Deep Processing of Korean and the Development of the Korean Resource Grammar*

Jong-Bok Kim Jaehyung Yang, Sanghoun Song,
Francis Bond

December 14, 2011

Abstract

The Korean Resource Grammar (KRG) is a broad-coverage, linguistically precise, and open-source grammar of Korean that has been developed since 2003. This paper reports how the KRG, aiming at deep processing of Korean, has gone through two main developing phases, in particular from a linguistically-oriented deep grammar to a wider-coverage with grammar customization. Deep linguistic processing has been obstacles in theoretical computational linguistics mainly because of the issues in the robustness and efficiency for the real life applications. The development of the KRG in these two phases, also participating in the international collaboration project DELPH-IN (Deep Language Processing in HPSG), has resulted in promising directions for the improvement of deep parsing as well as generation methods in a more systematic way so that it can be geared toward real-life NLP applications such as machine translation.

Key words: Korean Resource Grammar, deep processing, parsing, generation, HPSG, Minimal Recursion Semantics

1 Deep Linguistic Processing

For the past decade or more, linguistically oriented methods and statistical or machine learning approaches to NLP have often been perceived as incompatible or even competing paradigms. While shallow and probabilistic processing techniques (yielding simple parsed trees and meaning representations) have produced useful results in many classes of applications, they have not met the full range of needs

*Numerous people have contributed to the development of the KRG (Korean Resource Grammar). Among others, we thank the following people: Emily M. Bender, Chung Chan, Incheol Choi, Jae-Woong Choe, Sae-Youn Cho, Dan Flickinger, Yasunari Harada, Valia Kordoni, Eric Nichols, Stephan Oepen, Byong-Rae Ryu, Ivan Sag, Peter Sells, and Kiyotaka Uchimoto, among others. Our thanks also go to the anonymous reviewers of this journal. This work was supported by the National Research Foundation of Korea Grant funded by the Korean Government (NRF-2011-32A-A00066).

for NLP, particularly where precise interpretation is important, or where the variety of linguistic expression is large relative to the amount of training data available (cf. Uszkoreit 2002). On the other hand, due to advances in algorithm efficiency and rich deep processing resources, deep processing approaches to NLP (yielding rich and detailed syntactic and semantic representations) have only recently achieved broad enough grammatical coverage and sufficient processing efficiency to allow the use of precise linguistic grammars in real-world applications (cf. Baldwin et al. 2007).[1]

This need for deep language processing has also given motivations for the international collaboration DELPH-IN (Deep Linguistic Processing with HPSG). Aiming to provide an open-source collection for deep linguistic processing of human language within the HPSG framework, this joint collaboration tries to work together to promote the robustness of deep processing for natural language, focusing on areas such as (i) robustness, disambiguation, and specificity of deep processing with HPSG, (ii) the application of HPSG deep processing to information extraction, and (iii) multilingual grammar engineering (cf. Kordino and Neu 2005). As positive results, there have been several successful resource grammars developed, including the ERG (English Resource Grammar, Flickinger 2000), JACY (Siegel and Bender 2002 for Japanese), resource grammars for languages like German and other Indo-European languages.

The KRG (Korean Resource Grammar) has also been developed under this open-source NLP consortium since 2003, aiming to build a computational open-source grammar of Korean (Kim and Yang 2004b). The grammar has gone through two main development phrases. At the first phase, the KRG focused on linguistic phenomena, while in the second phase, the grammar has tried to adopt the grammar customization system using the LinGO Grammar Matrix (Bender et al. 2002) This direction not only improves the parsing efficiency but also adds generation capacity, prerequisite to its NLP applications (cf. Song et al. 2010). This paper aims to present some core properties of the deep computational grammar of Korean. In particular, we discuss how the grammar has been improved in two main phases, providing future directions for deep linguistic processing, which plays key roles in real-life applications.

2 Main Architecture for Deep Processing

The KRG has been constructed within the following open-source infrastructure, and is released at http://krg.khu.ac.kr under the MIT license[2]

[1] As noted here, we use the term 'deep processing' as a term for methods with full grammatical and semantic analyses while 'shallow processing' as a term referring to methods using a diminished level of linguistic precision. See Baldwin et al. (2007).

[2] The licence's philosophy is to give people the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies so long as the MIT copyright notice and this permission notice is included.

HPSG: The grammatical framework the KRG is couched upon is the framework of HPSG (Sag et al. 2003 and references therein) and most of the main grammatical analyses follow those set forth in the KPSG (Korean Phrase Structure Grammar, Kim 2004). HPSG is a constraint-based, non-derivational, lexicalist approach to grammatical theory that seeks to model human languages as systems of constraints on typed feature structures. In particular, the grammar adopts the mechanism of type hierarchy in which every linguistic sign is typed with appropriate constraints and hierarchically organized. The characteristic of such typed feature structure formalisms facilitates the extension of grammar in a systematic and efficient way, resulting in linguistically precise and theoretically motivated descriptions of languages including Korean. The grammar HPSG is thus well suited to the task of multilingual development of broad coverage grammars.

In addition,for semantic representations the KRG uses a flat semantic formalism Minimal Recursion Semantics (MRS) (Copestake et al. 2005). MRS offers an interface between syntax and semantics using feature structures. In MRS, the meaning of a given expression is represented as a flat bag of elementary predications (EPs), combining naturally with typed feature structures and allowing structures underspecified for scopal information.

LKB: The basic tool for grammar writing, testing, and processing the KRG is the LKB (Linguistic Knowledge Building) system developed by researchers at CSLI, Stanford University (Copestake 2002, Copestake et al. 2005). The LKB system is a grammar and lexicon development environment for use with constraint-based linguistic formalisms such as HPSG. The LKB system, freely available with open source (http://ling.stanford.edu) has a compiler for TFS (typed feature structure) grammars and parser and generator that maps from strings to meaning and vice versa. One strong advantage of this system is that a type-feature based grammar like HPSG can be easily encoded in the system as we will see in what follows.

The Grammar Matrix: Our system also uses the Grammar Matrix for the development of the KRG. The matrix system is an open source tool designed for the rapid development of precision-based grammars, within the HPSG and MRS formalism (Bender et al. 2002). This framework plays an important role in describing a HPSG/MRS-based grammar in a short-period of time, and improving it continuously.

Testing and Parsing Tools: The main software from the DELPH-IN collaboration is packaged with the so-called LOGON infrastructure. The system contains software packages, such as LKB for parsing and generation, PET for parsing (Callmeier 2000), and a management tool [incr_tsdb()] (Oepen 2001). The platform [incr_tsdb()] produces detailed diagnostic reports and complex multi-dimensional comparisons between alternative system. There are also several grammars included in the LOGON such as the ERG, JACY, resource grammars for Spanish, Greek, French, German, and so forth. Along with these, some pre-compiled versions of

preprocessing or experimental tools are packaged in the LOGON distribution.[3]

The architecture of the KRG incorporated in this LOGON system can thus be represented in the following figure.

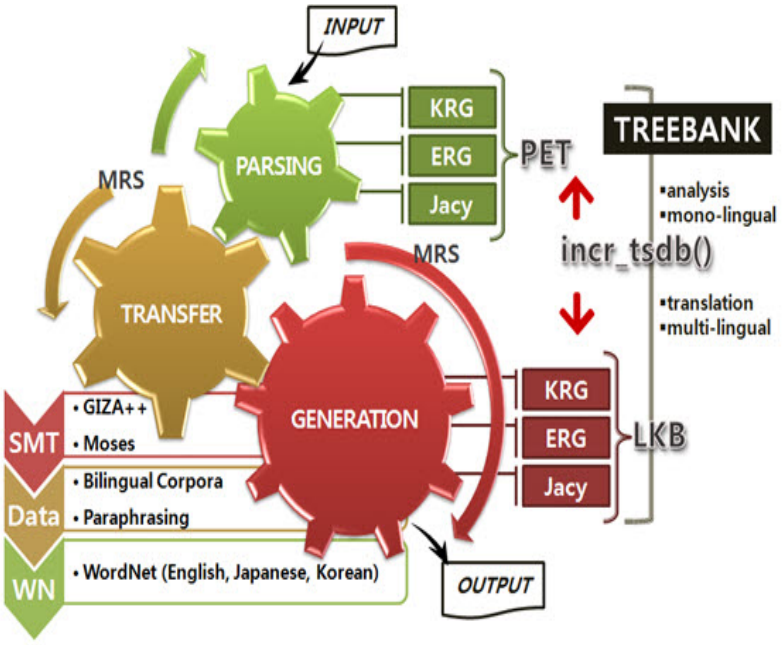


Figure 1: KRG in the LOGON system

3 Korean Resource Grammar: the First Phase

3.1 Basic Picture of the Grammar

Aiming at the development of a deep processing grammar for Korean within the sketched LOGON architecture, the KRG has been developed since 2003 and ever since gone through two main phrases (cf. Kim and Yang 2003, Kim and Yang 2004b). The main components of the KRG can be illustrated in Figure 2.

