A Rule-Based Analysis from Raw Korean Text to Morphologically Annotated Corpora

Kiyong Lee*†
Korea University

Markus Schulze‡
Universität Erlangen-Nürnberg

K. Lee and M. Schulze. 2002. A Rule-Based Analysis from Raw Korean Text to Morphologically Annotated Corpora. *Language and Information 6.2*, 105–128. Morphologically annotated corpora are the basis for many tasks of computational linguistics. Most current approaches use statistically driven methods of morphological analysis, that provide just POS-tags. While this is sufficient for some applications, a rule-based full morphological analysis also yielding lemmatization and segmentation is needed for many others. This work thus aims at [1] introducing a rule-based Korean morphological analyzer called **Kormoran** based on the principle of linearity that prohibits any combination of left-to-right or right-to-left analysis or backtracking and then at [2] showing how it can be used as a POS-tagger by adopting an ordinary technique of preprocessing and also by filtering out irrelevant morpho-syntactic information in analyzed feature structures. It is shown that, besides providing a basis for subsequent syntactic or semantic processing, full morphological analyzers like Kormoran have the greater power of resolving ambiguities than simple POS-taggers. The focus of our present analysis is on Korean text. **(Korea University and Universität Erlangen-Nürnberg)**

Key words: ambiguity, allomorph, base form, Hangul, morphological analysis, output filter, preprocessor, rule-based, tagging, word-form

1. Introduction

Efficient preprocessing and tagging techniques are necessary for linguistic corpus research because it depends not on raw but preprocessed tagged corpora. The purpose of this work is thus to develop a technique for preprocessing raw Korean

* Department of Linguistics, Korea University, 5-ka, Anam-dong, Sungbuk-ku, Seoul 136-701, Korea. E-mail: klee@korea.ac.kr.
† This research was part of the joint project MIRAC(1997-2000) supported by the Korea Research Foundation. The authors would like to thank Prof. Roland Hausser of the Friedrich-Alexander University of Erlangen-Nuremberg for providing us with an excellent research environment and generous support and also Björn Beutel for his constant improvement of Malaga especially for processing Hangul in Korean text. We also owe many thanks to three anonymous reviewers for their detailed constructive comments that we believe helped improve the paper and made it more concise and readable.
‡ Abteilung Computerlinguistik, Universität Erlangen-Nürnberg, Bismarckstrasse 6, 91054 Erlangen, Germany. E-mail: max@linguistik.uni-erlangen.de.

corpora and then automatically analyzing word-forms in text and tagging them with categorial features such as part of speech. As a tool for analyzing Korean word-forms, we use the Korean morphological analyzer **Kormoran**[1] based on Malaga, a system for developing Hausser's (2001) Left Associative Grammars with attribute-value structures.[2]

The process of morphological analysis and tagging of text involves at least four tasks:

(1) Tasks for Tagging Text

i. Preprocessing raw text to build word lists
ii. Building a unique word list and index
iii. Morphologically analyzing unique word lists
iv. Filtering and disambiguation to produce part-of-speech tagged text from Malaga output

As itemized here, this paper will proceed in four steps in an attempt to formulate precise specifications for overall Korean text preprocessing and tagging techniques.

Since a corpus consists of actual text written in a particular language, particular problems arise from the peculiarities of its writing system and also of its formatting or documentation style. Korean text, for instance, is not composed of characters of a single writing system Hangul, but, mainly for historical reasons, mixed with Chinese characters. It can also have Latin characters as in Western text. Unlike the alphabet letters or the Chinese or Japanese Kana characters, Hangul characters can be viewed both as alphabetic and syllabic.[3] Hence, an encoding system with a wider range of characters is necessary for Hangul as well as for Chinese characters that are used in Korean text.[4] These language-specific textual problems must be taken into account when we design a modular system consisting of a preprocessor (with output filters), a morphological analyzer, and a part-of-speech tagger.

