A Feature-Based Lexicalized Tree Adjoining Grammar for Korean

Chung-hye Han, Juntae Yoon, Nari Kim and Martha Palmer

Institute for Research in Cognitive Science
University of Pennsylvania
3401 Walnut St., Suite 400A
Philadelphia, PA 19104

http://www.cis.upenn.edu/~xtag/koreantag

Sept., 2000

Contents

Abstract

This document describes an on-going project of developing a grammar of Korean, the Korean XTAG grammar, written in the TAG formalism and implemented for use with the XTAG system enriched with a Korean morphological analyzer. The Korean XTAG grammar described in this report is based on the TAG formalism (Joshi et al. (1975)), which has been extended to include lexicalization (Schabes et al. (1988)), and unification-based feature structures (Vijay-Shanker and Joshi (1991)). The document first describes the modifications that we have made to the XTAG system (The XTAG-Group (1998)) to handle rich inflectional morphology in Korean. Then various syntactic phenomena that can be currently handled are described, including adverb modification, relative clauses, complex noun phrases, auxiliary verb constructions, gerunds and adjunct clauses. The work reported here is a first step towards the development of an implemented TAG grammar for Korean, which is continuously updated with the addition of new analyses and modification of old ones.

Acknowledgment

We are extremely grateful to Aravind Joshi for his continued support and encouragement. We also thank the XTAG group at Penn for numerous useful discussions and comments on our approach. In particular, we thank Anoop Sarkar for his help with modifying the XTAG system for parsing Korean, and Fei Xia for valuable discussions on many occasions. Special thanks are also due to Tony Kroch and Owen Rambow for their contributions to some of the analyses.

The work reported in this document was partially supported by contract DAAD 17-99-C-0008 awarded by the Army Research Lab to CoGenTex, Inc., with the University of Pennsylvania as a subcontractor and NSF Grant - VerbNet, IIS 98-00658. Nari Kim and Juntae Yoon's contributions to this project were made when they were postdoctoral fellows at IRCS, UPenn.

Chapter 1

Introduction

This technical report describes a Feature-Based, Lexicalized Tree Adjoining Grammar for Korean (Korean FB-LTAG) currently being developed at the University of Pennsylvania. In Chapter 2, we first briefly present some formal background concerning feature-based lexicalized tree adjoining grammar (FB-LTAG) which is necessary in understanding the Korean FB-LTAG as well as the rest of the technical report. Chapter 3 contains an overview of the Korean XTAG system that comprises the morphological analyzer and the parser and a discussion of how they interact with the grammar. Then in Chapter 4, we describe how the grammar handles inflectional and derivational morphology. The actual description of the grammar begins with Chapter 5, in which various TREE FAMILIES are described, where each tree family roughly corresponds to a subcategorization frame. In Chapter 6, we discuss how some of the linguistic phenomena are handled in the grammar.

Chapter 2

Feature-Based, Lexicalized Tree Adjoining Grammars

The Korean grammar described in this report is based on the TAG formalism (Joshi et al. (1975)), which has been extended to include lexicalization (Schabes et al. (1988)), and unification-based feature structures (Vijay-Shanker and Joshi (1991)). Tree Adjoining Languages (TALs) fall into the class of mildly context-sensitive languages, and as such are more powerful than context-free languages. The TAG formalism in general, and lexicalized TAGs in particular, are well-suited for linguistic applications. As first shown by Joshi (1985) and Kroch and Joshi (1987), the properties of TAGs permit us to represent diverse syntactic phenomena in a very natural way. For example, TAG's extended domain of locality and its factoring of recursion from local dependencies lead, among other things, to a localization of so-called unbounded dependencies.

2.1 TAG formalism

The primitive elements of the standard TAG formalism are known as elementary trees. ELEMENTARY TREES are of two types: initial trees and auxiliary trees (see Figure 2.1). In describing natural language, INITIAL TREES are minimal linguistic structures that contain no recursion, i.e. trees containing the phrasal structure of simple sentences, NP's, and so forth. Initial trees are characterized by the following: 1) all internal nodes are labeled by non-terminals, 2) all leaf nodes are labeled by terminals, or by non-terminal nodes marked for substitution. An initial tree is called an X-type initial tree if its root is labeled with type X.

Recursive structures are represented by AUXILIARY TREES, which represent constituents that are adjuncts to basic structures (e.g. adverbials). They can also represent recursive portions of the grammar such as long distance or cyclic movement. Auxiliary trees are characterized as follows: 1) all internal nodes are labeled by non-terminals, 2) all leaf nodes are labeled by terminals, or by non-terminal nodes marked for substitution, except for exactly one non-terminal node, called the foot node, which can only be used to adjoin the tree to

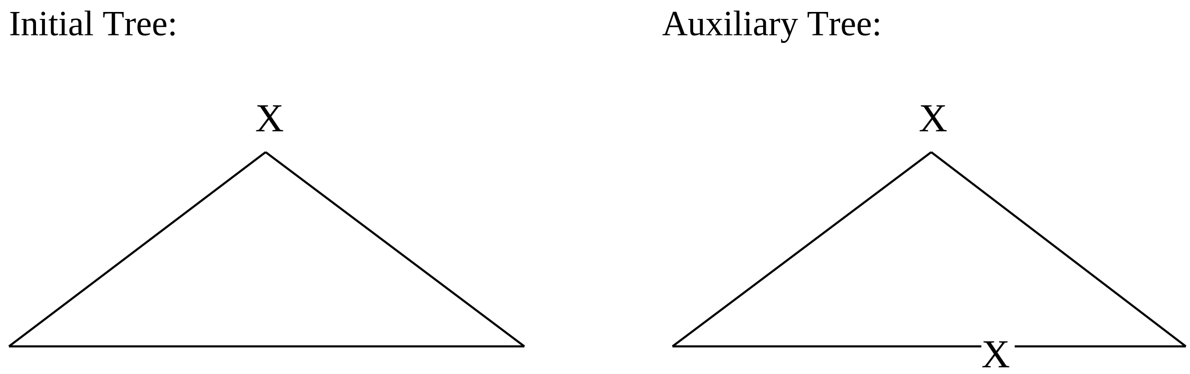

Figure 2.1: Elementary trees in TAG

another node[1], 3) the foot node has the same label as the root node of the tree.

There are two operations defined in the TAG formalism, substitution[2] and adjunction. In the SUBSTITUTION operation, the root node on an initial tree is merged into a non-terminal leaf node marked for substitution in another initial tree, producing a new tree. The root node and the substitution node must have the same name. Figure 2.2 shows two initial trees and the tree resulting from the substitution of one tree into the other.

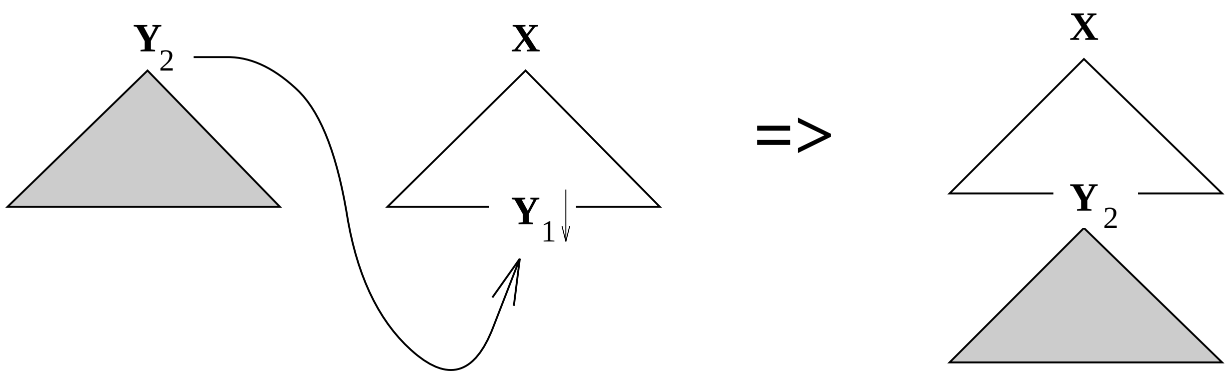

Figure 2.2: Substitution in TAG

In an ADJUNCTION operation, an auxiliary tree is grafted onto a non-terminal node anywhere in an initial tree. The root and foot nodes of the auxiliary tree must match the node at which the auxiliary tree adjoins. Figure 2.3 shows an auxiliary tree and an initial tree, and the tree resulting from an adjunction operation.

A TAG G is a collection of finite initial trees, I, and auxiliary trees, A. The TREE SET of a TAG G, $\mathcal{T}(G)$ is defined to be the set of all derived trees starting from S-type initial trees

[1]A null adjunction constraint (NA) is systematically put on the foot node of an auxiliary tree. This disallows adjunction of a tree onto the foot node itself.

[2]Technically, substitution is a specialized version of adjunction, but it is useful to make a distinction between the two.

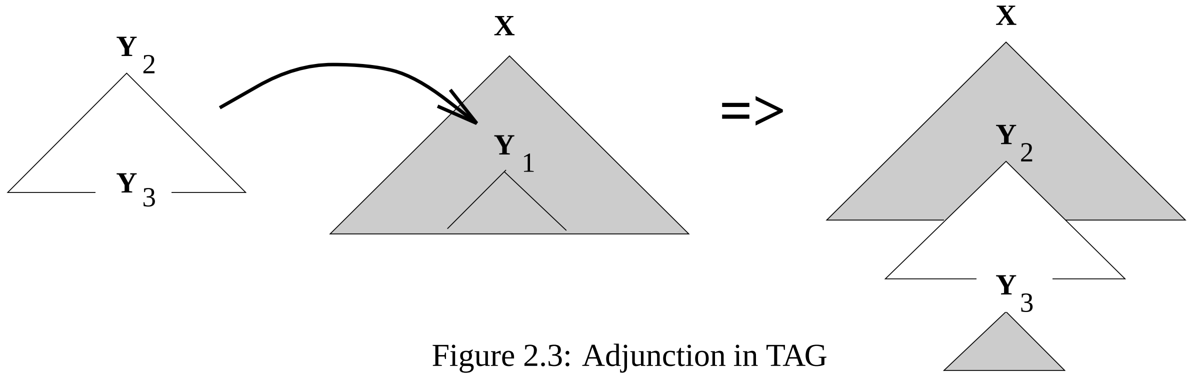

Figure 2.3: Adjunction in TAG

in I whose frontier consists of terminal nodes (all substitution nodes having been filled). The STRING LANGUAGE generated by a TAG, $\mathcal{L}(G)$, is defined to be the set of all terminal strings on the frontier of the trees in $\mathcal{T}(G)$.

2.2 Lexicalization

'Lexicalized' grammars systematically associate each elementary structure with a lexical anchor. This means that in each structure there is a lexical item that is realized. It does not mean simply adding feature structures (such as head) and unification equations to the rules of the formalism. These resultant elementary structures specify extended domains of locality (as compared to CFGs) over which constraints can be stated.

Following Schabes et al. (1988) we say that a grammar is LEXICALIZED if it consists of 1) a finite set of structures each associated with a lexical item, and 2) an operation or operations for composing the structures. Each lexical item will be called the ANCHOR of the corresponding structure, which defines the domain of locality over which constraints are specified. Note then, that constraints are local with respect to their anchor.

Not every grammar is in a lexicalized form.[3] In the process of lexicalizing a grammar, the lexicalized grammar is required to be strongly equivalent to the original grammar, i.e. it must produce not only the same language, but the same structures or tree set as well.

Examples of initial trees and the substitution operation being performed on these trees are given in Figure 2.4. Substitution sites are marked by a $\downarrow$.

In Figure 2.4, $\alpha 1$, $\alpha 2$ and $\alpha 3$ are initial trees for the transitive verb *meknunta* ('eat'), and the proper noun *chelswuka* and a noun *sakwalul* ('apple') respectively. After $\alpha 2$ and $\alpha 3$ substitute at NP_0 and NP_1, we get the derived tree Γ for the sentence in (1).

[3]Notice the similarity of the definition of a lexicalized grammar with the off line parsability constraint (Kaplan and Bresnan (1983)). As consequences of our definition, each structure has at least one lexical item (its anchor) attached to it and all sentences are finitely ambiguous.