As shown in Figure 2, once a Korean source sentence is given as an input, the system as the preprocess level performs a morphological analysis and then performs its syntactic and semantic analyses. The syntactic and semantic parsing steps are built upon the basic grammatical components of the KRG such as grammar rules, inflection rules, lexical rules, type definitions, and lexicon. The type descriptions include *matrix.tdl* for general principles, *korean.tdl* for language particular rules, *types-lex.tdl* for lexical types, *types-ph.tdl* for phrasal types, etc. The characteristic of such typed feature structure formalisms facilitates the extension

[3] `wiki.delph-in.net/moin/LogonTop`

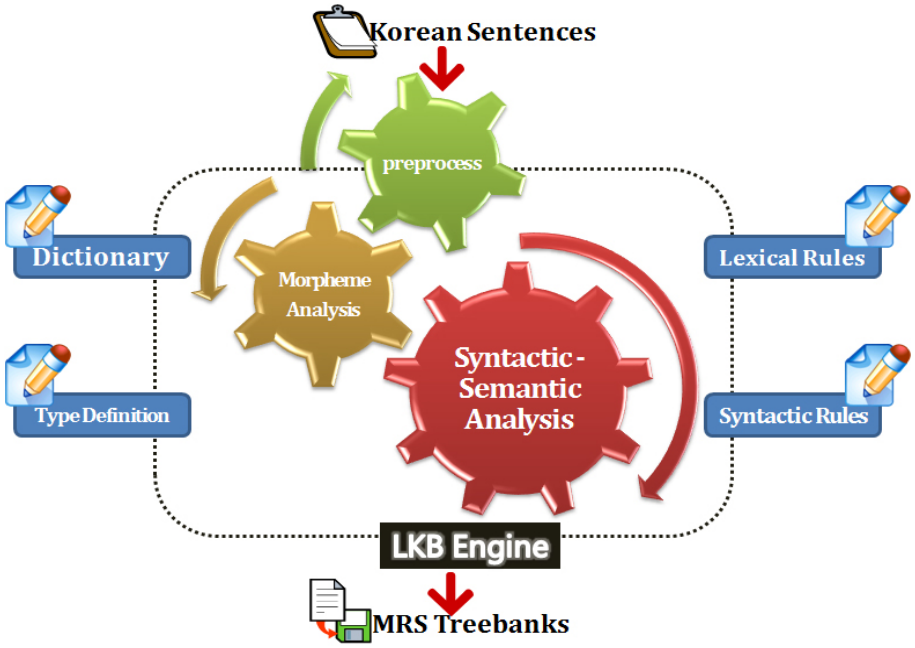
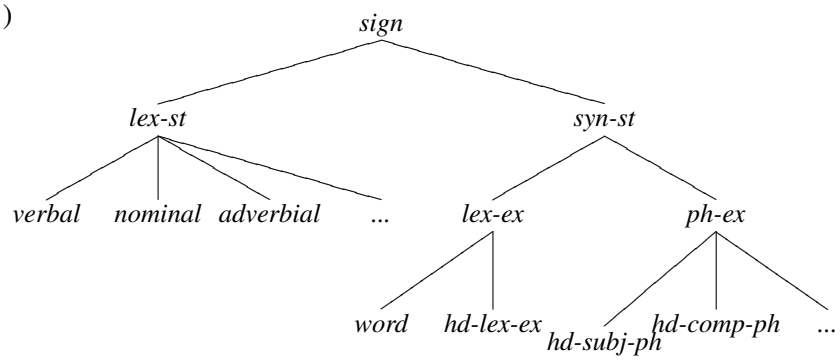


Figure 2: Main Components of the KRG

of grammar in a systematic and efficient way, resulting in linguistically precise and theoretically motivated descriptions of languages including Korean (see Copestake (2002)).

As noted, the grammar of the KRG basically follows the KPSG developed within the HPSG formalism (Sag et al. 2003, Kim and Sells 2008). The KRG starts with the following type hierarchy system in which every linguistic sign is typed with appropriate constraints and hierarchically organized (cf. Kim 2004):

(1)



The elements in *lex-st* type, forming the basic components of the lexicon, are built from lexical processes such as lexical rules and type definitions.[4] Parts of these elements will be realized as *word* to function as syntactic elements since those inflected as *word* can occur in syntax. The type *word* has subtypes such as *n-word*, *v-word*, and *adv-word*. As noted earlier, though all *nominal* elements can be

[4]For the detailed lexicon structure, refer to Kim and Yang (2004b).

projected into *n-word*, only *v-free* can be mapped (or pumped up) to *v-word* since not fully inflected stems like *mek-ess* 'eat-PST' cannot appear in syntax.[5] One important constraint that *word* in the KRG observes is the following Argument Realization Constraint (cf. Kim and Sells 2008):

(2) Argument Realization Constraint (ARC):

$$word \rightarrow \begin{bmatrix} \text{SYN} \mid \text{VAL} & \begin{bmatrix} \text{SUBJ} & \boxed{A} \\ \text{COMPS} & \boxed{B} \end{bmatrix} \\ \text{ARG-ST} & \boxed{A} \oplus \boxed{B} \end{bmatrix}$$

The constraint means the elements in the ARG-ST will be realized as SUBJ and COMPS in syntax (cf. Kim 2004, Kim and Sells 2008). This in turn means that ARG-ST is sensitive to the word-level only, while the syntactic features SUBJ and COMPS are relevant at syntax. For example, when the lexeme *ilk-* 'read' specified only with the ARG-ST information is fully inflected as *ilk-ess-ta* 'read-PAST-DECL', its two arguments will be realized as SUBJ and COMPS as represented in (3):[6]

(3)

$$\begin{bmatrix} \textit{v-tr} \\ \text{ORTH } \langle \text{ilk-} \rangle \\ \text{ARG-ST} \quad \langle \text{NP, NP} \rangle \end{bmatrix} \rightarrow \begin{bmatrix} \textit{v-word} \\ \text{ORTH } \langle \text{ilk-ess-ta} \rangle \\ \text{SYN} \begin{bmatrix} \text{HEAD} \begin{bmatrix} \text{POS } \textit{verb} \\ \text{V } + \\ \text{STATIVITY } - \end{bmatrix} \\ \text{VAL} \begin{bmatrix} \text{SUBJ } \langle \boxed{1} \rangle \\ \text{COMPS } \langle \boxed{2} \rangle \end{bmatrix} \end{bmatrix} \\ \text{ARG-ST } \langle \boxed{1}\text{NP, } \boxed{2}\text{NP} \rangle \end{bmatrix}$$

The morphological processing of adding the past tense *ess* and the mood marking *ta* adds the HEAD features (e.g., POS, VFORM, and STATIVITY).[7]

Once we have *word* elements in syntax, these elements will be combined with other syntactic elements. It is the type *ph-ex* that places restrictions on the combination of syntactic elements including *word*. This in turn means that the subtypes of *ph-ex* will tell us what kind of well-formed phrases is available in the language. The following are the grammar rules that license the subtypes of the type *ph-ex*:

(4) Grammar Rules:

[5]The type *v-free* thus means a fully-inflected verb that can appear in syntax. See Kim and Yang (2004b) for a detailed description of the verbal inflection system in Korean.

[6]In the lexicon, the ARG-ST value is not encoded since its information is predicted from the type *v-tr*.

[7]The space does not allow us to explicate the morphological system of the grammar. This system segments words into sequences of morphemes with POS tags and morphological information. In the KRG, it is in fact the type definitions on each morphological elements that play more crucial roles in forming morphological elements and incurring relevant grammatical information.

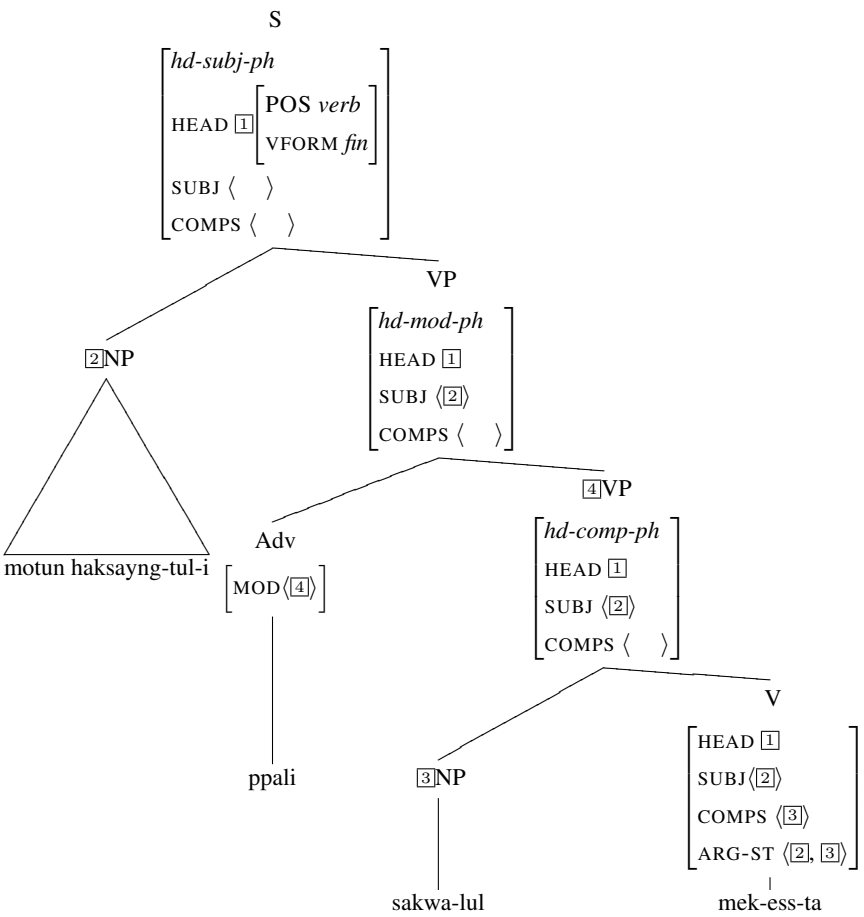
a. Head-Subject Rule:
XP[*hd-subj-ph*] → 1, **H**[SUBJ ⟨1⟩]

b. Head-Complement Rule:
XP[*hd-comp-ph*] → 1, **H**[COMPS ⟨..., 1, ...⟩]

c. Head-Modifier Rule:
XP[*hd-mod-ph*] → [MOD ⟨1⟩], 1**H**

These main grammar rules can license major phrases in the language. The Head-Subject Rule, generating a *hd-subj-ph*, allows a VP to combine with its subject. The Head-Complement Rule ensures a head to combine with one of its COMPS (COMPLEMENTS) elements, forming a *hd-comp-ph*. The Head-Modifier Rule allows a head to form a well-formed phrase with an adverbial element that modifies the head, resulting in *hd-mod-ph*. These grammar rules, interacting with general principles and lexical information, can generate well-formed Korean sentences like the following:

(5)



The sentence meaning 'All students ate apples quickly' is generated observing all the grammatical constraints declared in the KRG.[8] The verb *mek-ess-ta* 'eat-PST-DECL' selects two arguments, each of which is realized as SUBJ and COMPS according to the ARC. The head verb then combines with its COMPS *sakwa-lul* 'apple-ACC', forming a well-formed *hd-comp-ph* in accordance with the Head-Complement Rule. The resulting VP then is modified by an adverb *ppalli* 'quickly' according to the Head-MOD rule. This phrase eventually combines with the subject *motun haksayng-tul-i* 'all student-PL-NOM', forming a *hd-subj-ph* licensed by the Head-Subject Rule. Each phrasal combination also observes high-class constraints such as the Head-Feature Principle.

As illustrated here, the KRG generates binary syntactic structures, observing tightly interacting grammatical constraints. These constraints are lexical, phrasal, as well as constructional.

3.2 Semantics

In representing the semantics, the KRG adopts the MRS semantic representations designed to enable semantic composition using the unification of typed feature structures. The system allows us to produce for each phrase or sentence a description of the meaning representation in a systematic and compositional way. For example, the following is the meaning representation we obtain for the sentence in (5):

[8] Such a sentence can induce scope ambiguities which can be captured in the KRG, but is not discussed in detail here because of the space limit. See Kim (2006) for further discussion.

(6)
$$\begin{bmatrix}
\text{HOOK}\begin{bmatrix}\text{LTOP } h1\\ \text{INDEX } e1\end{bmatrix} & \\
\begin{bmatrix}\text{PRED } _prpstn_m_rel\\ \text{LBL } h1\\ \text{MARG } h9\end{bmatrix}, \begin{bmatrix}\text{PRED } _eat_v_rel\\ \text{LBL } h3\\ \text{ARG0 } e1\\ \text{ARG1 } i\\ \text{ARG2 } j\end{bmatrix}, & \\
\text{RELS}\left\langle \begin{bmatrix}\text{PRED } _all_q_rel\\ \text{LBL } h7\\ \text{ARG0 } i\\ \text{RESTR } h2\\ \text{BODY } h11\end{bmatrix} \begin{bmatrix}\text{PRED } _student_n_rel\\ \text{LBL } h10\\ \text{ARG0 } i\end{bmatrix}, \right\rangle & \\
\begin{bmatrix}\text{PRED } _exist_q_rel\\ \text{LBL } h9\\ \text{ARG0 } j\\ \text{RESTR } h4\\ \text{BODY } h12\end{bmatrix} \begin{bmatrix}\text{PRED } _apple_n_rel\\ \text{LBL } h8\\ \text{ARG0 } j\end{bmatrix} & \\
\text{HCONS}\left\langle \boxed{1}\begin{bmatrix}qeq\\ \text{HARG } h2\\ \text{LARG } h10\end{bmatrix}, \begin{bmatrix}qeq\\ \text{HARG } h4\\ \text{LARG } h8\end{bmatrix} \right\rangle &
\end{bmatrix}$$

As noted here, the three main semantic features for each sentence are RELS, HOOK, and HCONS. The feature RELS is a bag of elementary predications (EP) whose value is a *relation*. Each *relation* has at least three features LBL (label), PRED, and ARG0. The feature HOOK is a group of distinguished externally visible attributes of the atomic predications in RELS and includes the attributes LTOP, INDEX, and XARG. The final feature HCONS represents set of handle constraints on scoping. This value can be resolved in such a way that the quantifiers 'float in' wherever there is a 'space' left by a *qeq* (equality modulo quantifiers) constraint through the attributes LARG and HARG. When words with such semantic information combine with other syntactic (elements of *syn-st*) expressions, the meaning composition occurs in accordance with the Semantic Compositional Principles ensuring that in any well formed phrase structure, the mother's RELS value is the sum of the RELS values of the daughters (See Copestake et al. 2005 for details).

3.3 Main Phenomena Covered

3.3.1 Sentence Internal Scrambling

One welcoming consequence of the KRG with this basic picture is that it can capture sentence internal scrambling facts in a straightforward way, one of the most complicated facts in the SOV types of language. For example, the sentence in (7a) with five syntactic elements can induce 24 (4!) different scrambling possibilities. The language allows all the following word order possibilities as variations of (7a):

(7) a. mayil John-i haksayng-tul-eykey yenge-lul kaluchiessta
everyday John-TOP student-PL-DAT English-ACC taught
'John taught English to students everyday.

b. John-i yenge-lul mayil Tom-eykey kaluchi-ess-ta

c. John-i mayil Tom-eykey yenge-lul kaluchi-ess-ta.

d. John-i Tom-eykey mayil yenge-lul kaluchi-ess-ta.

e. John-i Tom-eykey yenge-lul mayil kaluchi-ess-ta.

f. ...

The grammar rules in (4) allow only binary structures. This is possible due to the fact that the Head-Complement Rule allows a head to combine with just one of the complements and that the Head-Subject Rule allows the subject to be combined at any stage.

It is needless to say that a more desirable grammar would be one that can capture all such scrambling possibilities within minimal processing load. The KRG grammar we sketched here requires no additional mechanism (i.e. movement operations) to allow such diverse word order possibilities. The grammar rules given in (4) can license and parse all these with no additional mechanisms such as complex movement processes.

3.3.2 Case Phenomena

As is well-known, Korean has a rich inflectional system for verbal and nominal elements. Nominal affixes, encoding various grammatical functions including case, are optional but are generated in tightly restricted ordering.[9] Once we have the right generation of nominal elements with case information, the next issue is how argument-selecting heads and grammar rules contribute their case information to nominal elements.[10] Let us consider intriguing case alternation phenomena:

(8) a. John-i nokcha-ka/*lul coh-ta
John-NOM green.tea-NOM/*ACC like-DECL
'John is fond of green tea.'

b. John-i nokcha-lul/*ka coh-a hanta
John-NOM green.tea-ACC/*NOM like-COMP do
'John likes green tea.'

The stative verb *coh-ta* assigns NOM to the object. However, when it combines with the auxiliary verb *hanta*, the object can get ACC.

The KRG adopts the lexeme-based lexicon where all the verbal lexemes will minimally have the following information:

[9] See Kim and Yang (2004b) and Kim (2004) for discussion of the language's nominal system.
[10] Most of the discussion here follows Kim and Yang (2005).

(9) $\begin{bmatrix} \textit{v-lxm} \\ \text{ORTH } \langle \text{ilk-} \rangle \\ \text{ARG-ST } \langle \text{NP}\left[\text{GCASE } \textit{vcase}\right], \text{NP}\left[\text{GCASE } \textit{vcase}\right] \rangle \end{bmatrix}$

This means that any element in the ARG-ST gets the value *vcase* as its GCASE (grammatical case) value: the *vcase* value can be either *nom* or *acc* in syntax. The elements in the ARG-ST will, in accordance with a realization constraint, be realized as SUBJ and COMPS in syntax as indicated in the following:

(10) $\begin{bmatrix} \langle \text{ilk-ess-ta 'read-PST-DECL'} \rangle \\ \text{SYN} \begin{bmatrix} \text{HEAD} \mid \text{POS } \textit{verb} \\ \text{VAL} \begin{bmatrix} \text{SUBJ } \langle \boxed{1} \rangle \\ \text{COMPS } \langle \boxed{2} \rangle \end{bmatrix} \end{bmatrix} \\ \text{ARG-ST } \langle \boxed{1}\text{NP}\left[\text{GCASE } \textit{vcase}\right], \boxed{2}\text{NP}\left[\text{GCASE } \textit{vcase}\right] \rangle \end{bmatrix}$

With this declarative verb *ilk-ess-ta* 'read-PST-DECL', the SUBJ element can be *nom* whereas the COMPS can be *acc*, but not the other grammatical case value as noted in (11):

(11) John-i/*ul chayk-ul/*i ilk-ess-ta
John-NOM/ACC book-ACC/NOM read-PAST-DECL
'John read a book.'