1. **Kormoran** is the German name for the bird cormorant. Since the morphological analyzer **Kormoran** is of German origin, being first developed in 1994 by Kiyong Lee, who was a DAAD scholar visiting Abteilung Computerlinguistik, FAU Erlagen, Germany, it seems appropriate to take the German name.
2. Malaga is an acronym standing for "*M*erely *a* *l*eft-*a*ssociative *g*rammar *a*pplication".
   Malaga is more than just a language: it is a programming language with a compiler, a debugger, and graphical interface. It was developed and upgraded by Björn Beutel (2002). The most recent version is Malaga-6.7, which is available with documentation, binaries, and sources through http://www.linguistik.uni-erlangen.de/ bjoern/Malaga.html or ../malaga-6.7.-win32.zip, or by writing to malaga@linguistik.uni-erlangen.de.
   Malaga can process Hangul accommodating the Korean standard coding system KSC5601-1987 of Hangul. For information on the Korean morphological analyzer Kormoran, please contact klee@korea.ac.kr(Prof. Kiyong Lee) or malaga@linguistik.uni-erlagen.de, if urgent.
   An integrated version of Malaga using unicode is in preparation.
3. See Lee (1995) for details on Hangul characters.
4. Unicode could be the solution for this problem. Most Korean systems currently running on a larger machine, however, rely on the Korean standard coding system KSC5601-1989 which encoded 2,350 Hangul syllable characters and 4,888 Chinese characters.

2. Preprocessing Raw Korean Text

Linguistic corpora consist of text from various sources. For many years, the main sources used to be printed articles, books, and newspapers.[5] Manual labor was necessary to type in all these materials to build big corpora. In recent years, however, worldwide web sites have provided tons of material in the easily available electronic form including Korean material. Hence no technical difficulties aside from copyright problems arise from collecting raw material.

For linguistic corpus research, the preprocessing of raw material is absolutely necessary. It must clear out unwanted (graphical) markup like typesetting information and artifacts, since they do not constitute the content of text itself. On the other hand, it must preserve the original text without erasing punctuation marks or other meaningful elements in text. These must be kept because they contribute to the structure and content of the text.

For big corpora, a preprocessor must also provide formats suited for feeding into various applications such as a corpus database. For general purposes, it should keep a uniform storing format. But for particular applications, it should have a different but connected output filter module which produces output in various formats. This output filter should, for instance, produce unique word-form lists for running Malaga with Kormoran.

2.1 Hypertext or Text with Markup

Like English or German text, Korean text available through the electronic medium may often be in postscript format, in LaTeX format or enriched with HTML markup. An ordinary online newspaper article nowadays has the form of hypertext with HTML markup for hyper-links and page formatting. The main part of the original article may, for instance, consist of two or three short paragraphs. But the entire text is normally filled with various and yet routine markup lines which provide no content concerning the article itself.

Although web-browsers like Explore or Netscape provide a function for erasing HTML markup from text, the Korean text downloaded through such an erasure function still contains some page-layout markings along with the copyright information, as shown below:

(2)

```
                                        [Image][Image][Image]
                                    01/18(일) 19:00 [Image]

[Image]      내일 더 춥다...서울 영하11도

[부가서비스] [Image] 대한을 하루 앞둔 19일 전국의 낮 최고 기온이
             대부분 영하로 떨어지는 등 올들어 가장 추운
```

5. Much of the information contained in this section is provided for those who are not well acquainted with Hangul text. In fact, being written in English, the entire paper aims at foreign readers with little knowledge of Korean and its writing system, Hangul.

날씨를 보이겠다. . . .

Since markup expressions for hyper-linking, formatting, and page layout contain no information directly related to the content of the text itself, preprocessing requires a finer-grained way of erasing all these unwanted elements from text.

2.2 Punctuation Marks

Preprocessing must preserve punctuation marks, quotation marks, parentheses, and brackets as in the original text. Properly speaking, these symbols do not belong to any particular writing system like the Latin or Hangul writing system. They do, however, belong to the general character set, thus a being genuine part of the text.

Although they are neither Hangul characters nor alphabet letters, they constitute an essential part of text, for they belong to the general character set independent of other characters belonging to a particular (natural) writing system.

Although it contained some diacritic marks for reading purposes, old Korean text written in Chinese characters didn't contain any punctuation marks. It has been a century or less since Korean writings, whether written in Chinese characters or in Hangul, started to be marked with the period, comma, hyphen, brackets, and other punctuation or parenthetical marks as commonly found in Western text. Because of their late introduction, the convention of their use has not been fully established, thus making it sometimes difficult to set up a consistent way of analyzing them.