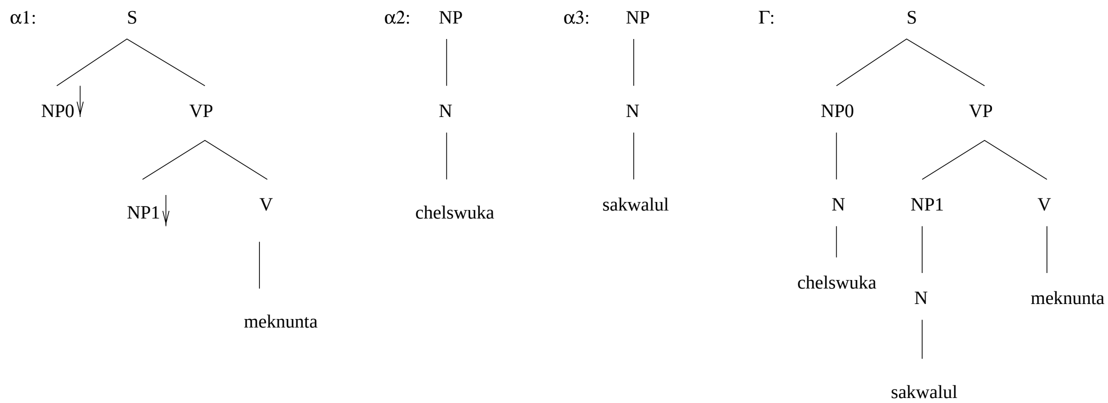

Figure 2.4: Initial trees for *chelswu*, *sakwalul* ('apple') and the derived tree for *Chelswuka sakwalul meknunta* ('Chelswu eats an apple.')

(1) Chelswu-ka sakwa-lul mek-nun-ta.
Chelswu-Nom apple-Acc eat-Pres-Dec
'Chelswu eats an apple.'

Examples of an auxiliary tree and its adjunction operation on an initial tree are given in Figure 2.5. Foot nodes are marked by an asterisk ($*$). In Figure 2.5, $\beta 1$ is an auxiliary tree for the adverb *cacwu* ('often') and $\alpha 1$ is an initial tree for the intransitive verb *nolayhanta* ('sing'). After the adjunction operation of $\beta 1$ on $\alpha 1$, we get the derived tree Γ for *cacwu nolayhanta* ('often sing').

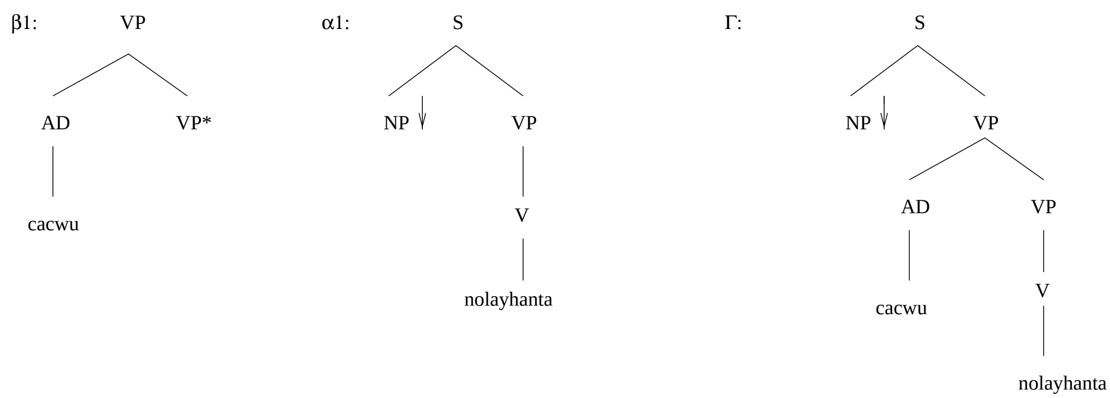

Figure 2.5: Auxiliary tree for *cacwu* ('often') and initial tree for *nolayhanta* ('sing'), and derived tree for *cacwu nolayhanta* ('often sing')

2.3 Unification-based features

In a unification framework, 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 supernode, and a bottom part, which generally contains information relating to the subnode. Substitution nodes, however, have only the top features.

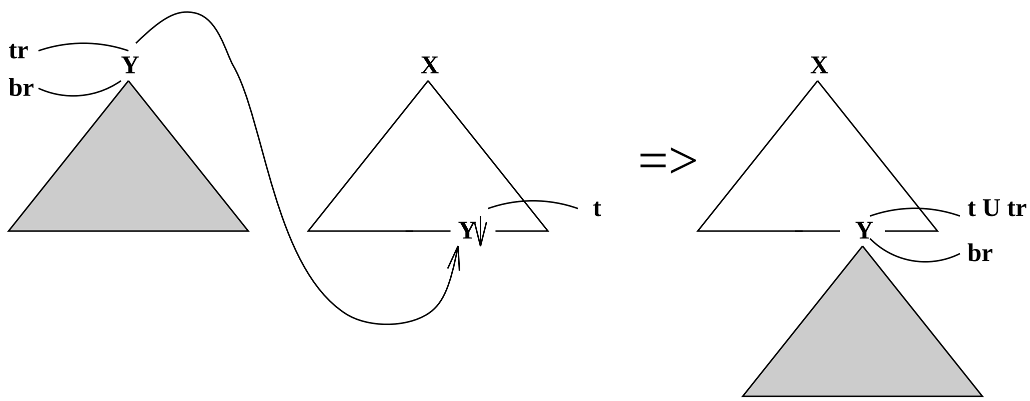

Figure 2.6: Substitution in FB-LTAG

The notions of substitution and adjunction must be augmented to fit within this new framework. The feature structure of a new node created by substitution inherits the union of the features of the original nodes. The top feature of the new node is the union of the top features of the two original nodes, while the bottom feature of the new node is simply the bottom feature of the top node of the substituting tree (since the substitution node has no bottom feature). Figure 2.6[4] shows this more clearly.

Adjunction is only slightly more complicated. The node being adjoined into splits, and its top feature unifies with the top feature of the root adjoining node, while its bottom feature unifies with the bottom feature of the foot adjoining node. Again, this is easier shown graphically, as in Figure 2.7[5].

The embedding of the TAG formalism in a unification framework allows us to dynamically specify local constraints that would have otherwise had to have been made statically within the trees. Constraints that verbs make on their complements, for instance, can be implemented through the feature structures. The notions of Obligatory and Selective Adjunction, crucial to the formation of lexicalized grammars, can also be handled through the

[4]Abbreviations in the figure: t=top feature structure, tr=top feature structure of the root, br=bottom feature structure of the root, U=unification

[5]Abbreviations in the figure: t=top feature structure, b=bottom feature structure, tr=top feature structure of the root, br=bottom feature structure of the root, tf=top feature structure of the foot, bf=bottom feature structure of the foot, U=unification

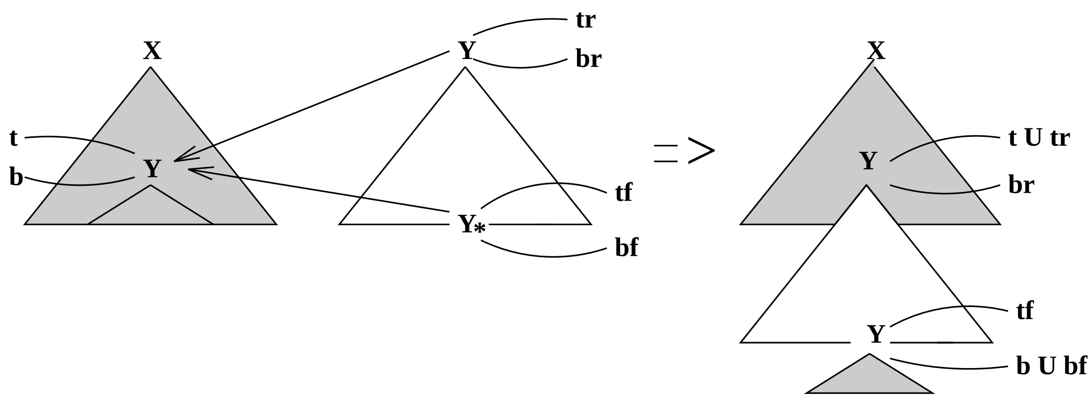

Figure 2.7: Adjunction in FB-LTAG

use of features.[6] Perhaps more important to developing a grammar, though, is that the trees can serve as a schemata to be instantiated with lexical-specific features when an anchor is associated with the tree. The use of feature structures is discussed in more detail in Section 4.1, and throughout the rest of the technical report.

[6]The remaining constraint, Null Adjunction (NA), must still be specified directly on a node.

Chapter 3

Overview of the Korean XTAG System

This section describes various components of the Korean XTAG system including the morphological analyzer and the parser and how they interact with the Korean grammar. We share the parser (LTAG parser) and the grammar development system with the English XTAG system, which can be exploited for various LTAG systems. While having access to the English XTAG system facilitated the construction of the Korean FB-LTAG, it was also necessary for us to modify and customize some of the modules in the existing XTAG system to efficiently handle Korean.

We therefore mainly describe novel parts in the Korean XTAG system and some of the differences between the Korean and the English XTAG systems, instead of explaining the parser in detail. See The XTAG-Group (1998) for the English XTAG system components and Sarkar (2000) for the LTAG parser. Persons interested only in the linguistic analyses in the Korean FB-LTAG may skip this section without loss of continuity, although a quick glance at the tagset used in Korean grammar and the set of non-terminal labels used will be useful.

3.1 System Description

Figure 3.1 shows the overall flow of the Korean XTAG system when parsing a sentence; a summary of the database used is presented in Table 3.1. See Yoon et al. (1999) for a more detailed description of the morphological analyzer and Sarkar (2000) for the parser. The current Korean XTAG system uses the morphological analyzer developed by Yoon et al. (1999) and the parser described in Sarkar (2000). In contrast to the older Common LISP XTAG parser which we refer to as XTAG parser, this new parser (written in C) which is called LEM can accept Korean characters in 2 byte code. Also in our Korean parsing system the databases are used differently (as described later in this chapter).

LEM was meant to use the XTAG English grammar (The XTAG-Group (1998)), and so it uses the lexical databases that are a part of the English grammar. In the English grammar, all the morphological variants of each word are listed in the morphological database (Morph DB), where they are mapped to a stem and lexical feature structures. The stem is then used

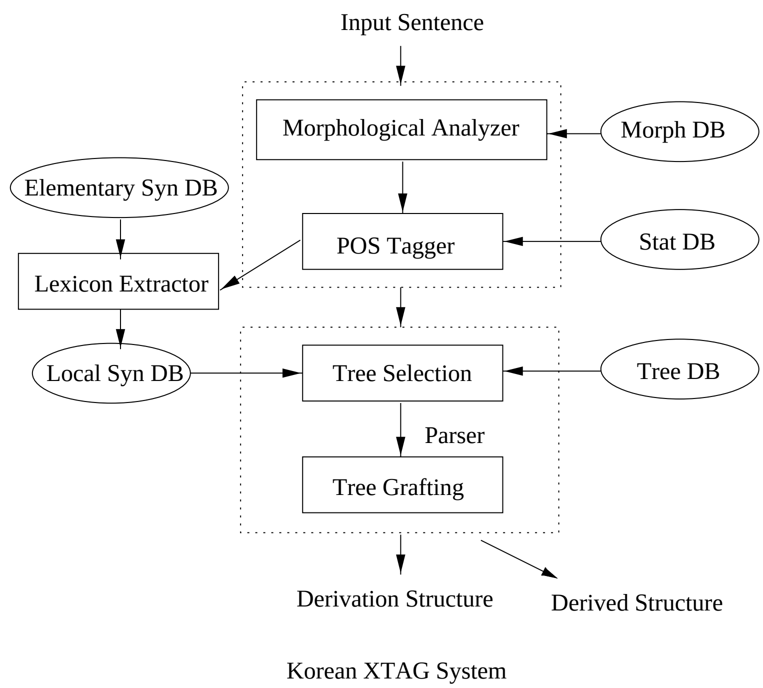

Figure 3.1: Overview of Korean XTAG system

Component	Details
Syntactic Database (ESDB)	Each entry consists of: the uninflected form of the content word in Korean, its POS, the list of trees or tree-families associated with the word, and a list of feature equations that capture lexical idiosyncrasies.
Tree Database	289 trees, divided into 15 tree families. Tree families represent subcategorization frames.

Table 3.1: Summary of core parts in the Korean XTAG system

to select a set of elementary trees in the syntactic database. The older XTAG parser keeps these databases separate, but LEM combines them into a single database. LEM uses this database (Syn DB) in order to select appropriate trees for the words in the input sentence.