Then, the question is which part of the grammar makes sure the SUBJ is *nom* whereas COMPS is *acc*. The determination of case value in the VAL is not by a lexical process but imposed by syntactic rules. That is, the Head-Subject Rule will have the following case constraint:

(12) Head-Subject Rule:
$\left[\textit{hd-subj-ph}\right] \Rightarrow \boxed{1}\left[\text{CASE} \mid \text{GCASE } \textit{nom}\right], \mathbf{H}[\text{COMPS } \left\langle \boxed{1} \right\rangle]$

The rule simply says that when a head combines with the SUBJ, the SUBJ element is *nom*. As for the case value of a complement, it is a little bit more complicated since there are cases where the nonsubject argument gets NOM rather than ACC as in (8). In the language, nonagentive verbs like *coh-* 'be.fond.of' assign NOM to their complements. Reflecting this type of case assignment, we adopt the head feature AGT (AGENTIVITY) and ramify the Head-Complement Rule into two as the following:[11]

[11] The positive value of the AGT (AGENTIVITY), similar to STATIVITY, is assigned to the verbs that have an external argument whereas the negative value is assigned to those with no external argument.

(13) a. Head-Complement Rule A:

$$\left[\textit{hd-comp-ph}\right] \Rightarrow \boxed{1}\left[\text{CASE} \mid \text{GCASE } \textit{acc}\right], \mathbf{H}\left[\begin{array}{l}\text{HEAD} \mid \text{AGT} +\\ \text{COMPS}\left\langle \ldots, \boxed{1}, \ldots \right\rangle\end{array}\right]$$

b. Head-Complement Rule B:

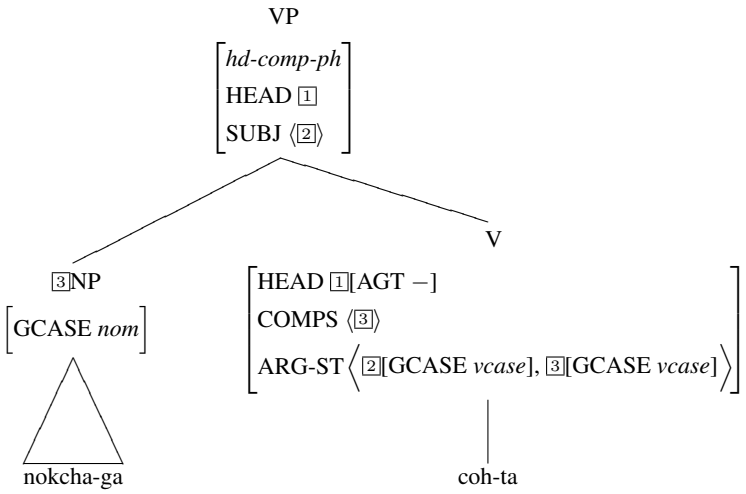
$$\left[\textit{hd-comp-ph}\right] \Rightarrow \boxed{1}\left[\text{CASE} \mid \text{GCASE } \textit{nom}\right], \mathbf{H}\left[\begin{array}{l}\text{HEAD} \mid \text{AGT} -\\ \text{COMPS}\left\langle \ldots, \boxed{1}, \ldots \right\rangle\end{array}\right]$$

Within this system, we then do not need to specify *nom* to the nonsubject complement of psych verbs, diverging from the traditional literature. Just like other verbs, the complement(s) of such psych verbs like *coh-ta* 'like-DECL' will bear just *vcase*, as a general constraint on verbal elements as represented in (14):

(14)
$$\left[\begin{array}{l}\text{HEAD}\left[\begin{array}{l}\text{POS } \textit{verb}\\ \text{AGT} -\end{array}\right]\\ \text{ARG-ST}\left\langle \text{NP}\left[\text{GCASE } \textit{vcase}\right], \text{NP}\left[\text{GCASE } \textit{vcase}\right]\right\rangle\end{array}\right]$$

This lexical information would then project the following partial structure for (8a):

(15)



As noted here, the verb *coh-ta* 'like' bears the head feature [AGT −]. This means that the complement of this verb will get NOM even though in the ARG-ST its case value is *vcase*. This is guaranteed by the Head-Complement Rule B in (13). However, for (8b), the auxiliary verb head *ha-ta* 'do' carries the feature [AGT +] and assigns *acc* feature to its COMPS element in accordance with the Head-Complement Rule A.

As illustrated here, the KRG can capture simple as well as intriguing case assignment phenomena observed in the language, based on the interaction between lexical properties and constraints in the grammar rules.

3.3.3 Complex Predicate Constructions

One of the most prevalent constructions in Korean is complex predicates that consist the argument structures of two separate predicates (V2-V1) being brought together somehow or other. Canonical complex predicate constructions include auxiliary and light verb constructions. The constructions are syntactically intriguing in that (a) it is V2 that theta-marks internal arguments and V1 thus has no influence on the number and types of arguments (b) the V2 takes an agentive subject but inherits its other arguments to the final predicate V1, and (c) V1 and V2 form a tight syntactic unit. For example, the two verbs must occur in a fixed order, always following immediately after a main verb as illustrated in (16b):

(16) a. John-un sakwa-lul mek-ko/*e siph-ess-ta.
John-TOP apple-ACC eat-COMP like-PAST-DECL
'John wanted to eat apples.'

b. *sakwa-lul mek-ko John-un siph-ess-ta.

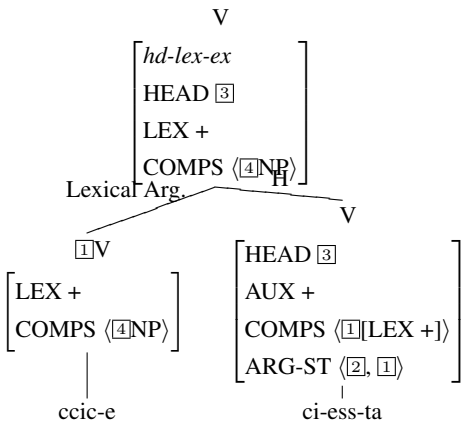
The syntactic passive construction also forms a complex predicate as a type of auxiliary construction. As seen from the contrast in the following, syntactic passive with the auxiliary verb *ci-ta* has a tight connection with its active form:

(17) a. haksayngtul-i paci-lul ccic-ess-ta
students-NOM pants-ACC tear-PAST-DECL
'Students tore trousers.'

b. paci-ka ccic-e ci-ess-ta
pants tear-COMP become-PAST-DECL
'The trousers were torn off.'

The various properties of the construction support the analysis that treats the auxiliary verb and the preceding main verb as a complex predicate, as sketched in the following structure:

(18)



The structure basically allows the auxiliary verb *ci-ess-ta* to combine with its lexical complement, the main verb *ccic-e* 'tear'. The resulting expression forms a *hd-lex-ex* (head-lexical-expression), first and then inherit the main verb's arguments to the resulting expression.[12] The combination of these two is licensed by the following grammar rule in the KRG:

(19) Head-LEX Rule:

$$\begin{bmatrix} \textit{hd-lex-ex} \\ \text{COMPS } \boxed{A}\ \text{LEX} + \end{bmatrix} \rightarrow \boxed{1} \begin{bmatrix} \text{LEX} + \\ \text{COMPS } \boxed{A} \end{bmatrix}, \text{H} \begin{bmatrix} \text{AUX} + \\ \text{COMPS } \langle \boxed{1} \rangle \end{bmatrix}$$

The rule specifies that the auxiliary verb ([AUX +]) combines not with a phrasal but with a lexical (LEX) complement ($\boxed{1}$), the result of whose combination is still a LEX expression. This system, interacting with appropriate lexical entries for auxiliary verbs, will license a complex-predicate structure and allow us to capture the syntactic independence of the main verb from the passive auxiliary as well as the tight syntactic unit between the two.

The analysis sketched here once again relies on the tight interaction between lexical information and the grammar rule based on argument composition can successfully parse a variety of complex predicate constructions including auxiliary constructions, light verb constructions, bound noun constructions, and so forth.[13] In particular, we assume that this type of Head-LEX Rule and argument composition is language particular.

3.3.4 Relative Clause Constructions

Unlike English, Korean employs no relative pronouns like *who* or *which*. In addition, the predicate of the relative clause preceding the head noun is marked with a morphological marker depending on the type of tense information.[14]

(20) Tom-i $\_\__i$ ilk-nun/un/ul chayk$_i$
Tom-NOM read-PRES.PNE/PST.PNE/FUT.PNE book
'the book that Tom reads/read/will read'

The prenominal markers in (20) in a sense function both as a relative pronoun and tense marker. As also expected, the language also allows relativization from an embedded clause:

[12] This kind of argument composition is different from previous analyses, mainly in that the composition happens in syntax rather than in the lexicon.

[13] The properties of complex predicates can be found in many phenomena in the language. For the analyses in the similar vein, see Kim and Yang (2004a) for Korean auxiliary constructions, Kim and Yang (2008) for passive constructions, Kim and Yang (2007) for bound constructions, Kim et al. (2007) for light verb constructions.

[14] Korean also employs the so-called IHRC (internally headed relative clause). The KRG also deals with such relative clause constructions. See Kim (2006) for the analysis and its implementation in the LKB.