The use of punctuation marks in Korean writing differs from that in Western writing at some cases. The colon ":", for instance, is hardly used as a clausal marker in Korean writing, although it may be used as representing the ratio as in a mathematical expression like "3:2". The center dot "·" which does not appear in ordinary Western text may appear in Korean text and is sometimes used like the ampersand "&", meaning "and", especially when it represents co-authorship like "이기용(Lee Kiyong)·슐제(Schulze)". These variations again require a specific way of processing the use of punctuation marks in Korean text.

Quotation marks and parentheses play an important role in interpreting text. A sentence with a quoted or parenthesized phrase may become syntactically ill-formed or be interpreted differently, when the quotation marks and parentheses are deleted.

(3) Text with or without Quotation Marks

(i) a. The woman says: ''I'll die.''

b. The woman says that I'll die.

(ii) a. ''Rome'' consists of four letters.
b. Rome consists of four letters.

In (ia), "I" refers to the woman, but in (ib) to the hearer. (iia) is both syntactically and semantically well-formed because "Rome" here refers to the word "Rome". On the other hand, (iib) is semantically anomalous because Rome here refers to the city of Rome. Hence, these quotation marks must be preserved as in the original text for syntactic as well as semantic reasons.

In no way, preprocessing attempts to resolve or interpret all these usage or conventional differences or meanings. It just preserves them intact for subsequent steps of processing text.

2.3 Numeric Symbols

Korean text may contain not only Roman and Arabic numerals, but also numerals in various types of Chinese characters.[6] They often form a genuine part of Korean text. When they do, they behave like their phonetic equivalents in Hangul.

(4) Numerals as Proper Part of Text

a. 나는 60이다
na-nun 60-ita
''I-TOPIC 60-am''= I am sixty
b. 나는 육십이다
na-nun yuksip-ita
c. *나는 60다
*나는 육십다

Just like the Hangul word "육십"(yuksip), the Arabic numeral "60" is part of the sentence, taking a copular verb "이다". If it is deleted, the sentence becomes ill-formed. It cannot be concatenated with the ending "다"(ta) directly, just as its phonetic equivalent "육십"(yuksip) cannot, for this ends in a closed syllable.

Chinese numerals never appear in mathematical or other technical papers nor in financial or accounting bookkeeping. They may, however, be used in legal documents as well as scholarly articles. Preprocessing again must keep them all intact, for they form proper parts of text.

2.4 Chinese and Foreign Characters

Korean text may contain not only numerals but also other words in Chinese characters. The use of Chinese characters used to be predominant in Korean text

6. Chinese characters are often of two different forms, full and simplified. For example, the Chinese numeral standing for one or 1 can be represented in a full form "壹" or a simplified form "一" which are both pronounced the same, as "il" in Korean.

for many centuries. It was slowly reduced only after Hangul was adopted by Korea as its official writing system after the World War II. Despite its adoption more than half a century ago, Korean newspapers began to be printed in pure Hangul just in the recent years. Nevertheless, Chinese words may still be used in Korean text, as illustrated below:

(5) Chinese Words as Proper Part of Text

```
a. 한국은        아름답다
   hankuk-un     arumtapta
  ''Korea-TOPIC  is beautiful''
b. 韓國은        아름답다
```

Here the Chinese "韓國"(han.kuk) stands for "한국"(han.kuk). Since both have the same pronunciation "hankuk" with the final syllable ending in a consonant, they both take the topic particle "은"(un), instead of "는"(nun). Hence, the Chinese word "韓國"(han.kuk) is treated morphologically the same as the word "한국"(han.kuk) written in Hangul and part of the sentence. The string "은 아름답다"(un arum.tap.ta) with the omission of "韓國"(han.kuk) is totally meaningless, since it is the stem for the topic particle "은"(un) and, together with it, becomes the subject of the sentence. The preprocessing of Korean text must thus preserve Chinese characters as in the original text.

Words in Latin characters may also appear in ordinary Korean text. This is especially the case with foreign names and technical words which are hard to transliterate into Hangul.