Since a word in English has a small number of inflections, it is possible to describe as separate entries all the inflected forms in the Syn DB. However, this way of describing lexicons for Korean is inefficient, due to its agglutinative morphology. That is, Korean has a very large number of inflections, resulting in a large number of morphological variants for a given word. To resolve this problem, the Korean XTAG system divides the Syn DB into Elementary Syn DB (ESDB) and Local Syn DB (LSDB). The parser accesses LSDB which is constructed from the ESDB using the result of morphological analysis results. ESDB and LSDB are described in more detail in Section 3.1.1.

3.1.1 Tree Selection

In Lexicalized TAGs, each word in the sentence selects at least one tree. As is well-known, the advantage of a lexicalized formalism like LTAGs is that the parser can parse with only the trees selected by the words in the input sentence rather than parsing with all the trees in the grammar.

Morphological Analysis and POS Tagging The input sentence first goes through the morphological analyzer and the POS tagger. In parsing English, the morphological analysis step is simple. All the inflected forms are included with their feature information in **Morph DB**. For Korean, the morphological analysis is very important since a word in Korean has a much larger number of morphological variants. In the current Korean XTAG system, we use the morphological analyzer and the POS tagger developed in Yonsei University (Yoon et al. (1999)).

Unlike the Morph DB in the English XTAG system, the Morph DB in the Korean XTAG system contains information for online morphological analysis. The morphological analyzer uses the Morph DB to produce morphological constituents which is then fed to the Syn DB to lexicalize the word.

Syntactic Database The syntactic database contains the mapping between particular stem(s) and the tree templates or tree-families stored in the **Tree Database** (see Table 3.1). The syntactic database also contains a list of feature equations that capture lexical idiosyncrasies (The XTAG-Group (1998)).

The Syn DB in Korean XTAG system consists of the **Elementary Syn DB** (**ESDB**) and the **Local Syn DB** (**LSDB**). The ESDB contains only stems of words entered by grammar developers, and is not directly accessed by the parser. The LSDB is dynamically generated by the **Lexicon Extractor** from the morphological analyses of *eojeols* in a sentence.[1]

From the results of morphological analysis, the lexicon extractor associates lexicalized trees and features structures with each eojeol. Lexicalized trees are associated with words by looking up the ESDB and feature structures are associated with inflections by the morphological analysis. With this data collected, it generates the LSDB which is then accessed by the parser. Table 3.2 shows the LSDB generated for the sentence in (2).

(2) *Sotaycang-i mwucenki-lul swuliha-yess-ta.*
platoon-leader-Nom radio-Acc repair-Past-Decl
The platoon leader repaired the radio.

[1] An *eojeol* is the syntactic unit delimited by spacing characters in Korean.

⟨⟨INDEX⟩⟩*sotaycangi* ⟨⟨ENTRY⟩⟩*sotaycangi* ⟨⟨POS⟩⟩N ⟨⟨TREES⟩⟩αNP βNP βNP-V βNP-S ⟨⟨FEATURES⟩⟩@nom							
⟨⟨INDEX⟩⟩*mwucenkilul* ⟨⟨ENTRY⟩⟩*mwucenkilul* ⟨⟨POS⟩⟩N ⟨⟨TREES⟩⟩αNP βNP βNP-V βNP-S ⟨⟨FEATURES⟩⟩@acc							
⟨⟨INDEX⟩⟩*swulihayessta* ⟨⟨ENTRY⟩⟩*swulihayessta* ⟨⟨POS⟩⟩V ⟨⟨FAMILY⟩⟩Tnx0nx1V ⟨⟨FEATURES⟩⟩@past @cls-main @end-*ta*							

Table 3.2: LSDB generated from the sentence "*Sotaycang-i mwucenki-lul swuliha-yess-ta*"

3.1.2 Tree Database

The **Tree Database** in the Korean XTAG system has the same format as the one in English XTAG system (The XTAG-Group (1998)).

The tree database contains the tree templates that are lexicalized in several steps as given in section 3.1.1. The lexical items are inserted into distinguished nodes in the tree template called the anchor nodes. The part of speech of each word in the sentence corresponds to the label of the anchor nodes in the trees. The tagset used in the Korean XTAG system is given in Table 3.3. The lexical categories N, V, A and D each project to phrasal categories NP, VP, AP and DP respectively. In addition, the symbol S is used for clause-level category.

part-of-speech	examples
N	sakwa ('apple'), pokose ('report')
V	nolayhata ('sing'), kongkuphata ('supply')
A	kanunghata ('possible'), philyohata ('necessary')
AD	cacwu ('often'), ppalli ('quickly')
D	motun ('all'), say ('new')
CONJ	ttonun ('or'), mich ('and')
IJ	aikwu, eme

Table 3.3: Part of speech tags and examples

3.1.3 Tree Grafting

Once a particular set of lexicalized trees for the sentence have been selected, the Korean XTAG uses the LTAG parser LEM (Sarkar (2000)) to find all derivations. The derived and derivation trees can be viewed using an X-interface.

For example, a derived and derivation trees for the example sentence (2) are shown in Figure 3.2. A derived tree is similar to a phrase structure tree and the derivation tree represents the derivation history of the parse tree (The XTAG-Group (1998)). The nodes of the derivation tree are the tree names anchored by the lexical items.

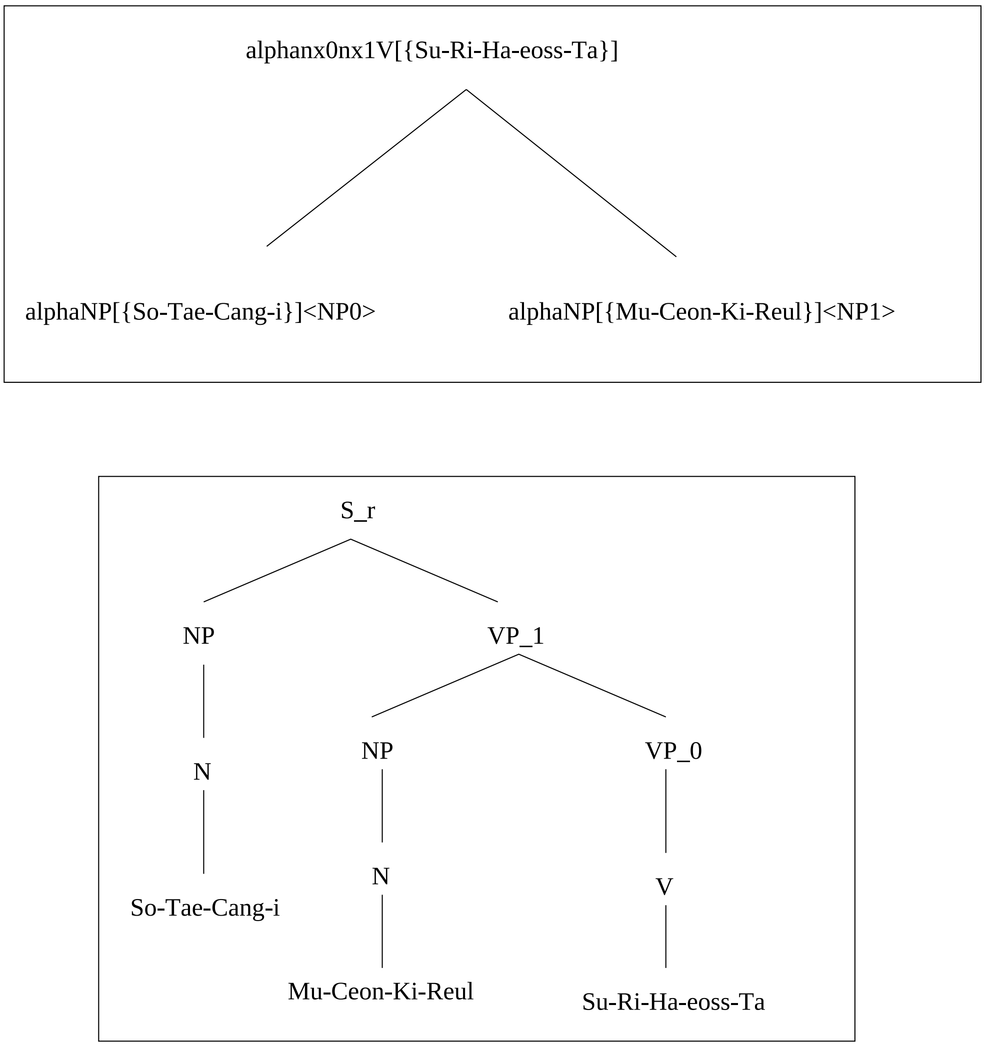

Figure 3.2: Derivation and derived trees for the example sentence (2)

Chapter 4

Preliminaries

4.1 Handling Inflectional Morphology

Korean is an agglutinative language with a very productive inflectional system. The part of speech of an inflected lexical item is the same as the part of the speech of its stem. Inflections include postpositions on nouns; tense morphemes and endings that indicate sentence types on verbs and adjectives; among others. Furthermore, these inflections can combine with each other to form compound inflections.

(3) Noun

a. *hakkyo-ka*
school-Nom

b. *hakkyo-eyse-ka*
school-from-Nom

c. *hakkyo-eyse–man*
school-from-only

d. *hakkyo-eyse-man-un*
school-from-only-Topic

(4) Verb

a. *ka-ss-ta*
go-Past-Decl

b. *ka-si-ess–ta*
go-Honor-Past-Decl

c. *ka-ki-ka*
go-Nominalizer-Nom

The ability to combine inflections to form compound inflections implies that a word in Korean can have a very large number of morphological variants.

In our current Korean grammar, the inflectional morphology instantiated on the lexical item is represented as features on the tree node. For instance, a noun with a nominative case marker is associated with the feature ⟨case:nom⟩ in the lexicon and when this lexical item is anchored by an NP tree, the feature ⟨case:nom⟩ is passed up to the NP node. Examples of NP trees that anchor the noun *cihwikwan* (‘commander’) inflected with different case markers are given in Figure 4.1. It can be seen that the case features are passed from N node up to NP node.

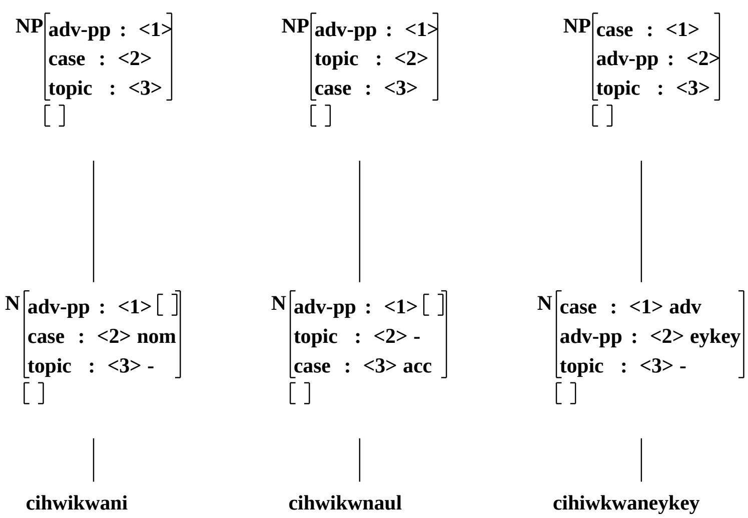

Figure 4.1: NP trees with different case features

In Korean, combining inflections is a highly productive process with some restrictions. For example, nominative, accusative and genitive CASE postpositions occur in a complementary distribution, but ADVERBIAL postpositions (which correspond to English prepositions) such as -*ey* (‘at’), *eykey* (‘from’), -*kkaci* (‘to’), etc. can be followed by nominative case or genitive case. Cases and adverbial postpositions are assumed to be assigned by the predicate of the sentence. Moreover, AUXILIARY postpositions which have semantic content such as -*man* (‘only’) and -*to* (‘even’) can combine with an adverbial postposition. The topic marker -*(n)un* can combine with an adverbial postposition and/or an auxiliary postposition but not the case postposition.

Moreover, predicates[1] in Korean are inflected with several morphemes. They carry CLAUSE-TYPE morphemes that indicate whether the clause is a main, coordinate, subordinate, relative clause, or a gerund. If a clause is a main clause, the verb carries a MODE morpheme that indicates whether the clause is declarative, imperative, interrogative, excla-

[1]In Korean, both verbs and adjectives play the role of a predicate in a sentence, both of which can be inflected.

mation, or propositive, etc. Clause-type morphemes and mode morphemes occur at the end of the verb. In addition, verbs also carry TENSE inflections right before the clause-type and mode morphemes. Further, all these inflections can be expressed in many different ways. All these verbal inflections are also represented as features on the tree node. Examples of S trees that anchor the verb *nolayha* ('sing') inflected with different verbal morphology are given in Figure 4.2.