(21) John-i [Mary-ka $\__i$ mekessta-ko] malha-n sakwa$_i$
John-NOM Mary-NOM ate-COMP say-PNE apple
'the apple that John said Mary ate yesterday'

The key point of our treatment of relative clauses includes the lexical constraints on the *v-rel-mod* verb heading the relative clause, a gap-introducing rule, and a grammar rule licensing the combination of a nominal head with a relative clause modifying it. The lexical constraints on the *v-rel-mod* will add the feature MOD, guaranteeing that a *v-rel-mod* element marked with a prenominal ending will modify a nominal element through the head feature MOD. The gap-introducing rule ensures the relative clause to be an incomplete sentence with one missing gap. As specified in the following feature description in the LKB (Linguistic Knowledge Building) System, the rule allows any of the elements in the SUBJ or COMPS to be introduced as a GAP element:

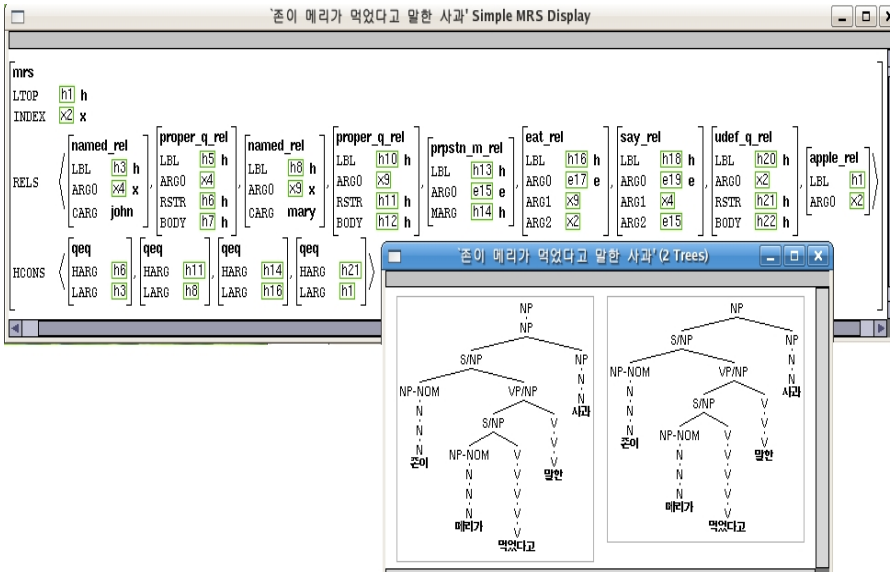
```
binary-start-gap-rule-1 := binary-sg &
 [ SYN.VAL [ SUBJ <>,
             COMPS <>,
             GAP <! #2 !> ],
   ARGS < #1 & [ SYN [ HEAD [ CASE.GCASE nom, PRD - ],
                       VAL [ SUBJ <>, COMPS <> ] ] ],
          [ SYN.VAL [ SUBJ < #1 >,
                      COMPS < #2 > ] ] > ].
```

This GAP value is passed upto the tree until it meets its filler to generate a long distance dependency like (21). For example, the word *mek-ess-ta-ko* 'eat-PST-DECL-COMP' selects two arguments. However, its COMPS can be realized as a GAP element according to the gap introducing rule described in the above. The *v-rel-mod* word *malha-n* has the information that it modifies a nominal element. In addition, the relative-clause modifying rule given in the below will terminate this GAP value when the index value of the GAP is identical with the modified nominal element:

```
head-rel-mod-rule := binary &
 [ SYN.VAL.GAP <! !>
   ARGS < ph-ex & [ SYN.VAL [ MOD < #1 & [ SYN.HEAD.POS noun,
                                           SEM.INDEX #2 ] >,
                                 GAP <! [ SEM.INDEX #2 ] !> ] ],
          syn-st & #1 & [ SYN.VAL [ GAP <! !>,
                                    ... ] ] > ].
```

As indicated in the first element of the ARGS value, the relative clause modifies a nominal element whose index value is identical with that of the GAP's value.

Equipped with these three fundamental mechanisms, the grammar allows us to parse the syntactic as well semantic structures of relative clause constructions. For example, the following parsed structures and MRS for sentences like (21) are what the grammar obtains within the system:



Leaving aside other semantic relations, we can at least observe that the ARG2 value of *eat_rel*, x2, is coindexed with the ARG0 value of *apple_rel*. The grammar can correctly parse relative clauses as well as generate a proper MRS meaning representation.

3.4 Other Phenomena Covered in the Grammar

Including the phenomena we discussed in the previous sections, the first phase of the KRG also covers major constructions in the language as given in the following and the results of the implementation are presented in the cited references:

- Multiple Nominative Constructions: Kim et al. (2007)
- Honorification: Kim et al. (2006)
- Coordination: Kim and Yang (2006)
- External and Internally Head Relative Clause: Kim (2006)
- Bound noun construction: Kim and Yang (2007)
- Comparative construction: Kim et al. (2010)
- Passive construction: Kim and Yang (2008)

Most of these constructions have also been challenges to theoretical linguists. The first phrase of the KRG has shown us that it is possible to build a deep linguistic processing grammar for Korean that can be applicable for computational purposes.

3.5 Shortcomings and Issues in the First Phase

As briefly shown, the first phase of the KRG had been quite successful in terms of analyzing major Korean constructions. Focusing on linguistic data, this phase of the KRG, however, ran into several important issues concerning efficiency in parsing and lack of a generation system.

That is, the first phase of the KRG, focusing on linguistically significant data, eventually had limits to parsing a large scale of naturally occurring data, which is a prerequisite to the practical uses of the developed grammar in the area of MT (machine translation). In addition, the first phase KRG focused only on parsing with no generation module and thus could not be applicable for a real-life use such as an MT system. The MT architecture the DELPH-IN project employs also parsing, transfer, and generation, as shown in Figure 3 Bond et al. (2005):

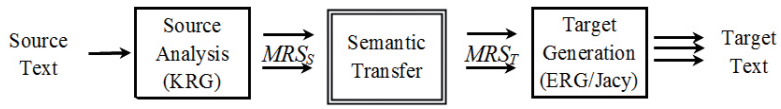


Figure 3: HPSG/MRS-based MT Architecture

For the KRG to be part of this multilingual-text based MT system, the grammar needs to be tailored for the other grammars participating in the DELPH-IN consortium so that the KRG's semantic representations can be compatible with those of other resource grammars like the ERG and JACY.

In addition, some types and rules defined in the KRG have caused problems for generation. In the KRG, the types and rules were defined from linguistic perspectives, not considering generation issues, causing memory overflow errors even for some simple cases. In particular, the complex morphological system in the first phase blocked the generation of most of the sentences headed by a heavily inflected verb.

Another issue is the refinement of the grammar, in particular with respect to the issue of robust parsing and generation. Consider the following set of examples:[15]

(22) a. ney/ni cham ippu/yeppu-ney
you/you really pretty-DECL
'You are really pretty.'

b. ney/ni cham ippu/yeppu-kwun

c. ney/ni cham ippu/yeppu-kwuna

d. ney/ni cham ippu/yeppu-e

All these examples have variant forms of the subject pronoun, verb form, head verb, and mood suffixes, but have the identical truth conditional meaning. For

[15] As a reviewer point outs, each of these may have different semantic or pragmatic effects.

example, the second person subject can alternate between *ney* and *ni* depending on the colloquial style. The predicate form can also be either the standard one *yeppu* or a colloquial one *ippu*. The mood suffix can be different too. However, all these variants need to have the identical MRS representation. Similar phenomena are prevalent in the language, but the KRG at the first phase is not sophisticated enough to distinguish such stylistically different sentences.

4 Korean Resource Grammar: the Second Phase

4.1 Directions for the Improvement

One of the main motivations for the grammar improvement in the second phase has been thus to achieve more balanced approaches between linguistic-based or deep processing and practical purposes.[16] To figure out the problems and issues of the first phase KRG, we first evaluated its coverage and performance using a large size of data. This experiment would enable us to track down what causes parsing inefficiencies and generating clog.

In developing the KRG further and to resolve the shortcomings in the first phase, we have employed two strategies for improvement; (i) using shared grammar libraries and (ii) exploiting large text corpora (using the Korean portions of multilingual texts). These two strategies are essential in expanding the coverage with deep linguistic processing. We shared grammar libraries with the Grammar Matrix in the grammar Bender et al. (2002) as the foundation of the KRG in the second phase. We tried to customize the KRG to the Grammar Matrix more, so that the KRG can also be part of the multilingual grammar engineering. In addition, we exploit naturally occurring texts as the generalization corpus. To perform the adopted strategies in a more systematic way, we set up our working principles as following:

- The Grammar Matrix is applied when a judgment about structure (e.g. semantic representation) is needed.
- The KRG is applied when a judgment about Korean is needed.
- The resulting grammar has to run on both PET and LKB without any problems.
- Parsing needs to be accomplished as robustly as possible, and generation needs to be done as strictly as possible.

4.2 Adding the Generation Module

As pointed out, the important missing part in the first phase of the KRG is a generation module. It was not an easy task to alter the structure of the KRG in the first phase from top to bottom in a relatively short time, mainly because the difficulties arise from converting each grammar module (optimized only for parsing)

[16] Part of the discussion here follows Song et al. (2010).

into something applicable to generation, and further from making the grammar run separately for parsing and generation.

