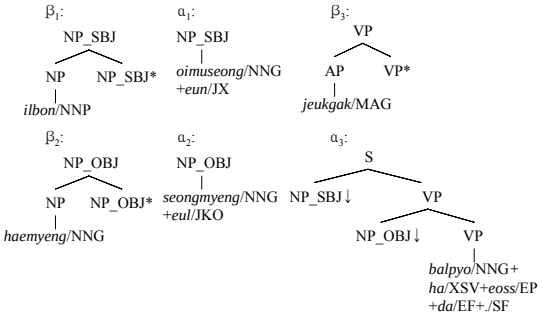



Figure 6. Extracted lexicalized grammars $G_1$

[2] For extracting trees of symbols and of postposition, we newly add SYM and POSTP syntactic tags which SJTree does not use. See Figure 11 for extracted symbol and postposition trees.

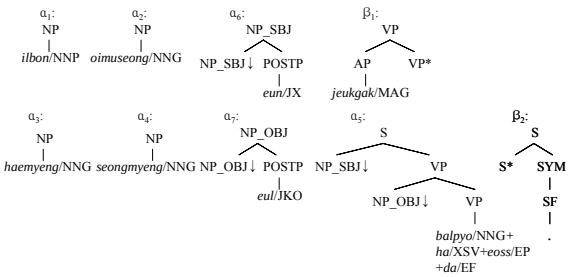



Figure 7. Extracted lexicalized grammars $G_2$

### 4.2 Extraction of feature-based lexicalized trees

We extract feature-based lexicalized trees using reduced tagset because FB-LTAG grammars contain their syntactic information in features structures. Extracted grammars $G_3$ remove syntactic tags, eventually use reduced tagset, add extracted feature structures and use infinitive forms as lexical anchor.

- $G_3$: Using reduced tagset and a lexical anchor is an infinitive and adding extracted feature structures.

$G_3$ row in Table 1 below shows the results of extraction procedures above. Figure 8 shows extracted feature-based lexicalized grammars $G_3$ from (1)

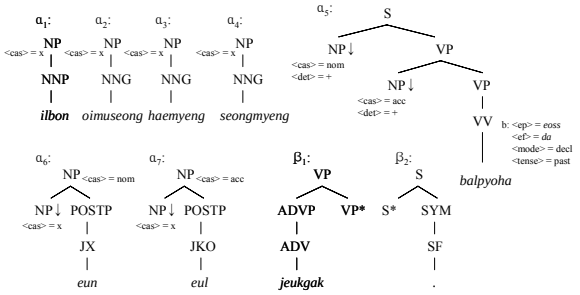



Figure 8. Extracted feature-based lexicalized grammars $G_3$ [3].

| | # of `ltrees` (lexicalized tree) | Average frequencies per `ltrees` |
|---|---|---|
| $G_1$ | 18,080 | 1.38 |
| $G_2$ | 15,551 | 2.57 |
| $G_3$ | 12,429 | 3.21 |

Table 1. Results of experiments in extracting lexicalized and feature-based lexicalized grammars

[3] To simplify the figure, we note only feature structure which is necessary to understand.

### 4.3 Extraction of tree schemata

As mentioned in the Introduction, one of the most serious problems in automatic grammar extraction is its limited lexical coverage. To resolve this problem, we enlarge our extracted lexicalized grammars using templates which we call tree schemata. The lexical anchor is removed from extracted grammars and anchor mark is replaced to form tree schemata (for example, @NNG where the lexicalized anchor in extracted lexicalized grammars is a common noun). The number of tree schemata is much reduced against that of lexicalized grammars. Table 2 shows the number of template trees and the average frequency for each template grammars. $T_1$ means $G_1$'s tree schemata.

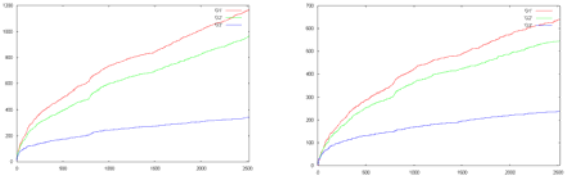
| | # of tree schemata | Average frequencies per tree schemata |
|---|---|---|
| $T_1$ | 1,158 | 21.55 |
| $T_2$ | 1,077 | 37.05 |
| $T_3$ | 385 | 103.65 |

Table 2. Results of experiments in converting template grammars

## 5 Evaluations

First of all, the lexical coverage for $G_1$ and $G_2$ is tested on the part of *Sejong* corpus which contains about 770,000 "morphologically analyzed" *eojeol*s. After modification of SJTree, the extracted grammar $G_2$ is increased to 17.8 % compared with $G_1$ for its lexical coverage. $G_2$ and $G_3$ have same lexical coverage since they have same lexical entries.