(6) Hangul Text with Western Names

```
Original: a. Hausser 교수는 Erlangen에서 가르친다
             (Professor Hausser teaches in Erlangen)
Modified  b. 하우서  교수는 에어랑엔에서 가르친다
```

Here the original text contains two names "Hausser"and "Erlangen" in Latin characters as its proper parts. The word "Erlangen" like ordinary Korean nouns takes the particle "에서"(ey.se), meaning "in", and concatenates with it. Hence these foreign words written in Latin characters must again be preserved as in the original text.

3. Overall Architecture of the Preprocessor

This section will give an overview about the general architecture of the preprocessor. This architecture is designed to be language-independent and flexible. The main steps (Compare with figure 1) involved are:

A. Input filtering – Conversion of input text from any format to standardized preprocessing format

B. Core preprocessing – Involving general preprocessing steps like e.g. tokenizing or segmentation into sentences

C. Storage – Storing the preprocessed text or corpus using in an intermediate format from which all output-formats needed can be generated

D. Output filtering – Converting the stored corpus to produce the output needed (e.g. a word list or a sentence list)

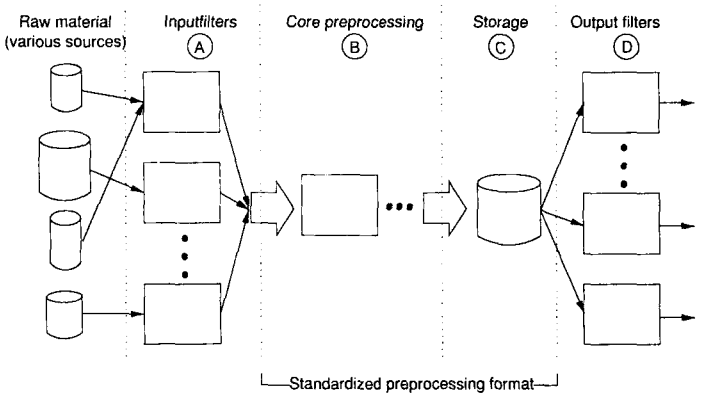


Figure 1
overall architecture of the preprocessor

3.1 Input Filtering

As discussed earlier, the raw material used to build up corpora will come from various sources and in various formats (e.g. HTML, TEI-format or even proprietary Markup of some text processor).

Source-specific input filters (A - in figure 1) are needed to bring the raw material into a *standardized preprocessing format* used throughout the subsequent steps B and C in Figure 1.

Several approaches to provide a standardized preprocessing format exist in principle. We favor the usage of SGML to formulate a Preprocessing Markup Language (PML)[7]. PML provides a small and simple set of markup-tags that can be used during preprocessing as well as the intermediate storage format for the corpus.

Using an SGML-language as the preprocessing format has several advantages. First of all, exchange of corpora with other institutions is very much simplified

7. Currently being developed with the aid of Christian Otto and Jai-Ho Suk.

by using SGML. Furthermore, there are various public domain SGML-Parsers capable of parsing any kind of SGML-document given the appropriate document type definition (DTD). In addition, programming tools for use with these SGML-parsers do exist for several languages e.g. for perl. These packages speed up the implementation of preprocessing modules significantly.

3.2 Core preprocessing

Core preprocessing comprises all tasks aside from erasing/converting unwanted original markup from input text. Typical preprocessing tasks are tokenizing, marking sentence ends, proper name identification, etc.

The modules needed and their implementation may vary from language to language. So the modules of the core preprocessing step may need rearrangement when switching to other languages in the present implementation. Some works like Palmer (1994) do show that adaptive, language independent approaches to certain tasks like sentence segmentation (period disambiguation) can be found. This way, future preprocessors might be just switched to other languages by loading configuration data generated beforehand by training.

3.3 Storage

As the desired format for the preprocessed corpus will vary with the intended application (such as lists of words or sentences for morphological or syntactical analysis, or an index for linguistic research on big corpora), it is not advisable to store the preprocessed corpus in any of these formats.