Accordingly, we first rebuilt the basic schema of the KRG on the Grammar Matrix customization system, and then imported each grammar module from the KRG to the matrix-based frame (S4.1). In addition, we reformed the inflectional hierarchy to avoid any impediment to generation any longer (S 4.2). We also introduced the STYLE feature structure to discriminate different sentence (mood) styles. In what follows, we will see what kind of modifications and improvements we made in the grammar.

4.3 Modifying the Modular Structures

To improve the system's compatibility with other resource grammars in the LOGON, we first reclassified and cleaned up the organization of the file systems. This work, though not affecting the grammar's parsing efficiency, can help the DELPHIN research group refer to the KRG more easily.[17]

Additionally, we revised grammar modules in order to use the Grammar Matrix to a full extent. In this process, when inconsistencies arise from the previous KRG, we followed the strategies and working principles given in 4.1. We further transplanted each previous module into the second phase KRG (KRG2), while checking the attested test items used in the first phase KRG (KRG1). The test items consist of 6,180 grammatical sentences and 118 ungrammatical sentences, reflecting main linguistic phenomena in the language.

4.4 Refining the Grammar

4.4.1 Simplifying the Inflectional Hierarchy

As we have noted, the rich inflection system and the complex morphological system built in the KRG1 has induced clogging effects in generation. As a way of solving this, we simplified the inflectional system. As discussed in Kim and Yang (2004b), Korean has rigid ordering restrictions in the morphological paradigm for verbs, as shown in the following template:

(23) V-base + (Passive/Caussative) + (Hon) + (Tense) + Mood + (Comp)

The first phase KRG dealt with this ordering of suffixes by using a type hierarchy that represents a chain of inflectional slots (Figure 4: Kim and Yang (2004b)).

This hierarchy has its own merits, building a verbal element by step-by-step processes from the verb lexeme, as exemplified by an example like the following:

[17] For example, the root folder `krg` in the KRG includes the basic type definition language files (`*.tdl`). As a simplification, we subdivided the `types.tdl` into two files: `matrix.tdl` file covering general principles in the grammar and `korean.tdl` with language particular rules. This tdl includes two sub-files: `types-lex.tdl` for lexical types and `types-ph.tdl` for phrasal types. In addition, we reorganized the KRG1's `lexicons.tdl` file into the `lex` folder consisting of several sub-files in accordance with the POS values (e.g.; `lex-v.tdl` for verbs).

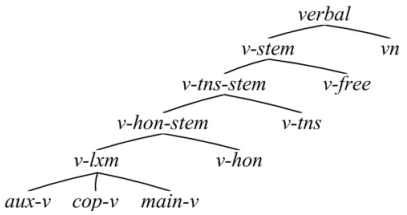


Figure 4: Korean Verbal Hierarchy

(24) a. [[[[[cap + hi]+si] +ess]+ ta] + ko] 'catch-Caus-Hon-Past-Decl-Comp'
b. *v-lxm* → *v-hon (v-hon-stem)* → *v-tns (v-tns-stem)* → *v-free (v-stem)* → *v-comp*

However, this complex system requires a large number of calculations in the generation process. Figure 5 and Table 1 explains the difference in computational complexity according to each structure.

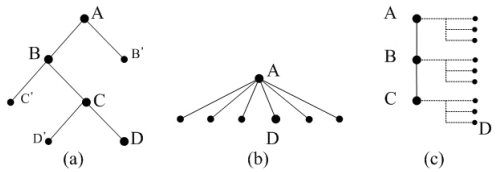


Figure 5: Calculating Complexity

In Figure 5, (a) is similar to Figure 4, while (b) is on the traditional template approach. Let us compare each complexity to get the target node D. For convenience' sake, let us assume that each node has ten constraints to be satisfied. In (a), since there are three parents nodes (i.e. A, B, and C) on top of D, D cannot be generated until A, B, and C are checked previously. Hence, it costs at least 10,000 (10[A] ×10[B] ×10[C] ×10[D]) calculations. In contrast, in (b), only 100 (10[A] ×10[D]) calculations is enough to generate node D. This means that the deeper the hierarchy is, the more the complexity increases. Table 1 shows (a) requires more than 52 times as much complexity as (b), though they have the same number of nodes.

Table 1: Complexity of (a) and (b)

	(a)		(b)
B'	10[A]×10[B']	100	10[A]×10[B']
C'	10[A]×10[B]×10[C']	1,000	10[A]×10[C']
D'	10[A]×10[B]×10[C]×10[D']	10,000	10[A]×10[D']
D	10[A]×10[B]×10[C]×10[D]	10,000	10[A]×10[D]
Σ		21,100	400

When generation is processed by LKB, all potential inflectional nodes are made before syntactic configurations according to the given MRS. Thus, if the hierarchy becomes deeper and contains more nodes, complexity of (a)-styled hierarchy

grows almost by geometric progression. This makes generation virtually impossible, causing memory overflow errors to the generation within the KRG1. A fully flat structure (b) is not always superior to (a). First of all, the flat approach ignores the fact that Korean is an agglutinative language. Korean morphological paradigm can yield a wide variety of forms; therefore, to enumerate all potential forms is not only undesirable but also even impossible.

The KRG2 thus follows a hybrid approach (c) that takes each advantage of (a) and (b). (c) is more flattened than (a), which lessens computational complexity. On the other hand, in (c), the depth of the inflectional hierarchy is fixed as two, and the skeleton looks like a unary form, though each major node (marked as a bigger circle) has its own subtypes (marked as dotted lines). Even though the depth has been diminished, the hierarchy is not a perfectly flat structure; therefore, it can partially represent the austere suffix ordering in Korean. The hierarchy (c), hereby, curtails the cost of generation.

4.4.2 Encoding the Sentence Style Information

In the newly developed grammar KRG2, we also introduced the feature STYLE, in order to enhance the performance of generation. As noted earlier in (22), different style (largely formal and informal) markings on the mood slot can be used with the identical truth conditional meaning (or same MRS representations). The choice between formal or informal sentence styles depends on context:

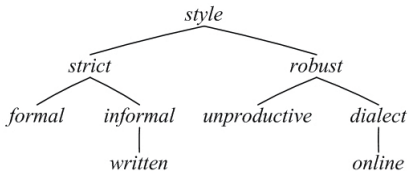


Figure 6: Type Hierarchy of STYLE

A robust parser should cover both styles, but in generation, the system needs to be consistent. In such a case, the grammar resorts to STYLE to filter out infelicitous results. The type hierarchy is sketched out in Figure 6. On the other hand, some variant forms that stem from the corresponding canonical forms falls under *robust* in Figure 6. For instance, if the text domain for generation is newspaper, we can select only *written* as our sentence choice, which excludes other styled sentences from our result.

Let us see (22) again. *ni* 'you' in (22b) is a dialect form of *ney*, but it has been used more productively than its canonical form in daily speech. In that case, we can specify STYLE of *ni* as *dialect* as given below. In contrast, the neutral form *ney* has an unspecified STYLE feature:

```
ni := n-pn-2nd-non-pl &
[ STEM < ``ni'' >, STYLE dialect ].
```

```
ney := n-pn-2nd-non-pl &
[ STEM < ``ney'' > ].
```

Likewise, since the predicate in (22) *ippu* 'pretty' stems from *yeppu* in (22), they share the predicate name '*_yeppu_a_1_rel*' (i.e. the RMRS standard for predicate names such as '_lemma_pos_sense_rel'), but differ in each STYLE feature. This means the examples in (22) share the same MRS structure regardless of their different style mood markings. These kinds of stylistic differences can take place at the level of (i) lexicon, (ii) morphological combination, and (iii) syntactic configuration. The KRG2 revised each rule with reference to its style type, yielding total 96 robust rules. As a welcoming result, we could enrich the possible outputs of our generation. Let us call the version reconstructed so far '**base**'.

5 Applications and Evaluation

5.1 Resources

To test the improvements in the grammar, we have used two multilingual corpora: *Sejong Bilingual Corpora*: SBC Kim and Cho (2001) and the *Basic Travel Expression Corpus*: BTEC Kikui et al. (2003). We exploited the Korean parts in each corpus, taking them as our generalization corpus data. Table 2 represents the configuration of two resources (KoEn: Korean-English, KoJa: Korean-Japanese):

Table 2: Generalization Corpora

	SBC	**BTEC**
Type	Bilingual	Multilingual
Domain	Balanced Corpus	Tourism
Words	KoEn : 243,788 KoJa : 276.152	914,199
T/T ratio	KoEn : 27.63 KoJa : 20.28	92.7
Avr length	KoEn : 16.30 KoJa : 23.30	8.46

We also adopted nine test suites sorted by three types (each test suite includes 500 sentences). As the first type, we used three test sets covering overall sentence structures in Korean; Korean Phrase Structure Grammar (**kpsg**; Kim (2004)), Information-based Korean Grammar (**ibkg**; Chang (1995)), and the SERI test set (**seri**; Sung and Jang (1997)).

Second, we randomly extracted sentences from each corpus, separately from our generalization corpus; two suites were taken from the Korean-English and Korean-Japanese pair in SBC (**sj-ke** and **sj-kj**, respectively). The other two suites are from the BTEC-KTEXT (**b-k**), and the BTEC-CSTAR (**b-c**); the former consists of relatively plain sentences, while the latter is composed of spoken ones.