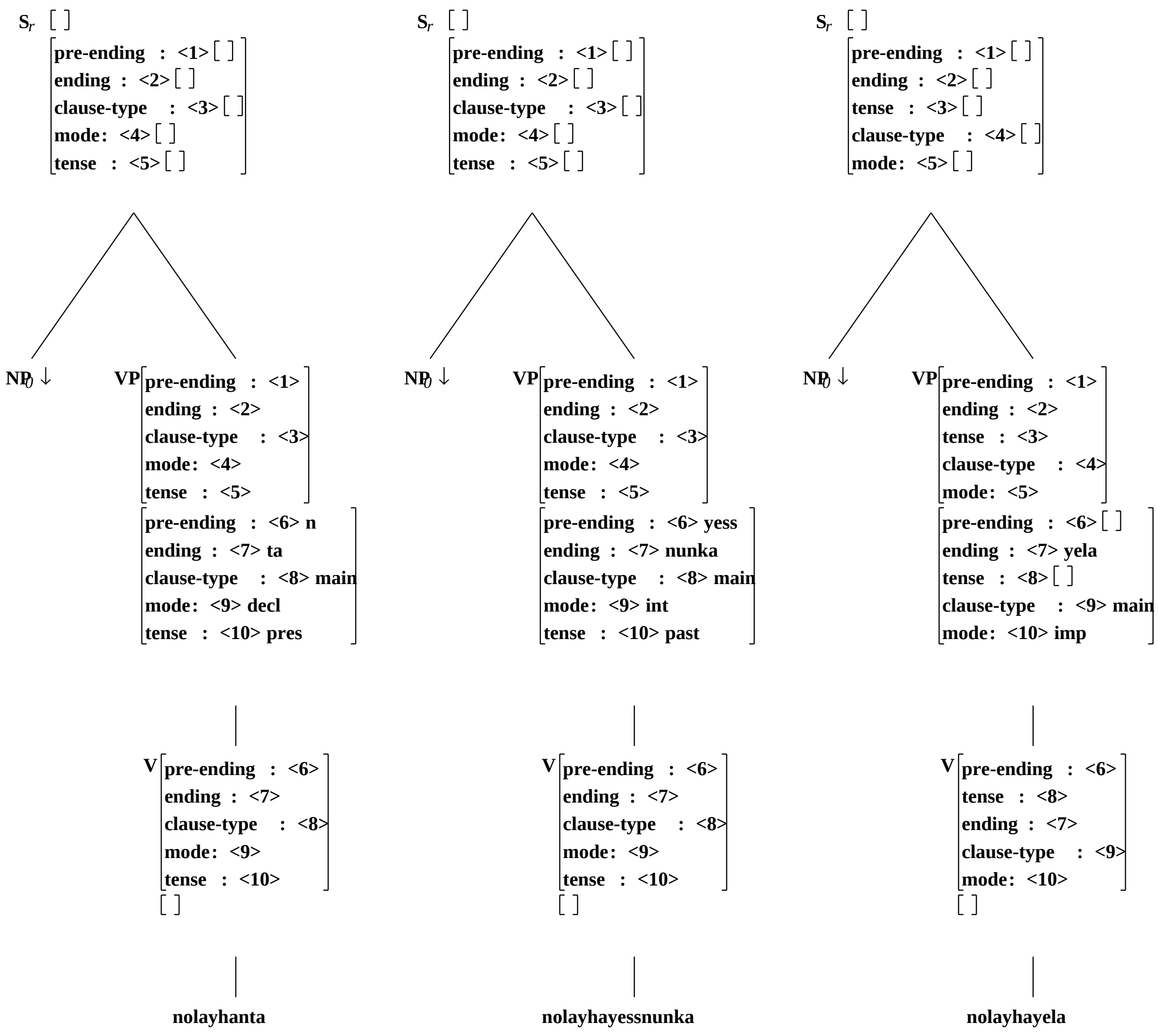

Figure 4.2: S trees that anchor verbs inflected with different verbal morphology

In order to handle all the possible ways of combining inflections, we imposed a hierarchy among various types of inflections and represented this by assigning different types of inflections to different feature attributes. Table 4.1 summarizes the list of inflectional feature attributes and the corresponding feature values currently being used by our grammar. Note that verbal features include ⟨ending⟩ which allows us to store the string values of mode and clause-type morphemes in the tree node for later semantic interpretation.

On nouns		
⟨**case**⟩	a case feature assigned by predicate	nom, acc, gen, adv
⟨**adv-pp**⟩	a feature assigned by predicate only if ⟨case:adv⟩, corresponds to English prepositions such as *to, from, in*	string values such as *ey, eyse, lo, wa, ya, kkaci, pwute, pota, lako, kathi, losse, chelem, eykeyse, eykeylo, mankum ...*
⟨**top**⟩	marks presence/absence of topic marker *(n)un*	+, -
⟨**aux-pp**⟩	adds specific meaning such as *only*, *also* etc.	string values such as *to, man*
On predicates		
⟨**clause-type**⟩	a feature that indicates the type of the clause that contains the predicate	main, coord, subord, adnom, nominal, aux-connect
⟨**mode**⟩	a feature on a predicate only if ⟨clause-type:main⟩	decl, imp, int, excl, propos
⟨**tense**⟩	encodes temporal interpretation	pres, past, future
⟨**ending**⟩	a feature marked for different ways of instantiating mode and the clausal type	string values such as *ta, nunka, ela, ki, nun, tako, ...*
⟨**pre-ending**⟩	a feature that marks for different ways of instantiating tense and honorific	string value such as *ess, syessess, keyss, ...*

Table 4.1: Features for Inflectional Morphology

4.2 Handling Derivational Morphology

4.2.1 Simple cases

The part of speech of a lexical item with a derivational morphology may differ from that of its stem. For instance, a lexical item that consists of a noun stem and *ha* is a verb as a whole. And so, such a lexical item anchors to a verb node that projects to a sentence.

(5) a. poko+ha → kongpwuha
report(N.)+ha → report(V.)

b. swuyeng+ha $\rightarrow$ swuyengha
swim(N.)+ha $\rightarrow$ swim(V.)

Further, a lexical item that consists of an adjective stem and the morpheme *-key* is an adverb as a whole. Such a lexical item anchors an adverb tree.

(6) a. nuli+key $\rightarrow$ nuli+key
slow+key $\rightarrow$ slowly

b. swip+key $\rightarrow$ swipkey
easy+key $\rightarrow$ easily

4.2.2 Verb + nominalizer

However, in some cases, determining the part of speech of a lexical item with derivational morphology is not so easy. For instance, although a lexical item that consists of a verb stem and a nominalization suffix *ki* behaves like a noun as a whole (e.g., it takes a case marker), it also behaves like a verb in that it can be modified by an adverb and it maintains the subcategorization frame of the stem verb. For instance, in (7a), *mek-ki* takes an accusative case marker and it takes an object NP *sakwa-lul*. In (7b), *o-ki* takes a nominative case marker and it is modified by an adverb *ilccik*.

(7) a. Chelswu-ka [e sakwa-lul mek-ki-lul] sileha-n-ta.
Chelswu-Nom apple-Acc eat-Nominalizer-Acc dislike-Pres-Decl
'Chelswu does not like to eat apples.'

b. [Chelswu-ka ilccik o-ki-ka] elyeps-ta.
Chelswu-Nom early arrive-Nominalizer-Nom difficult
'For Chelswu to arrive early is difficult.'

To such a lexical item, we assigned V as the part of speech, and let it anchor a verb node which projects to a sentence that further projects to a noun phrase. These trees represent gerunds, which will be discussed in more detail in section 6.6.

4.2.3 Noun + CEK

Determining the part of speech for words with the morpheme *-cek* (CEK-words) was also difficult. This morpheme productively attaches to Sino-Korean nouns.[2]

(8) a. sahoy-cek
society-CEK
'social'

[2] Korean words that have Chinese origin are called *Sino-Korean* words.

b. cengchi-cek
politics-CEK
'political'

Syntactically, a CEK-word behaves as an adjective in that it modifies a noun, and it can be modified by an adverb.

(9) a. sahoy-CEK inkan
social man

b. acwu sahoy-CEK inkan
very social man

However, morphologically, a CEK-word behaves similarly to a noun in that it can be used with a copular marker to function as a predicate.

(10) a. inkan-un sahoy-CEK i-ta.
man-Top social-be Cop-Decl
'Man is social.'

b. i kenmwul-un hakkyo i-ta.
this building-Top school Cop-Decl
'This building is a school.'

Other adjectives do not take a copular marker to function as a predicate.

(11) a. sangca-ka mwukep-ta.
box-Nom heavy-Decl
'The box is heavy.'

b. * sangca-ka mwukep-i-ta.
box-Nom heavy-Cop-Decl
'The box is heavy.'

Moreover, sentences that have CEK-words as predicates can be negated with the form used for negating NP copular sentences.

(12) a. inkan-un sahoycek-i anita.
man-Top social-Nom not
'Man is not social.'

b. i kenmwul-un hakkyo-ka anita.
this building-Top school-Nom not
'This building is not school.'

In Korean, a nominative case marker is instantiated by *-ka* after a vowel, and *-i* after a consonant. In (12b), *-ka* on *hakkyo* is a nominative case marker, *hakkyo-ka* is a noun and *anita* takes two NPs. Thus, *-i* on *sahoycek* in (12a) must also be a nominative case marker and *sahoycek* must be a noun.

Based on these facts, we concluded that the best way to capture the empirical facts of the CEK-words is to say that they are nouns. In order to capture the fact that CEK-words can appear with adverbs, we decided to allow a tree where nouns are modified by an adverb. The nouns that can occur in this tree is restricted to CEK-nouns, and the adverbs that can anchor to this tree is also restricted to adverbs such as *acwu* and *maywu*.

One problem with our approach as it stands is that sentences with CEK-words as predicates can also be negated with the form for negating sentences with adjective predicates (as well as verbs). And this form cannot negate NP copular sentences.

(13) a. Sangca-nun mwukep-ci anhta.
box-Top heavy-CI not-Decl
'The box is not heavy.'

b. inkan-un sahoycek i-ci anhta.
man-Top social Cop-CI not
'Man is not social.'

c. * John-un inkan i-ci anhta.
John-Top man Cop-CI not-Decl
'John is not a man.'

For this reason, Sohn (1994) has argued that CEK-words by themselves are like nouns, but CEK-words with copula marker are like adjectives. In our current grammar, we do not yet have a way of capturing this distinction.

Chapter 5

Tree Families: Representing Subcategorization Frames

Tree families group together elementary trees that have the same subcategorization frame. They consist of elementary trees that are anchored by verbs or adjectives. These verbs and adjectives subcategorize for the other elements that appear in the tree, forming a clausal or sentential structure. For instance, the verb *mekta* ('to eat') is a transitive verb that subcategorizes for a subject NP and an object NP. It selects a transitive verb tree family and anchors to an elementary tree that includes nodes for the NP object and the NP subject. A tree family includes elementary trees that represent simple declarative structures as well as other structures that are transformationally related to each other, i.e., relative clauses, gerunds, adverbial clauses.

In this section, we provide a brief description of each tree family, and show the corresponding declarative elementary tree. Discussions on transformationally related elementary trees in a given tree family will be given in Chapter 6. In addition to the tree families described in this section, the grammar contains tree families, which are are not entirely complete yet, for various light verb constructions and idiomatic expressions.

5.1 Intransitive Verb: Tnx0V

Description: This tree family is selected by verbs that do not require an object complement of any type. Adverbs and other NPs may adjoin on, but are not required for the sentences to be grammatical.

Examples: *palsayngha* ('occur'), *kamsoha* ('decrease')

(14) a. thamci hwaltong-i palsayngha-n-ta.
detection activity-Nom occur-Pres-Decl
'Detection activity is occurring.'

b. kongkupmwul-i kamsoha-yess-ta.
supply-Nom decrease-Past-Decl
'Supplies decreased.'

Declarative tree: See Figure 5.1[1]

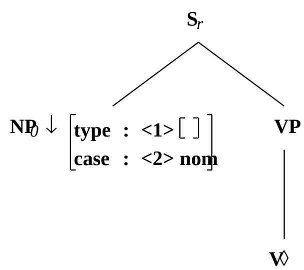

Figure 5.1: Declarative Tree: αnx0V

5.2 Transitive Verb: Tnx0nx1V

Description: This tree family is selected by verbs that require an NP subject and an NP complement marked with accusative case *-(l)ul*.