Instead, the corpus is stored in the application independent standardized preprocessing format. From this, any desired output format can be generated by the appropriate output filter in the subsequent step. Together with the PML-DTD, the storage format also provides a convenient way to exchange corpora with other researchers or institutions.

3.4 Output Filtering

Simple small output filters provide the means to convert the preprocessed corpus into any desired format, be it a word list, a unique word list together with an occurrence index of word-forms, a frequency list or any other format needed by a given application. This output can then, for instance, be further processed by Malaga-Analyzers such as Kormoran.

4. Building a Unique Word List and Index

An efficiently designed preprocessor for raw text must provide a compact format suitable for efficient access and searching algorithms. This format should, however, be general enough to easily accommodate any particular formats. An

application-oriented formatter should be designed as a separate module. It can thus convert a corpus from the storage format into any output in the general format from the preprocessor into any particular format required by a particular application. Malaga, for instance, requires a unique word-form list to execute the morphological analysis of a file. Hence the formatter should, for instance, be able to turn preprocessed Korean text into a unique list of Korean word-forms for Kormoran to run.

The process of building a unique list may take several steps. First, it can build a list of strings of symbols and characters by breaking a line at each space or each punctuation mark, thus creating a list of word-forms, as shown below:

(7) List of Symbols and Words

1	3	7	)	13	한다	20	그러
2	)	8	-	14	.	21	-
3	이른바	9	의	15	<	22	면
4	副動詞	10	어미들이	16	그러	23	>
5	(	11	그	17	-	24	등은
6	converb	12	구실을	18	나		
				19	,		

This list contains 21 strings, each of which is assigned a unique number for easy reference to its occurrence in the text. For building a simple list of strings or word-forms, we have introduced the following conventions:

(8) Conventions for Building String Lists

a. Treat each string separated by a space as a unit.
b. Treat each punctuation mark or bracket as a unit.

The list which is being compiled here mainly consists of word-forms. Since space generally marks off word-forms, (i) we first treat each string separated by a space as a unit in the list. (ii) We then treat each of the punctuation marks like the comma ",", period ".", and also each of brackets like parentheses "(" and ")" or angled brackets "<" and ">" as a separate unit, although they are always attached to other strings without any space between them, as in "(convention)". The hypeh "-" is not, however, separated from Hangul strings as in the concessive ending "-나"(na) meaning "but" or the conditional ending "-면"{myen}, for these endings are not complete word-forms by themselves, but only suffixal expressions. Likewise, stems like "그러-"(ku.re) meaning "such" also may retain the hyphen, for they again need be concatenated with suffixal expressions to form word-forms.

From this list, a sorted unique list can be built automatically. Before building such a list, however, we replace each of the Chinese characters in Korean text with a phonetically equivalent Hangul character because Malaga cannot treat Chinese characters at the present[8]

8. Although there is no wordprocessor or editor which has a program for converting Chinese

(9) Sorted Unique List or Word Index[9]

<	15	3	1	등은	24
>	23	converb	6	면	22
(	5	의	9	副動詞	4
)	2,7	구실을	12	어미들이	10
,	19	그	11	이른바	3
.	14	그러	16,20	한다	13
-	8,17,21	나	18		

The index thus created contains only one occurrence of each string of symbols and characters. Although this list does not show much reduction because there is one repeated co-occurrence, it can not only reduce the storage requirement of a large corpus, but also greatly speeds up its processing for corpus applications. Since it marks each item with its occurrences, it can also be converted back into the original text without any loss of its original information.