Third, we obtained two test suites from sample sentences in two dictionaries; Korean-English (**dic-ke**), and Korean-Japanese (**dic-kj**). These suites assume to have at least two advantages with respect to our evaluation; (i) the sentence length is longer than that of BTEC as well as shorter than that of SBC, (ii) the sample sentences on dictionaries are normally made up of useful expressions for translation.

5.2 Methods

We have tried to do experiments and improve the KRG, following the three steps repeatedly: (i) evaluating, (ii) identifying, and (iii) exploiting. In each step, we first tried to parse the nine test suites and generate sentences with the MRS structures obtained from the parsing results, and measured their coverage and performance. Here, 'coverage' means how many sentences can be parsed or generated, and 'performance' represents how many seconds it takes on average. In the second step, we identified the most serious problems. In the third step, we sought to exploit our generalization corpora in order to remedy the drawbacks. After that, we repeated the procedures until we obtain the desired results.

5.3 Experiments

The methods just sketched yielded two versions: **KRG1** and **base**. Our further experiments consist of four phases; **lex**, **MRS**, **irules**, and **KRG2**.

Expanding the lexicon: To begin with, in order to broaden our coverage, we expanded our lexical entries with reference to our generalization corpus and previous literature. Verbal items are taken from Song (2007) and Song and Choe (2008), which classify argument structures of Korean verbal lexicon into subtypes within the HPSG framework in a semi-automatic way.[18] For other word classes, we extracted lexical items from the POS tagged SBC and BTEC corpora. Table 3 explains how many items we extracted from our generalization corpus. Let us call this version '**lex**'.

Table 3: Expansion of Lexical Items

verbal nouns	4,474
verbs and adjectives	1,216
common nouns	11,752
proper nouns	7,799
adverbs	1,757
numeral words	1,172

MRS: Generation in LKB, as shown in Figure 3, deploys MRS as the input, which means our generation performance hinges on the well-formedness of MRS. In other words, if our MRS is broken somewhere or constructed inefficiently, generation outputs are directly affected. For instance, if the semantic representation

[18] We cannot automatically acquire the subcatergorization frames for new lexical items from the given corpora because of the issues related to accuracy.

does not scope, we will not generate correctly. We were able to identify such sentences by parsing the corpora, storing the semantic representations and then using the semantic well formedness checkers in the LKB. We identified all rules and lexical items that produced ill-formed MRSs using a small script and fixed them by hand. This had an immediate and positive effect on coverage as well as performance in generation. We refer to these changes as '**MRS**'.

Different inflectional forms for sentence styles: Texts in our daily life are actually composed of various styles. For example, spoken forms are normally more or less different from written ones. The difference between them in Korean is so big that the current version of KRG can hardly parse spoken forms. Besides, Korean has lots of compound nouns and derived words. Therefore, we included these forms into our inflectional rules and expanded lexical entries again (3,860 compound nouns, 2,791 derived words). This greatly increased parsing coverage. We call this version '**irules**'.

Grammaticalized and Lexicalized Forms: There are still remaining problems, because our test suites contain some considerable forms. First, Korean has quite a few grammaticalized forms; for instance, *kupwun* is composed of a definite determiner *ku* and a classifier for human *pwun* "the person", but it functions like a single word (i.e. a third singular personal pronoun). In a similar vein, there are not a few lexicalized forms as well; for example, a verbal lexeme *kkamek-* is composed of *kka-* "peel" and *mek-* "eat", but it conveys a sense of "forget", rather than "peel and eat". In addition, we also need to cover idiomatic expressions (e.g. "thanks") for robust parsing. Exploiting our corpus, we added 1,720 grammaticalized or lexicalized forms and 352 idioms. Now, we call this '**KRG2**'.

Table 4 compares KRG2 with KRG1, and Figure 7 shows how many lexical items we have covered so far.

Table 4: Comparison between KRG1 and KRG2

	KRG1	KRG2
# of default types	121	159
# of lexical types	289	593
# of phrasal types	58	106
# of inflectional rules	86	244
# of syntactic rules	36	96
# of lexicon	2,297	39,190

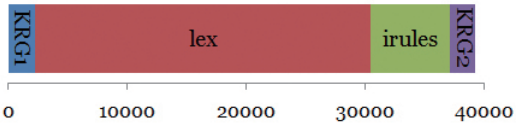


Figure 7: Size of Lexicon

5.4 Evaluating the Results

Table 5 shows the evaluation measure of this study. 'p' and 'g' stand for 'parsing' and 'generation', respectively. '+' represents the difference compared to KRG1. Since KRG1 does not generate, there is no 'g+'.

Table 5: Evaluation of the KRG2

	coverage (%)				ambiguity	
	p	p+	g	s	p	g
kpsg	77.0	-5.5	55.2	42.5	174.9	144.4
ibkg	61.2	41.8	68.3	41.8	990.5	303.5
seri	71.3	-0.8	65.7	46.8	289.1	128.4
b-k	43.0	32.6	62.8	27.0	1769.4	90.0
b-c	52.2	45.8	59.4	31.0	1175.8	160.6
sj-ke	35.4	31.2	58.2	20.6	358.3	170.3
sj-kj	23.0	19.6	52.2	12.0	585.9	294.9
dic-ke	40.4	31.0	42.6	17.2	1392.7	215.9
dic-kj	34.8	25.2	67.8	23.6	789.3	277.9
avr	**48.7**	**24.5**	**59.1**	**28.8**	**836.2**	**198.4**

On average, the parsing coverage increases **24.5%**. The reason why there are negative values in 'p+' of **kpsg** and **seri** is that we discarded some modules that run counter efficient processing (e.g., the grammar module for handling floating quantifiers sometimes produces too many ambiguities.). Since KRG1 has been constructed largely around the test sets, we expected it to perform well here. If we measure the parsing coverage again, after excluding the results of **kpsg** and **seri**, it accounts for **32.5%**.[19] The generation coverage of KRG2 accounts for almost **60%** per parsed sentence on average. Note that KRG1 could not generate at all. 's' (short for 'success') means the portion of both parsed and generated sentences (i.e. 'p'$\times$'g'), which accounts for about **29%**. Ambiguity means '# of parses/# of sentences' for parsing and '# of realizations/# of MRSes' for generation. The numbers look rather big, which should be narrowed down in our future study.

In addition, we can find out in Table 5 that there is a coverage ordering with respect to the type of test suites; 'test sets $>$ BTEC $>$ dic $>$ SBC'. It is influenced by three factors; (i) lexical variety, (ii) sentence length, and (iii) text domain. This difference implies that it is highly necessary to use variegated texts in order to improve grammar in a comprehensive way.

Figure 8 to 11 represent how much each experiment in Section 5.3 contributes to improvement. First, let us see Figure 8 and 9. As we anticipated, **lex** and **irules** contribute greatly to the growth of parsing coverage. In particular, the line of **b-c** in Figure 9, which mostly consists of spoken forms, rises rapidly in **irules** and **KRG2**. That implies Korean parsing largely depends on richness of lexical rules. On the other hand, as we also expected, **MRS** makes a great contribution to generation coverage (Figure 9). In **MRS**, the growth accounts for **22%** on average. That implies testing with large corpora must take precedence in order for coverage to grow.

[19] The running times, meanwhile, becomes slower as we would expect for a grammar with greater coverage. However, we can make up for it using the PET parser, as shown in Figure 10.

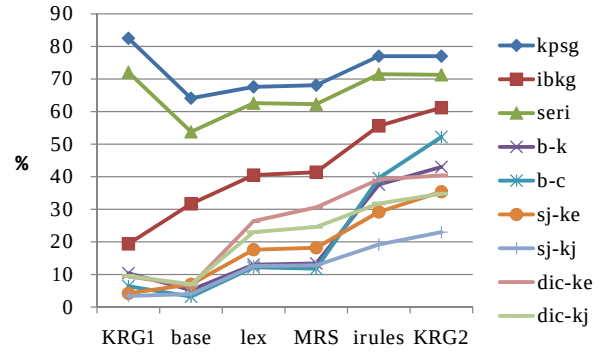


Figure 8: Parsing Coverage (%)

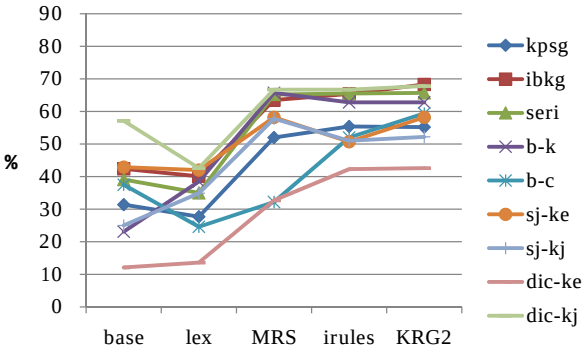


Figure 9: Generation Coverage (%)

Figure 10 and 11 shows performance in parsing and generation, respectively. Comparing to **KRG1**, our Matrix-based grammars (from **base** to **KRG2**) yields fairly good performance. It is mainly because we deployed the PET parser that runs fast, whereas KRG1 runs only on LKB. Figure 11, on the other hand, shows that the revision of MRS also does much to enhance generation performance, in common with coverage mentioned before. It decreases the running times by about **3.1** seconds on average.