Extracted grammars in this paper are evaluated by its size and its coverage. The size of grammars means tree schemata according to the number of sentences as shown in Figure 9. The coverage of grammar is the number of occurrences of unknown tree schemata in the corpus by the total occurrences of tree schemata as shown in Table 3.



(a) Threshold =1 (b) Threshold =2

Figure 9. The size of grammars

| | Threshold = 1 | Threshold = 2 |
|---|---|---|
| $G_1$ | 0.9326 | 0.9591 |
| $G_2$ | 0.9326 | 0.9525 |
| $G_3$ | 0.9579 | 0.9638 |

Table 3. Coverage of grammars: 90% of training set (2,273 sentences) and 10% of test set (253 sentences)

We manually overlap our 163 tree schemata for predicates from $T_3$, which contain 14 subcategorization frames with 11 subcategorization frames of a FB-LTAG grammar proposed in Han *et al*. (2000) to evaluate the coverage of hand-crafted grammars [4]. Our extracted template grammars cover 72.7 % of their hand-crafted subcategorization frames[5].

## 6 Conclusion

In this paper, we have presented a system for automatic grammar extraction that produces lexicalized and feature-based lexicalized grammars from a Treebank. Also, to resolve the problem of limited lexical coverage of extracted grammars, we separated symbols and postposition, and then converted these grammars into template grammars. Extracted grammars and lexical-anchor-less template grammars might be used for parsers to analyze the Korean sentences and frequency information might be used to remove ambiguities among possible syntactic analyses of parsers.

## References


Candito, Marie-Hélène. 1999. *Organisation modulaire et paramétrable de grammaire électronique lexicalisées*. Ph.D. thesis, Université Paris 7.

Chen, John. 2001. *Towards Efficient Statistical Parsing Using Lexicalized Grammatical Information*. Ph.D. thesis, University of Delaware.

Chiang, David. 2000. Statistical Parsing with an Automatically-Extracted Tree Adjoining Grammar. In *Data Oriented Parsing*, CSLI Publication, pp. 299-316.

Habash, Nizar and Owen Rambow. 2004. Extracting a Tree Adjoining Grammar from the Penn Arabic Treebank. In *Proceedings of Traitement Automatique du Langues Naturelles (TALN-04)*. Fez, Morocco, 2004.

Han, Chunghye, Juntae Yoon, Nari Kim, and Martha Palmer. 2000. *A Feature-Based Lexicalized Tree Adjoining Grammar for Korean*. IRCS Technical Report 00-04. University of Pennsylvania.

Johansen, Ane Dybro. 2004. *Extraction des grammaires LTAG à partir d'un corpus étiquette syntaxiquement*. DEA mémoire, Université Paris 7.

Nasr, Alexis. 2004. *Analyse syntaxique probabiliste pour grammaires de dépendances extraites automatiquement*. Habilitation à diriger des recherches, Université Paris 7.

Neumann, Günter. 2003. A Uniform Method for Automatically Extracting Stochastic Lexicalized Tree Grammar from Treebank and HPSG, In A. Abeillé (ed) *Treebanks: Building and Using Parsed Corpora*, Kluwer, Dordrecht.

Park, Jungyeul. 2006. *Extraction d'une grammaire d'arbres adjoints à partir d'un corpus arboré pour le coréen*. Ph.D. thesis, Université Paris 7.

Sag, Ivan A., Thomas Wasow, and Emily M. Bender. 2003. *Syntactic Theory: A Formal Introduction*, 2nd ed. CSLI Lecture Notes.

Vijay-Shanker, K. and Aravind K. Joshi. 1991. Unification Based Tree Adjoining Grammar, in J. Wedekind ed., *Unification-based Grammars*, MIT Press, Cambridge, Massachusetts.

Xia, Fei, Martha Palmer, and Aravind K. Joshi. 2000. A Uniform Method of Grammar Extraction and Its Application. In *The Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)*, Hong Kong, Oct 7-8, 2000.

Xia, Fei. 2001. *Automatic Grammar Generation from Two Different Perspectives*. Ph.D. thesis, University of Pennsylvania, PA.


[4] Our extracted tree schemata contain not only subcategorization frames but also some phenomena of syntactic variations, the number of lexicalized trees and the frequency information while Han *el al*. (2000) only presents subcategorization frames and some phenomena.

[5] Three subcategorization frames in Han *el al*. (2000) which contain prepositional phrases are not covered by our extracted tree schemata. Generally, prepositional phrases in SJTree are labeled with _AJT which is marked for adjunction operation. Since there is no difference between noun adverbial phrase and prepositional phrases in SJTree like [S *na.neun* [NP_AJT *ojeon.e* 'morning'] [NP_AJT *hakgyo.e* 'to school'] *ga.ss.da*] ('I went to school this morning'), we do not consider _AJT phrases as arguments.