5. Morphological Analysis

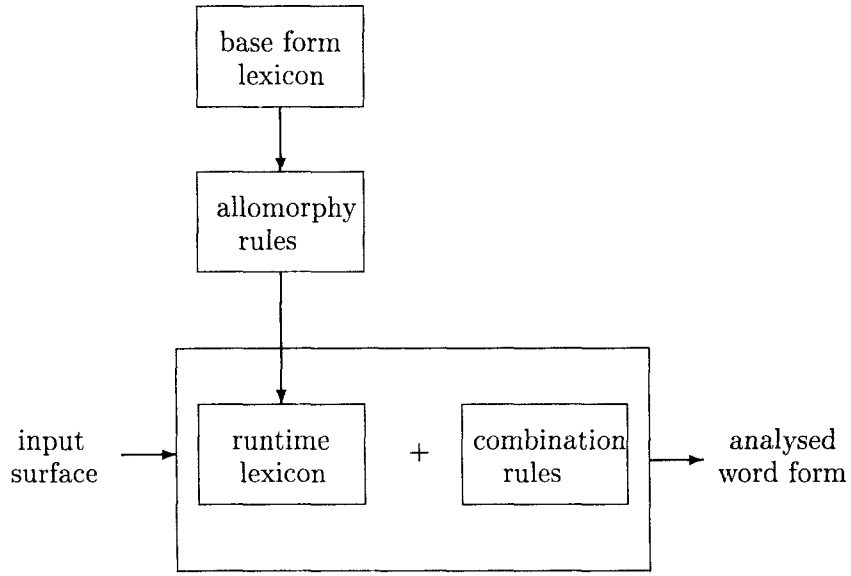
Like all other morphological analyzers developed with Malaga, Kormoran is a rule-based system consisting of two modules or sets of rules: (i) the allomorphy rules and (ii) the combination rules.[10] The first set of rules, namely the allomorphy rules in Kormoran, generate a runtime lexicon of allomorphs from the basic lexicon. Then, the second set of rules, namely the combination rules combine nominal or verbal stems with particles, endings, or suffixes to generate word-forms.

(10) Modules of Kormoran

characters into Hangul, one can easily build a program for such a conversion, for the process of conversion is a many-to-one function in most cases and, in other cases, can also be phonologically predictable. For example, 勞動 converts to 노동{no.tong}, while 勤勞 converts to 근로{kun.ro}. So the character 勞 which appears at the initial syllable changes to 노{no} because of the word-initial condition on liquid in Korean. On the other hand, it retains the original pronuncation 로ro in other places as in 근로{kun.ro}.

9. '副動詞' is sorted as if it were the same as its Hangul equivalent '부동사'.

10. Most of the ideas presented in this section are contained in Lee (1999). Since it is written in Korean, it is briefly outlined here to make the paper self-contained especially for non-Korean readers.



Being based on Hausser's (2001) Left-Associative Grammar, Kormoran strictly follows the principle of linearity, namely left-to-right analysis. But to overcome some of the shortcomings such as backtracking of this approach, Kormoran like other Left-Associative morphological systems introduces two types of lexicon: a basic lexicon and a runtime lexicon. A basic lexicon consists of a list of base forms with minimal information, whereas a runtime lexicon is derived by allomorphy rules from the basic lexicon that consists of allomorphs with fuller information necessary for morpho-syntactic combinations.

In a Malaga-based morphological analyzer, a runtime lexicon consisting of allomorphs is precompiled before combination rules start to apply. In English, for instance, two allomorphs "pretty" and "pretti" are generated from a basic lexicon and then precompiled and stored in the runtime lexicon to save time for word-form processing. Then to generate a word-form "prettier", the allomorph "pretti" is directly concatenated with the suffix "er". Likewise, in Korean, an adnominal word-form "추운"(chwu.wun) meaning 'cold' can be formed first by generating an allomorph "추우"(chwu.wu) from its base form "춥"(chwup) and then by combining it with the adnominal ending "ㄴ"(n). Such a linear processing eliminates any sideway process of changing "y" into "i" as in "pretty" : "pretti" or "wu" into "p" in chwup : chwu.wu, thereby avoiding the complexity of the morpheme method of word-form recognition.[11]

11. See Hausser (2001, section 13.5) for further discussions.

5.1 Rules for Generating Allomorphs

In Malaga, each lexical entry is represented as a record consisting of features, or attribute-value pairs. Here is an example in the base form lexicon of Kormoran:

(11) Lexcial Entry as a Feature Structure

```
[Phon: "춥", Class: adjective,
 Marked: p, Deriv: ha_nom,
 Sem: [Content: <"춥 다_cold">]]
```

This is an entry for the adjective stem "춥"(chwup). Since it is a semi-regular adjective ending in the consonant *p*, it has the feature "Marked: p". It can also form a *ha*-verb "추워하다"(chwue.ha.ta), so it has the derivational feature "Deriv: ha_nom".