Examples: *yuciha* ('maintain'), *cisiha* ('order')

(15) a. salyengpwu-ka database-lul yuciha-n-ta.
headquarter-Nom database-Acc maintain-Pres-Decl
'The headquarter maintains the database.'

b. cihwikwan-i cengpohwaltong-ul cisiha-n-ta.
commander-Nom intelligence-activity-Acc order-Pres-Decl
'The commander orders intelligence-activity.'

Declarative tree: See Figure 5.2

[1] We leave out feature specifications for the verb and adjective in the example declarative trees in this section for the sake of simplicity. See Section 4.1 and Figure 4.2 for features on verbs and how they percolate up the tree. Moreover, ⟨type:⟩ features in the NP substitution nodes are place-holders for semantic features such as *animate, inanimate, physical object, etc.* We are currently in the process of coming up with a well-defined ontology for semantic features.

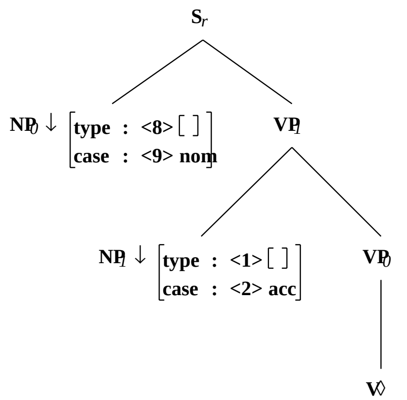

Figure 5.2: Declarative Tree: αnx0nx1V

5.3 Intransitive Verb with an NP marked with a Postposition: Tnx0nxp1V

Description: This tree family is selected by verbs that require an NP subject and an NP complement marked with an adverbial postposition.[2] Adverbial postpositions are comparable to prepositions in English.

Examples: *itongha* ('move'), *yulaytoy* ('originate')

(16) a. satan-i cenpang-ulo itongha-n-ta.
unit-Nom frontline-to move-Pres-Decl
'The unit is moving to the frontline.'

b. cengpo-ka yele chwulche-lopwute yulaytoy-n-ta.
intelligence-Nom many source-from originate-Pres-Decl
'Intelligence originates from many sources.'

Declarative tree: See Figure 5.3

5.4 Ditransitive Verb: Tnx0nxp1nx2V

Description: This tree family is selected by verbs that require an NP subject, an NP complement marked with an adverbial postposition, and an NP complement marked with accusative case.

[2]The letter *p* in *Tnx0nxp1V* stands for *postposition*.

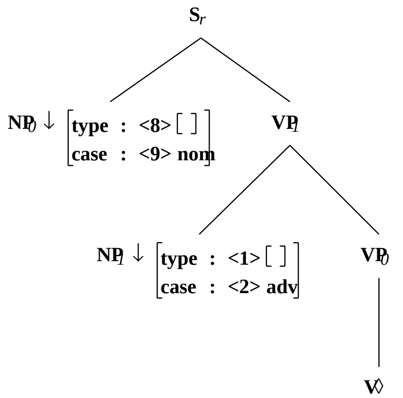

Figure 5.3: Declarative Tree: αnx0nxp1V

Examples: *cwu* (‘give’), *ponay* (‘send’)

(17) a. salyengpwu-ka 103 cenwiciwontaytay-eykey chwuka
headquarter-Nom 103 FSB-to additional
kongkupmwul-ul cwu-ess-ta.
supply-Acc give-Past-Decl
‘Headquarters gave 103rd FSB additional supplies.’

b. 9 satan-i salyengpwu-eykey pokose-lul ponay-ss-ta.
9 unit-Nom headquarter-to report-Acc send-Past-Decl
‘9th unit sent headquarters the report.’

Declarative tree: See Figure 5.4

5.5 Verb with Sentential Complement: Tnx0s1V

Description: This tree family is selected by verbs that require an NP subject and a sentential complement.

Examples: *sayngkakha* (‘think’), *mit* (‘believe’)

(18) a. salyengkwan-i cangkyo-uy poko-ka cenghwakha-tako
commander-Nom officer-Gen report-Nom accurate-Comp
sayngkakha-n-ta.
think-Pres-Decl
‘The commander thinks that the officer’s report is accurate.’

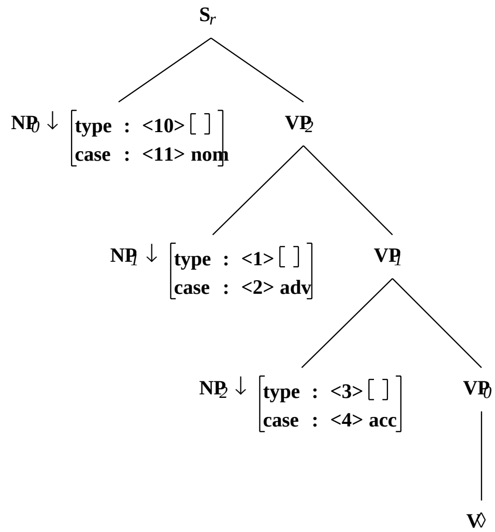

Figure 5.4: Declarative Tree: αnx0nxp1nx2V

b. salyengkwan-i cengpo-ka thuly-ess-tako
commander-Nom intelligence-Nom wrong-Past-Comp
mit-nun-ta.
believe-Pres-Decl
'The commander believes that the intelligence is wrong.'

Declarative tree: See Figure 5.5

5.6 Verb with Sentential Complement and an NP marked with a Postposition: Tnx0nxp1s2V

Description: This tree family is selected by verbs that require an NP subject, an NP with an adverbial postposition, and a sentential complement.

Examples: *myenglyengha* ('order'), *pokoha* ('report')

(19) a. salyengpwu-ka cenwiciwontaytay-eykey cengpohwaltong-ul
headquarter-Nom FSB-to intelligence-activity-Acc
sicakha-lako myenglyengha-yess-ta.
begin-Comp order-Past-Decl
'Headquarters ordered FSB to begin intelligence activity.'

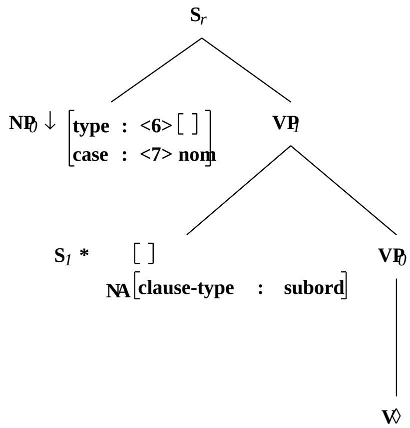

Figure 5.5: Declarative Tree: βnx0s1V

b. cangkyo-ka cihwikwan-eykey kongkupmwul-i tochakha-yess-tako
officer-Nom commander-to supply-Nom arrive-Past-Comp
pokoha-yess-ta.
report-Past-Decl
'The officer reported to the commander that the supplies arrived.'

Declarative tree: See Figure 5.6

5.7 Double Nominative Verb: Tnx0nxNOM1V

Description: This tree family is selected by verbs that require an NP subject, and an NP complement with nominative case marker.

Examples: *toy* ('become'), *iss* ('exist')

(20) a. chelswu-ka uysa-ka toy-ess-ta.
chelswu-Nom doctor-Nom become-Past-Decl
'Chelswu became a doctor.'

b. chelswu-ka cha-ka iss-ta.
chelswu-Nom car-Nom have-Decl
'Chelswu has a car.'

Declarative tree: see Figure 5.7

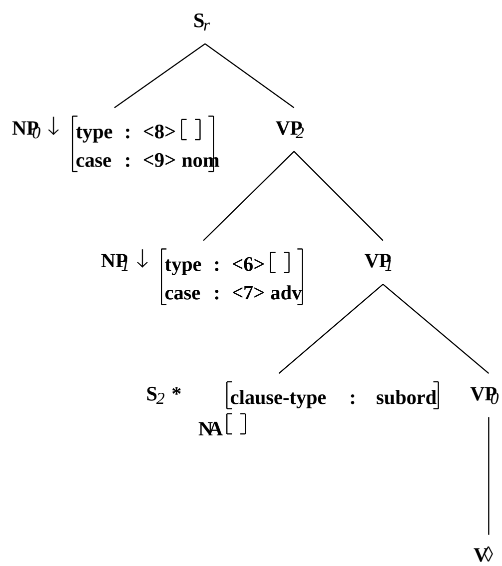

Figure 5.6: Declarative Tree: βnx0nxp1s2V

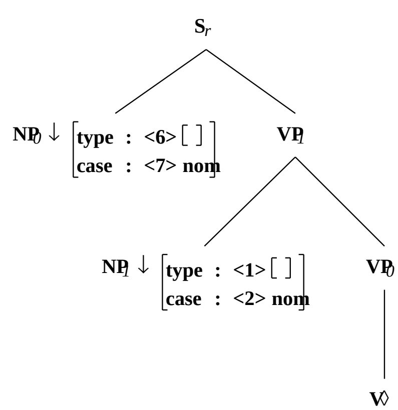

Figure 5.7: Declarative Tree: αnx0nxNOM1V

5.8 Copula: Tnx0nx1CO

Description: This tree family is selected by the copular *i*. It requires a subject NP and a complement NP. Currently, we do not make distinction between predicative copula sentences and equative copula sentences.

Examples: *uysa i* ('be a doctor'), *salyengkwan i* ('be a commander')

(21) a. chelswu-nun uysa i-ta.
chelswu-Top doctor Cop-Decl
'Chelswu is a doctor.'

b. ku namca-ka salyengkwan i-ta.
that man-Nom commander Cop-Decl
'That man is the commander.'

Declarative tree: see Figure 5.8

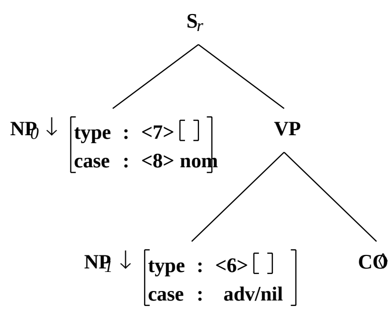

Figure 5.8: Declarative Tree: αnx0nx1CO

5.9 Intransitive Adjective: Tnx0A

Description: This tree family is selected by adjectives that require an NP subject.

Examples: *ku* ('be big'), *nolah* ('be yellow')

(22) a. ce sangca-nun acwu kuta.
that box-Top very big
'That box is very big.'

b. kkoch-i nolah-ta.
flower-Nom yellow-Decl
'The flower is yellow.'

Declarative tree: see Figure 5.9

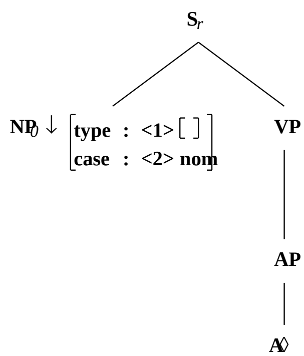

Figure 5.9: Declarative Tree: αnx0A

5.10 Intransitive Adjective with an NP marked with a Postposition: Tnx0nxp1A

Description: This tree family is selected by adjectives that require an NP subject and an NP complement marked with an adverbial postposition.

Examples: *talu* (‘be different’), *kath* (‘be same’)

(23) a. na-uy sayngkak-un ne-uy sayngkak-kwa talu-ta.
my idea-Top your idea-from different-Decl
‘My idea is different from yours.’

b. i mwuncey-nun ku mwuncey-wa kath-ta.
this problem-Top that problem-as same
‘This problem is the same as that problem.’

Declarative tree: see Figure 5.10

5.11 Double Nominative Adjective: Tnx0nxNOM1A

Description: This tree family is selected by adjectives that require an NP subject and an NP complement with a nominative case marker.

Examples: *philyoha* (‘be necessary’), *anita* (‘be not’)

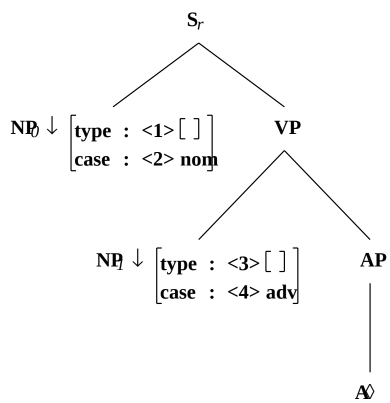

Figure 5.10: Declarative Tree: αnx0nxp1A

(24) a. salyengkwan-i pokose-ka philyoha-ta.
commander-Nom report-Nom necessary-Decl
'The commander needs the report.'

b. chelswu-nun papo-ka ani-ta.
chelswu-Top fool-nom not-Decl
'Chelswu is not a fool.'