5.5 Future Directions

The second phase of the grammar, KRG2, has improved the efficiency of the deep processing grammar significantly. In particular, the addition of the generation module opened a new direction for the KRG such as building deep-processed rich treebanks and developing an MT system. However, for the grammar to cover a wider ranger of naturally occurring data, whose results can be ultimately applied to NLP applications, much more work needs to be done as briefly summarized in the following:

- refining the current KRG so that it can efficiently cover phenomena such as *pro* drop, serial verb construction, coordination and subjunction, ellipsis, wh-constructions,

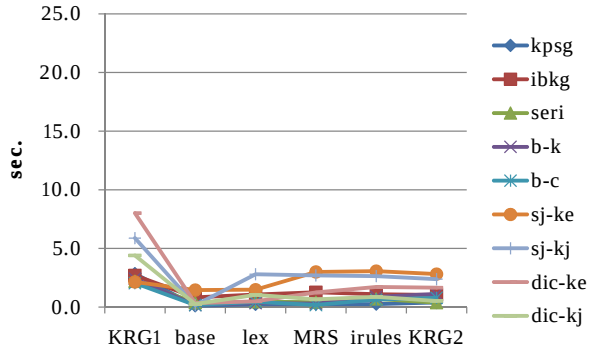


Figure 10: Parsing Performance (s)

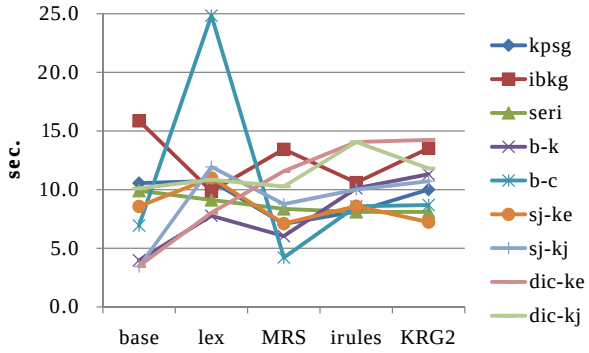


Figure 11: Generation Performance (s)

other semantics-related phenomena, etc.

- developing fine-grained semantics using MRS that can capture scope, event structures, message types, linking between syntax and semantics
- incrementally increasing coverage of clause internal syntax in Korean
- incorporating the use of default entries for words unknown to the Korean lexicon
- testing with real-life corpora and expanding more coverage
- developing a more sophisticated generator that can be used for real-life applications
- developing an MT system based on deep processing grammars

6 Conclusion

The second phase of the KRG has been successfully included in the LOGON repository and it is now available online (http://krg.khu.ac.kr). It is hard to deny the fact that in building an efficient grammar, expressive accuracy has often been sacrificed in order to achieve computational tractability (Siegel 2002, Oepen et al. 2002).

However, putting linguistic generalizations aside has brought difficulties expanding the coverage and eventually building a large scale of grammar. The morphological, syntactic, and semantic parsing system we have developed here with typed feature structures is an effort to solve such preexisting problems while keeping linguistic insights, thus making the Korean morphology, syntax, and semantics much simpler.

The work described here, even though it is an on-going project, achieves welcoming coverage of major constructions that have been widely discussed in the literature as well as real-life data. The research presented in this paper provides robust, deep parsing results and introduces effective generation outputs too. We hope to have shown that the ongoing project of developing the KRG is prospective in two respects: as a means of testing linguistic hypotheses and seeking the potential to be integrated in applications which require natural language understanding, including machine translation, question-answering system, NL interfaces, and so forth.

References

n.d.

Baldwin, Timothy, Mark Dras, Julia Hockenmaier, Tracy King, and Gertjan van Noord. 2007. The impact of deep linguistic processing on parsing technology. In *Proceedings of the 10th International Conference on Parsing Technologies*, 36–38.

Bender, Emily M., Dan Flickinger, and Stephan Oepen. 2002. The Grammar Matrix: An Open-Source Starter-Kit for the Rapid Development of Cross-Linguistically Consistent Broad-Coverage Precision Grammars. In *Procedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*.

Bond, Francis, Stephan Oepen, Melanie Siegel, Ann Copestake, and Dan Flickinger. 2005. Open Source Machine Translation with DELPH-IN. In *Proceedings of Open-Source Machine Translation: Workshop at MT Summit X*.

Callmeier, Ulrich. 2000. PET–a Platform for Experimentation with Efficient HPSG Processing Techniques. *Natural Language Engineering* 6(1), 99–107.

Chang, Suk-Jin. 1995. *Information-based Korean Grammar*. Seoul, Hanshin Publishing.

Copestake, Ann. 2002. *Implementing Typed Feature Structure Grammars*. Stanford, CA, CSLI Publications.

Copestake, Ann, Dan Flickinger, Carl Pollard, and Ivan A. Sag. 2005. Minimal Recursion Semantics: An Introduction. *Research on Language and Computation* 3(4), 281–332.

Flickinger, Dan. 2000. On Building a More Efficient Grammar by Exploiting Types. *Natural Language Engineering* 6 (1) (Special Issue on Efficient Processing with HPSG), 15 – 28.

Kikui, Genichiro, Eiichiro Sumita, Toshiyuki Takezawa, and Seiichi Yamamoto. 2003. Creating corpora for speech-to-speech translation. In *Proc. of the EUROSPEECH03*, Geneve, Switzerland. 381–384.

Kim, Jong-Bok. 2004. *Korean Phrase Structure Grammar*. Seoul, Hankwuk Publishing.

Kim, Jong-Bok. 2006. Parsing head internal and external relative clause constructions in Korean. *LNAI (Lecture Notes in Artifical Intelligence)* 4188, 111–117.

Kim, Jong-Bok, Kyung-Sup Lim, and Jaehyung Yang. 2007. Structural ambiguities in the light verb constructions: lexical relatedness and divergence. *Linguistics* 15(2), 207–231.

Kim, Jong-Bok, and Peter Sells. 2008. *English Syntax: An Introduction*. Stanford, CSLI Publications.

Kim, Jong-Bok, Peter Sells, and Jaehyung Yang. 2006. Parsing Korean honorification phenomena in a typed feature structure grammar. *LNAI (Lecture Notes in Artificial Intelligence)* 4013, 254–265.

Kim, Jong-Bok, Peter Sells, and Jaehyung Yang. 2007. Parsing two types of multiple nominative constructions: A constructional approach. *Language and Inforamtion* (11), 25–37.

Kim, Jong-Bok, and Jaehyung Yang. 2003. Korean Phrase Structure Grammar and Its Implementations into the LKB System. In *Proceedings of the 17th Pacific Asia Conference on Language, Information and Computation*.

Kim, Jong-Bok, and Jaehyung Yang. 2004a. Korean auxiliary system and a computational implementation (in Korean). *Language Research* 3, 195–226.

Kim, Jong-Bok, and Jaehyung Yang. 2004b. Projections from Morphology to Syntax in the Korean Resource Grammar: Implementing Typed Feature Structures. In *Lecture Notes in Computer Science*, Vol. 2945. Springer-Verlag, 13–24.

Kim, Jong-Bok, and Jaehyung Yang. 2005. Parsing Korean case phenomena in a typed-feature structure grammar. *LNCS (Lecture Notes in Computer Science)* 3406, 60–72.

Kim, Jong-Bok, and Jaehyung Yang. 2006. Coordination structures in a typed featuer structure grammar: Formalization and implementation. *LNCS/LNAI*.

Kim, Jong-Bok, and Jaehyung Yang. 2007. On the syntax and semantics of the Korean bound noun constructions: With a computational implementation. *Language Science* 43, 161–187.

Kim, Jong-Bok, and Jaehyung Yang. 2008. Syntax and semantics of passive and related constructions in Korean: A computational implementation. *Language Research* 24(1), 69–95.

Kim, Jong-Bok, Jaehyung Yang, and Sanghoun Song. 2010. Parsing Korean Comparative Constructions in a Typed-Feature Structure Grammar. *Language and Information* 14(1), 1–24.

Kim, Se-jung, and Nam-ho Cho. 2001. The progress and prospect of the 21st century Sejong project. In *ICCPOL-2001*, Seoul. 9–12.

Oepen, Stephan. 2001. [incr tsdb()] — competence and performance laboratory. User manual. Technical report, Computational Linguistics, Saarland University.

Sag, Ivan A., Thomas Wasow, and Emily M. Bender. 2003. *Syntactic Theory: A Formal Introduction*. Stanford, CA, CSLI Publications.

Siegel, Melanie, and Emily M. Bender. 2002. Efficient Deep Processing of Japanese. In *Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization*.

Song, Sanghoun. 2007. A Constraint-based Analysis of Passive Constructions in Korean.

Song, Sanghoun, and Jae-Woong Choe. 2008. Automatic Construction of Korean Verbal Type Hierarchy using Treebank. In *Proceedings of HPSG2008*.

Song, Sanghoun, Jong-Bok Kim, Francis Bond, and Jaehyung Yang. 2010. Development of the Korean resource grammar: Towards grammar customization. In *Proceedings of the 8th Workshop on Asian Language Resources, COLING 2010*, 144–152.

Sung, Won-Kyung, and Myung-Gil Jang. 1997. SERI Test Suites '95. In *Proceedings of the Conference on Hanguel and Korean Language Information Processing*.

Uszkoreit, Hans. 2002. New chances for deep linguistic processing. *Proc. of the 19th International Conference on Computational Linguistics*.