Unlike regular lexical items, semi-regular or semi-irregular forms may have more than one form.[12]. For example, the semi-regular adjective "춥"(chup) meaning "cold", as shown earlier, has two allomorphs: "춥"(chwup) and "추우"(chwu.wu) Checking the special feature "Marked: p", one of the allomorphy rules **korean.all** generates these two forms, as shown below:

(12) Allormorphs Generated from the Base Form of (chwup)

```
1. "추 우": [Phon: "추 우",
             Segmentation: "춥",
             Cat: adj,
             SEM: [Content: <"춥 다_cold">],
             Baseform: "춥 다",
             Order: 0,
             Form: non_bse,
             Stem: "춥",
             Syllable: d_open,
             Bridge: _e&nil,
             Deriv: ha_verb,
             Val: <<nom>>]

2.  "춥": [Phon: "춥",
           Segmentation: "춥",
           Cat: adj,
           SEM: [Content: <"춥 다_cold">],
           Baseform: "춥 다",
           Order: 0,
           Form: bse,
```

12. These are two of the four classes of regularity discussed in Hausser (2001, section 14.1.7)

```
Stem: "춥",
Syllable: closed&dark,
Deriv: ha_verb,
Val: <<nom>>]
```

These two feature structures contain the same syntactic and semantic information, but differ their morphological information.

These morphological differences characterize the combinatorial behavior of the allomorphs. The allomorph "춥"(chwup), for instance, may combine with the clausal verbal ending "고"(ko), but not with the adnominal ending "ㄴ"(n). On the other hand, the form "추우"(chwu.wu) can combine with "ㄴ"(n), but not with "고"(ko).

Besides generating allomorphs, the so-called allomorphy rules assign various routine features to each lexical entry which are needed by the runtime lexicon. For example, there are two nominative particles "가"(ka) and "이"(i) in Korean: the first one is attached to nominal stems ending with an open syllable, while the second one is attached to nominal stems ending with a closed syllable. Hence, the syllable structure of nominal stems must be analyzed. The allomorphy rules exactly do such an analysis. Consider the following two entries for "교수"(kyo.swu) meaning "professor" and "학생"(hak.sayng) meaning "student" in the base form lexicon **korean.lex**:

(13) Base Forms

```
[Phon: "교수", Cat: noun,
 Sem: [Person: referent, Content: <"교수","professor">]];
[Phon: "학생", Cat: noun,
 Sem: [Person: referent, Content: <"학생","student">]];
```

These each contain only three attributes **Phon**, **Cat**, and **Sem** with their respective values.

But their entries in the newly generated runtime lexicon contain many other attributes and their values as shown in the following.

(14) Generated Allomorphs

```
malaga> ma 교수
Analyses of "교수":
1: [Phon: "교수",
    Segmentation: "교수",
    Cat: noun,
    SEM: [Content: <characterized_as, "교수", "professor">,
          Type_of_Obj: human],
    Baseform: "교수",
```

```
    Tagged: "고수_noun"]
malaga> ma 학생
Analyses of "학생":
1: [Phon: "학생",
    Segmentation: "학생",
    Cat: noun,
    SEM: [Content: <characterized_as, "학생", "student">,
          Type_of_Obj: human],
    Baseform: "학생",
    Tagged: "학생_noun"]
```

The base form lexicon is manually compiled, while the allomorph lexicon is automatically generated. Hence, all the features which can be automatically assigned are specified in the allomorph or runtime lexicon.

5.2 Combination Rules

Once an adequately specified runtime lexicon with allomorphs is generated, the process of combining stems with their appropriate suffixes is straightforward in Kormoran.

This combination process may be constrained by adopting the categorial rule of the following form:

(15) Categorial Rule

If α is of category A and β of category B/A, then f(α,β) is of category B, where f is a syntactic operation like concatenation.