Declarative tree: see Figure 5.11

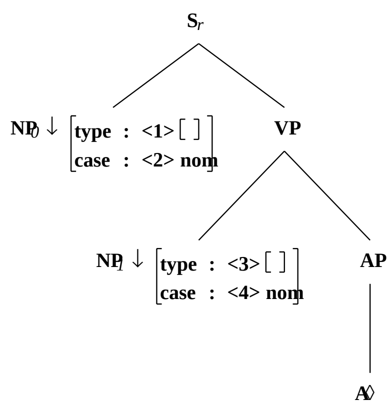

Figure 5.11: Declarative Tree: αnx0nxNOM1A

Chapter 6

Some Phenomena

6.1 Adverb Modification

Adverbs can occur at the beginning of the sentence, immediately before the object NP, or immediately before the verb.

(25) a. cacwu Chelswu-ka sakwa-lul mek-nun-ta.
frequently Chelswu-Nom apple-Acc eat-Pres-Decl
'Chelswu eats apples frequently.'

b. Chelswu-ka cacwu sakwa-lul mek-nun-ta.

c. Chelswu-ka sakwa-lul cacwu mek-nun-ta.

In the Korean XTAG grammar, we have two auxiliary trees to which adverbs can anchor to. One tree adjoins to S and the other tree adjoins to VP.

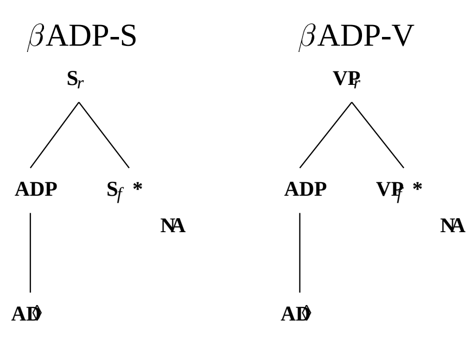

Figure 6.1: Auxiliary trees for adverbs

Furthermore, elementary trees that represent transitive sentences and ditransitive sentences have VP shells. That is, for transitive sentences, the verb projects VP, and then another VP projects to host an internal argument NP, and for ditransitive sentences, the verb projects VP, and then two VPs are projected on top of each other to host the two internal arguments respectively. We repeat in Figure 6.2 the transitive and ditransitive declarative sentence trees given in Figures 5.2 and 5.4.

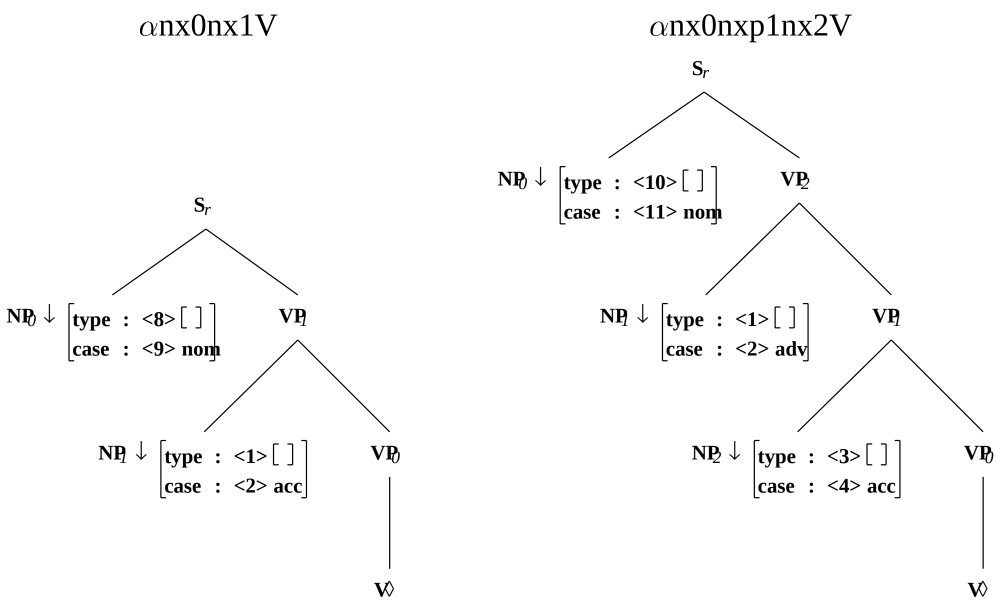

Figure 6.2: Declarative trees for transitive and ditransitive sentences

Given the two auxiliary trees for adverb modification and elementary trees with VP shells, we can capture all three adverb placements attested in (25). In (25a), the adverb *cacwu* anchors to βADP-S and adjoins onto S. In (25b), the adverb anchors to βADP-V and adjoins to VP_1 node in the transitive sentence tree. In (25c), the adverb again anchors to βADP-V and adjoins to VP_0 in the transitive tree. The derived trees for the sentences in (25) are given in Figure 6.3.

6.2 Empty Arguments

Korean freely allows empty subjects and/or empty objects.

(26) a. ϵ pokose-lul il-ess-ta.
ϵ report-Acc read-Past-Decl
'(I/you/he/we/they) read the report.'

b. salyengkwan-i ϵ il-ess-ta.
commander-Nom ϵ read-Past-Decl
'commander-Nom read (it/them).'

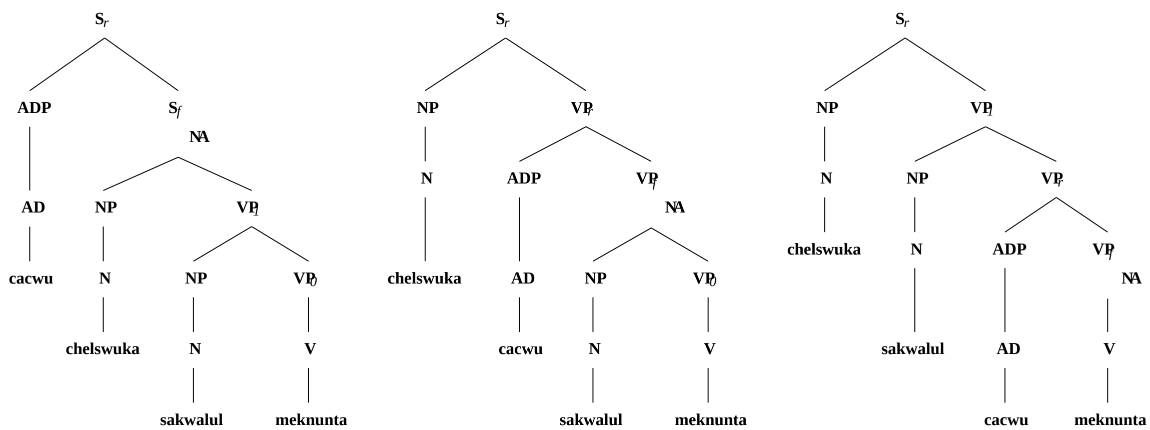

Figure 6.3: Derived trees for sentences with adverbs

c. ϵ ϵ il-ess-ta.
   ϵ ϵ read-Past-Decl
   ‘(I/you/he/we/they) read (it/them).’

To represent empty arguments in the Korean XTAG grammar, for each tree family, in addition to the declarative tree that has substitution NP nodes for each argument, we have included elementary trees whose argument NPs are associated with ϵ. For instance, the transitive tree family includes the elementary trees in Figure (6.4) to represent all the possible combinations of empty subject and/or empty object.

6.3 Relative Clauses

Relative clauses are NP modifiers that contain a gap. For instance, in (27), the head noun *silmwutanul* (‘administrative-team’) is modified by the relative clause with a subject gap *kak ceytayey haytanghanun* (‘which belongs to each unit’).

(27) salyengkwan-i [$_{NP}$ [$_S$ ϵ kak ceytay-ey haytangha-nun]
commander-Nom each unit-to belong-Adnom administrative-team-Acc
silmwutan-ul] kwusengha-n-ta.
compose-Pres-Decl
‘The commander composes administrative-team which belongs to each unit.’

Moreover, the main verb (or adjective) in the relative clause is inflected with a morpheme that indicates that the clause is modifying a noun. We call this morpheme *adnominal marker* (*Adnom* for short).

Relative clauses are represented in the Korean XTAG grammar by auxiliary trees that adjoin to NPs. These trees are anchored by the verb (or adjective) in the clause and appear in

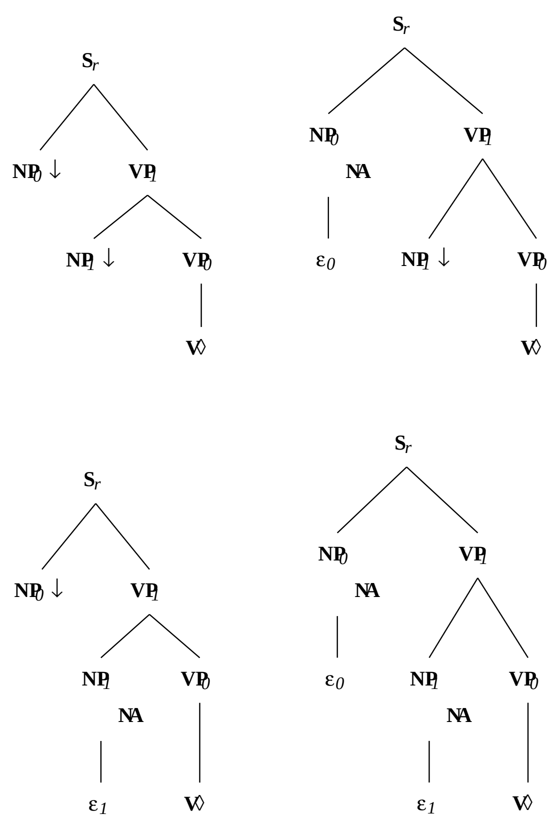

Figure 6.4: Representing empty arguments

the appropriate tree families for the various subcategorization frames. Within a tree family there will be groups of relative clause trees based on the declarative trees corresponding to each possible argument that can be the gap. Within each of these groups, there is a separate relative clause tree corresponding to all possible combinations of empty arguments. Some examples of auxiliary trees that represent relative clauses for the tree family Tnx0nxp1V are given in Figure 6.5.

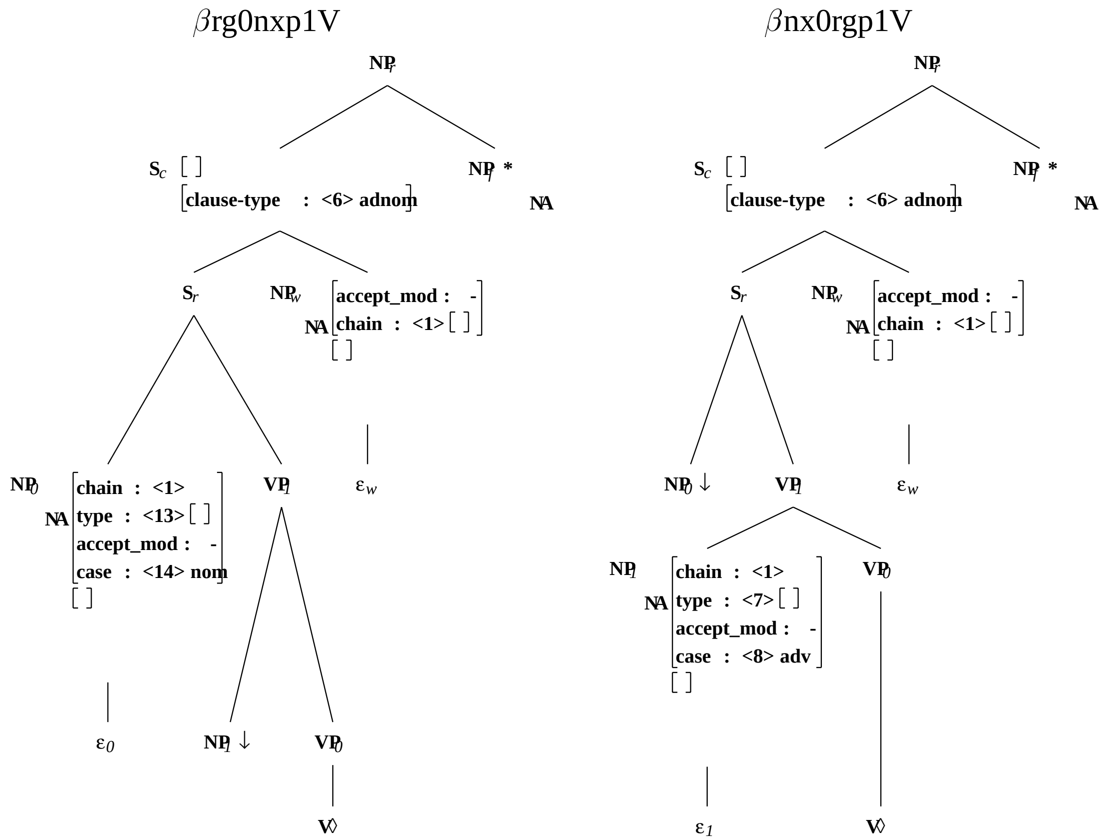

Figure 6.5: Auxiliary trees that anchor relative clauses

Although Korean does not have any overt relative pronoun as in English, we found it useful for semantic interpretation reasons to posit an empty relative pronoun operator that is syntactically related with the gap in the relative clause. This empty operator is represented in the relative clause tree as NP_w. The syntactic relation could be instantiated by movement of the empty operator from the gap in the relative clause to a higher position in the clause or by binding of the gap in the relative clause by the empty operator which is base generated higher in the clause. It does not matter for us which approach is adopted. We simply represent their syntactic relationship by coindexing the ⟨chain:⟩ features of the NP node that hosts the gap and the NP_w node that hosts the empty operator. There is no syntactic relationship between the head NP and the gap in the relative clause. The relationship between the relative clause and the head NP is treated as a semantic relationship (i.e., predicate modification) which will be provided by any reasonable compositional theory. Moreover, in order to restrict the main verbs in relative clause trees to those that are inflected with adnominal markers, we have specified S_c node with the feature ⟨mode:adnom⟩.

The derived tree for the sentence in (27) is given in Figure 6.6.

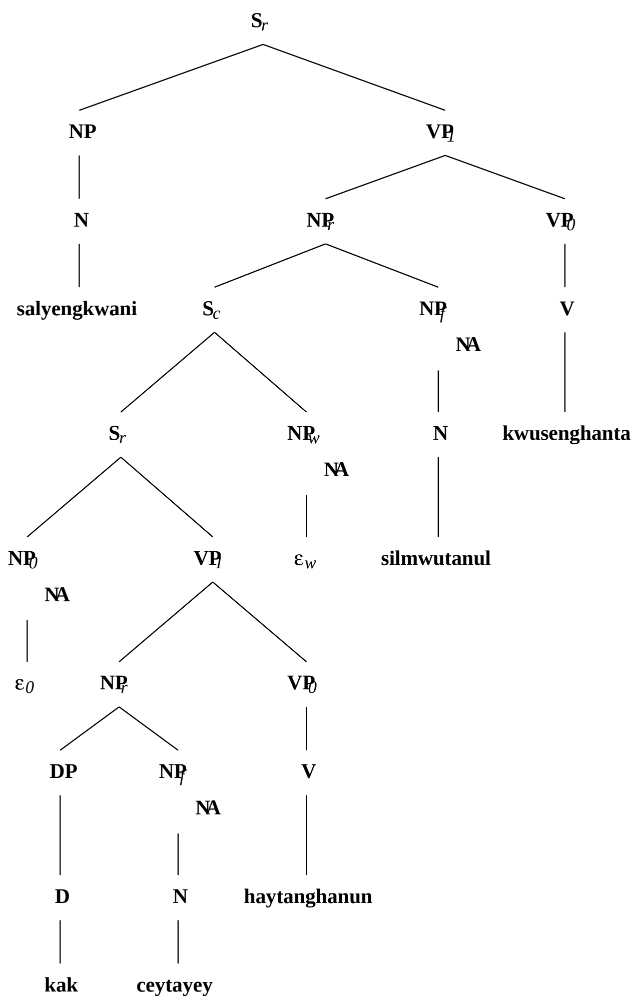

Figure 6.6: Derived tree for a sentence containing a relative clause

Adjunct relative clauses whose head nouns refer to place, time, reason, and manner are yet to be implemented.

6.4 Complex Noun Phrase

A complex noun phrase is similar to a noun phrase with a relative clause in that it is composed of a head noun and a clause that modifies the head noun and the main verb in the clause is inflected with an adnominal marker. But it differs from the noun phrase with a relative clause in that it does not have a gap in the clause that modifies the head noun.

(28)	[$_{NP}$ [$_S$ pwuntay-ka	salyengkwan-ul	pohoha-l]	philyo-ka]	epsta.
	unit-Nom	commander-Acc	protect-Adnom	necessity-Nom	non-existent

‘The necessity for the unit to protect the commander does not exist.’

The lexical item that can be a head noun of a complex NP is quite restricted. They include words such as *kes* (‘fact’), *philyo* (‘necessity’), *sasil* (‘fact’), *kyengwu* (‘situation’), etc. Given this, complex NPs are represented in the Korean grammar by an initial tree that is anchored by the head noun and that has a substitution node S for the clause that modifies the head noun. This tree is given in Figure 6.7. We have specified S$\downarrow$ with the feature $\langle$clause-type:adnom$\rangle$ to restrict the clauses that can substitute into this node to those whose main verbs are inflected with adnominal marker. The derived tree for the sentence in (28) is given in Figure 6.8.

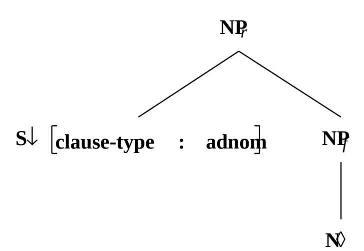

Figure 6.7: Initial tree that anchors complex NP: αSNP

Currently, in the Korean XTAG grammar, adjunct relative clauses whose head nouns refer to place, time, reason, and manner are treated as gapless relative clauses and they essentially get the same structural analysis as complex NPs.

6.5 Auxiliary Verbs

The auxiliary verb of a sentence carries tense information (if there is tense information) and sentence ending marker such as declarative marker, interrogative marker, imperative marker, etc. And the lexical verb (or adjective) carries an inflection (which we call *aux-connect*) that indicates that it is being followed by an auxiliary verb and it provides the predicate-argument structure information. An example sentence with an auxiliary verb is given in (29).

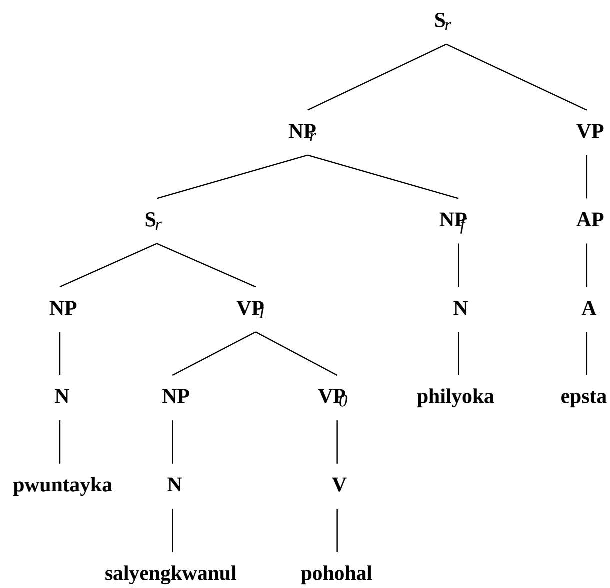

Figure 6.8: Derived tree for a sentence containing a complex NP

(29)	Chelswu-ka	sakwa-lul	mek-ko	issta.
	Chelswu-Nom	apple-Acc	eat-aux-connect	be-Pres-Decl
	‘Chelswu is eating an apple.’			

In (29), the auxiliary verb *issta* carries the present tense information and the declarative ending marker. And the information that the sentence is a transitive predicate argument structure comes from the lexical verb *mek* which is inflected with an aux-connect morphology *-ko*.

In the Korean XTAG grammar, the auxiliary tree called βVP anchors auxiliary verbs. This tree is given in Figure 6.9. We have specified VP_f* with ⟨clause-type:aux-connect⟩ to restrict adjunction of βVP tree to lexical verbs (or adjectives) that are inflected with the relevant morphology.

Given a transitive declarative sentence with an auxiliary verb, the lexical verb anchors to the declarative elementary tree from a transitive tree family, and the auxiliary verb anchors to βVP, which in turn adjoins to the topmost VP projection of the transitive declarative tree. The features ⟨tense:⟩, ⟨mode:⟩ and ⟨clause-type:⟩ in the root S node are inherited from the auxiliary verb. We block the tense, mode and clause-type features of the lexical verb from percolating up to the S node by disconnecting the top and bottom feature equations for these features at the topmost VP projection. We have also forced the auxiliary verb to assign nominative case to the subject NP, under the assumption that a tensed verb assigns nominative case. The declarative transitive tree with all the feature specifications are given in Figure 6.10. The derived tree for the sentence in (29) is given in Figure 6.11.

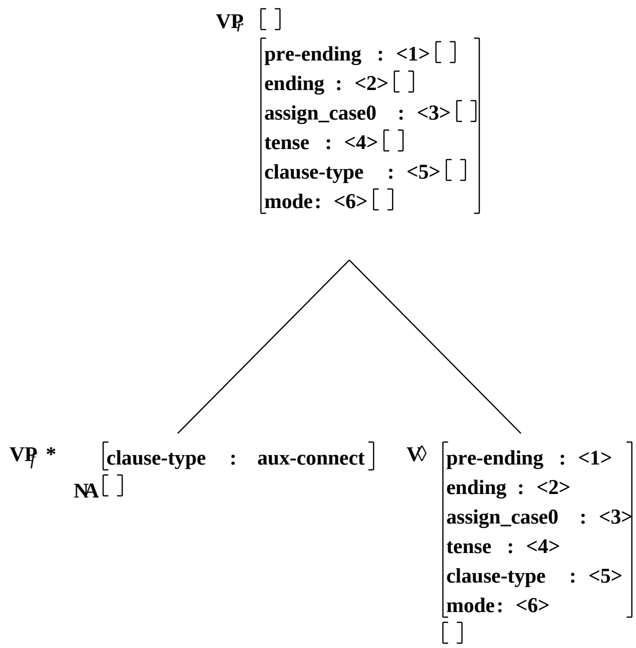

Figure 6.9: Auxiliary verb tree

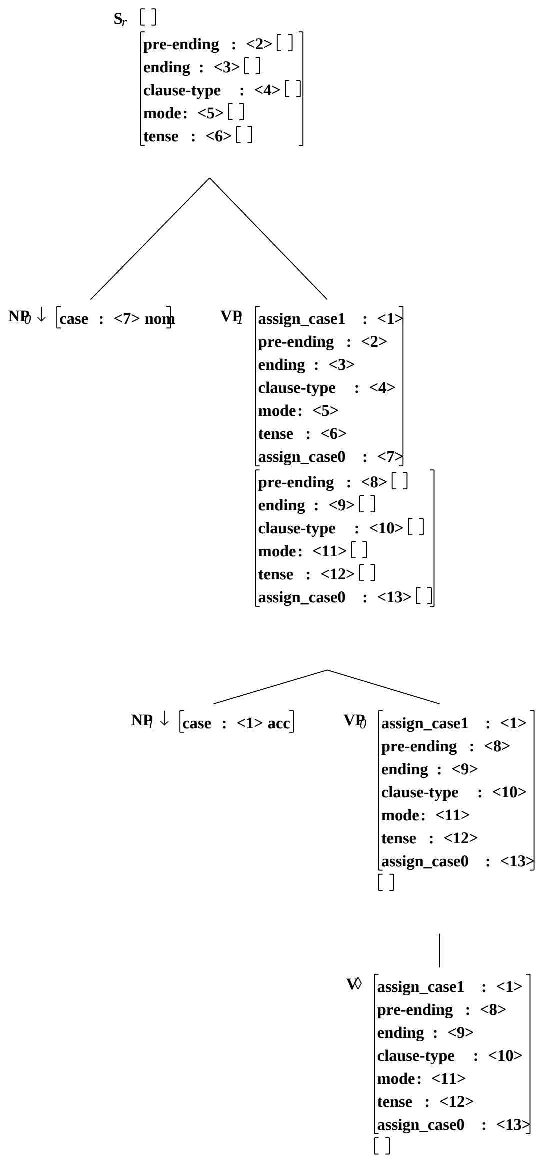

Figure 6.10: Declarative transitive tree with features

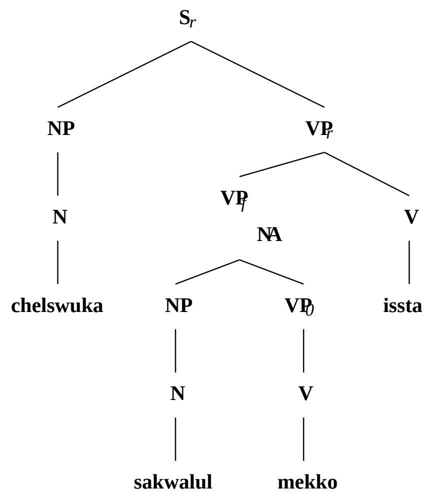

Figure 6.11: Derived tree for a transitive sentence with an auxiliary verb

6.6 Gerunds

We use the term *gerunds* for clauses whose main verbs (or adjectives) are inflected with a nominalizer marker *ki* or *um*. Gerunds have a double nature: Seen from the top, they behave as noun phrases in that the verb (or adjective) can be inflected with a case marker; but seen from the bottom, they behave just like any other sentential clause in that the predicate-argument structure for the main verb (or adjective) in the clause is preserved. Example sentences with gerunds are given in (7), repeated below as (30).

(30) a. Chelswu-ka [e sakwa-lul mek-ki-lul] sileha-n-ta.
Chelswu-Nom apple-Acc eat-Nominalizer-Acc dislike-Pres-Decl
'Chelswu does not like to eat apples.'

b. [Chelswu-ka ilccik o-ki-ka] elyeps-ta.
Chelswu-Nom early arrive-Nominalizer-Nom difficult
'For Chelswu to arrive early is difficult.'

Gerunds are represented in the Korean XTAG grammar by initial trees which are anchored by the verb (or adjective) in the clause and they appear in the appropriate tree families for the various subcategorization frames. Within a tree family there is a group of gerund trees based on the declarative trees and these trees represent all possible instantiations of empty arguments. This captures the gerund's clausal properties. In order to restrict the main verb in the gerund clause to the ones inflected with nominalizer *ki* and *um*, we specified S_r node with ⟨clause-type:nominal⟩ and ⟨ending:ki/um⟩. Further, as shown in Figure 6.12, the gerund tree project up to NP. The information on the case marker on the gerund verb (or adjective) is passed up and is retained as a case feature in the NP node.

This way, we capture the nominal properties of gerunds. The gerund NP can substitute into an appropriate NP substitution node in elementary trees. That is, a gerund whose main verb (or adjective) is marked with accusative case marker can substitute into the object NP substitution node in an elementary tree for transitive sentences. And a gerund whose main verb (or adjective) is marked with nominative case marker can substitute into the subject NP substitution node in any elementary trees for declarative sentences. A gerund tree with the feature structures in transitive verb tree family is given in Figure 6.12.

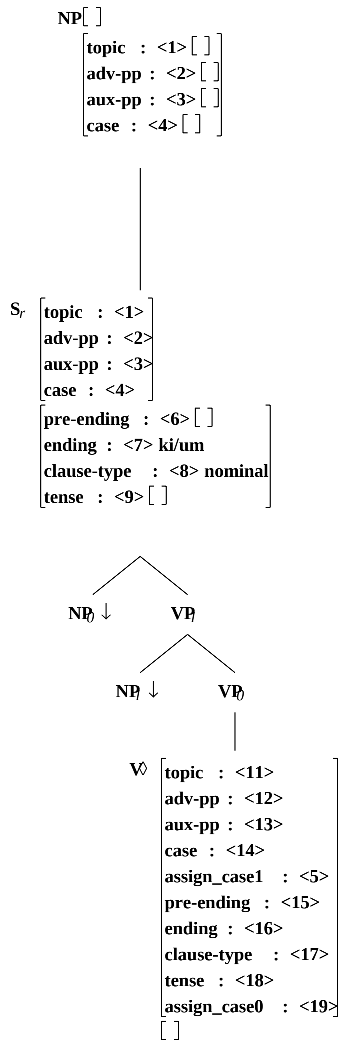

Figure 6.12: Transitive gerund tree with features

The four gerund trees in transitive verb tree family are given in Figure 6.13. Each tree represents different instantiation of empty arguments. The derived tree for the sentence in (30b) is given in Figure 6.14.

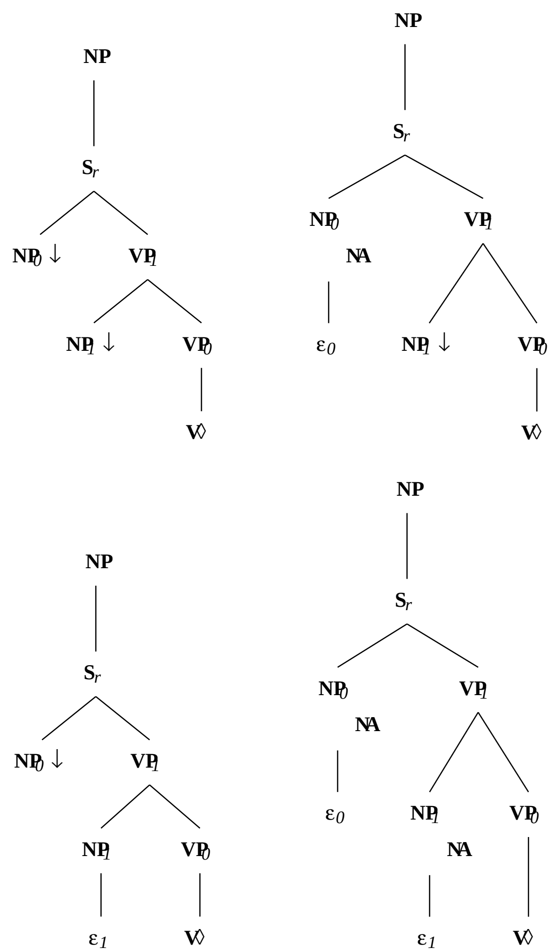

Figure 6.13: Gerund trees for transitive tree family

6.7 Adjunct Clauses

Clauses that modify a sentence is called *adjunct clauses*. An example of an adjunct clause is the *if*-clause of a conditional sentence, as in (31).

(31)	mwusenki-ka	kocangna-ss-tamyen	thongsin-i	pwulkanungha-ta.
	radio-Nom	broke-Past-if	communication-Nom	impossible-Decl
	'If the radio is broken, communication is impossible.'			

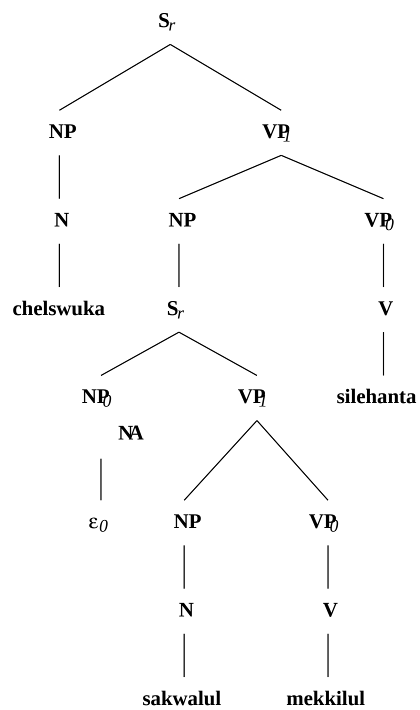

Figure 6.14: Derived tree for a sentence with a gerund

In the Korean XTAG grammar, adjunct clauses are represented as auxiliary trees that adjoin onto the S node. They are anchored by the verb (or adjective) in the clause and they appear in the appropriate tree families for the various verb (adjective) subcategorization. Within a tree family, there is a group of adjunct clause trees based on the declarative trees and these trees represent all possible instantiations of empty arguments. For instance, in intransitive verb tree family, there are two adjunct clause trees, as shown in Figure (6.15). The derived tree for the sentence in (31) is given in Figure 6.16.

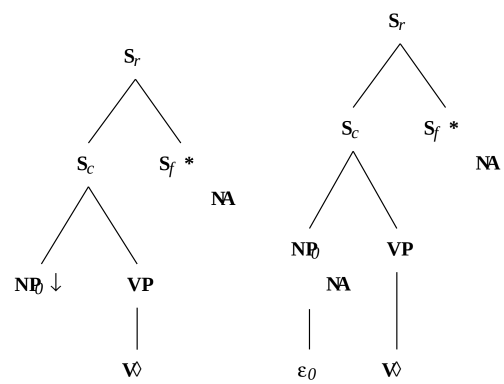

Figure 6.15: Adjunct clause trees for intransitive tree family

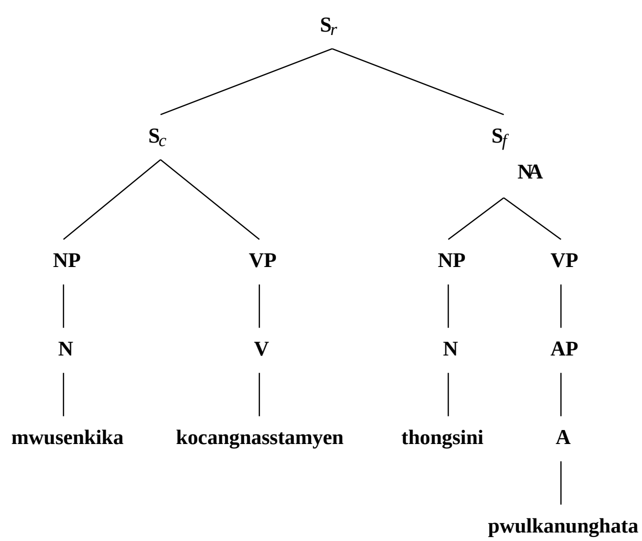

Figure 6.16: Derived tree for a sentence with an adjunct clause

Chapter 7

Conlusion

We have described the Korean XTAG system which comprises the morphological analyzer and the parser and how they interact with the Korean XTAG grammar. We then presented our general approach to inflectional and derivational morphology and the current status of our Korean XTAG grammar. Our immediate future plan is to expand the size of the ELEMENTARY SYNTACTIC DATABASE and the size of the elementary trees to handle more syntactic phenomena, such as scrambling, causative, resultative, light verb construction and coordination. We hope that our continued efforts can be applied to various natural language processing tasks such as parsing, machine translation, information extraction and generation.

Bibliography

Joshi, A. K. (1985) "Tree Adjoining Grammars: How much context Sensitivity is required to provide a reasonable structural description," in D. Dowty, I. Karttunen, and A. Zwicky, eds., *Natural Language Parsing*, Cambridge University Press, Cambridge, U.K., 206–250.

Joshi, A. K., L. Levy, and M. Takahashi (1975) "Tree Adjunct Grammars," *Journal of Computer and System Sciences*.

Kaplan, R., and J. Bresnan (1983) "Lexical-functional Grammar: A Formal System for Grammatical Representation," in J. Bresnan, ed., *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, Massachusetts.

Kroch, A. S., and A. K. Joshi (1987) "Analyzing Extraposition in a Tree Adjoining Grammar," in G. Huck and A. Ojeda, eds., *Discontinuous Constituents, Syntax and Semantics*, volume 20, Academic Press.

Sarkar, A. (2000) "A Probabilistic Head-corner Chart Parser for TAGs," ms. University of Pennsylvania.

Schabes, Y., A. Abeillé, and A. K. Joshi (1988) "Parsing strategies with 'lexicalized' grammars: Application to Tree Adjoining Grammars," in *Proceedings of the* 12^{th} *International Conference on Computational Linguistics (COLING'88)*, Budapest, Hungary.

Sohn, H. (1994) *Korean*, Descriptive Grammars, Routledge, New York.

The XTAG-Group (1998) "A Lexicalized Tree Adjoining Grammar for English," Technical Report IRCS 98-18, University of Pennsylvania.

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

Yoon, J., C.-H. Han, N. Kim, and M. Kim (2000) "Customizing the XTAG system for efficient grammar development for Korean," in *Proceedings of the 5th International Workshop on Tree Adjoining Grammars and Related Formalisms*, 221–226.

Yoon, J., C. Lee, S. Kim, and M. Song (1999) "Morphological Analyzer of Yonsei Univ., Morany: Morphological Analysis based on Large Lexical Database Extracted from a Corpus," In *Proceedings of Korean Language Information Processing* (In Korean).