Kormoran treats categories as bundles of features and function words like verbal endings as having the attribute **Requires** and **Results** which respectively stand for A and B of a functor B/A. For example, the verbal endings "고"(ko) and "ㄴ"(n) are specified as below:

(16) Verbal Endings as Functors

```
[Phon:"고",
 Cat:ending,
 Order:<48>,
 Requires:[Cat:verbal,
       Form: bse],
 Results:[Form:w_terminal,
      Bridge:nil,
      VForm: conj&prog]];
```

```
[Phon:"ㄴ",
 Cat:ending,
 Order:<48>,
 Bridge: u_bridged,
 Sem: [Tense: overlapping],
 Requires:[Cat:adj,
       Form:bse&non_bse,
       Syllable:d_open&l_open&open],
 Results:[Cat:adnoun,
      Form:w_terminal,
      vform: modifier]];
```

The combination rule then simply states that stems can concatenate with these endings, if they satisfy the features in the attribute **Requires**.

If such a matching condition is satisfied, then the combination rule will yield the following results:

(17) Concatenations

```
malaga> ma 춥ㄱ
Analysis of "춥ㄱ": 1: [Phon: "춥ㄱ",
    Segmentation: "춥/ㄱ",
    Cat: adj,
    SEM: [Content: <"춥ㄷㅏ_cold">],
    Baseform: "춥ㄷㅏ",
    Val: <<nom>>,
    VForm: conj&prog,
    Tagged: "춥_adj/ㄱ_ending"]
malaga> ma 추운 Analysis of "추운": 1: [Phon: "추운",
    Segmentation: "춥/ㄴ",
    Cat: adnoun,
    SEM: [Content: <"춥ㄷㅏ_cold">,
          Tense: overlapping],
    Baseform: "춥ㄷㅏ",
    Val: <<nom>>,
    VForm: modifier,
    Tagged: "춥_adj/ㄴ_ending"]
```

If necessary, one can check the step-by-step process of concatenation in Malaga by invoking the command like **tree** in the malaga mode.

Here is an analysis tree for "추운"(chwu.wun):

Figure 2 shows that the stem "추우"{chwu.wu} concatenates with the ending "ㄴ"(n). The concatenation here has gone through straightforwardly left-to-right.

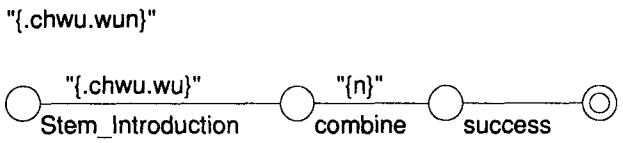


Figure 2
Analysis Tree for "추운"(chwu.wun)

Here no sidetracking or backtracking has occurred to replace the final consonant "ㅂ(p) in the base form "춥"(chwup) with the syllable "우"(wu), for, instead of the baseform, its allomorphic form "추우"(chwu.wu) is introduced as the input stem.

5.3 Processing word-form Lists

Malaga analyzes not only each individual word-form, but lists of word-forms of any length. Here is an example for the result of analyzing a sample list of three word-forms that are aligned in a single column.

(18) The Result of Analyzing A Sample List of Word-forms

```
1: "미아가": [Phon: "미아가", Segmentation: "미아/가",
              Cat: noun,
              SEM: [REF: <named, "미아_Mia">,
                    Type_of_Obj: human],
              Baseform: "미아", Case: nom,
              Tagged: "미아_noun/가_particle"]
2: "사과를": [Phon: "사과를", Segmentation: "사과/를",
              Cat: noun,
              SEM: [REF: <characterized_as, "사과", "apple">,
                    Type_of_Obj: fruit],
              Baseform: "사과", Case: acc,
              Tagged: "사과_noun/를_particle"]
3: "먹었다": [Phon: "?禿駄?", Segmentation: "먹/어/ㅆ/다",
              Cat: verb,
              SEM: [REL: <action, "먹다_eat">,
                    ARG: <[parameter: 1, Role: agent, Case: nom],
                          [parameter: 2, Role: theme, Case: acc]>,
                    SLv: plain, Tense: past],
              Baseform: "먹다", STyp: declarative, VForm: terminal,
              Tagged: "먹_verb/어_Bridge/ㅆ_ending/다_ending"]
```

For each total analysis, Malaga also provides some statistical information. For the sake of illustration, one unknown word-form and one ambiguous word-form are included in the test list of five word-forms.

(19) Statistical Information

```
malaga> ma-file test_list
Analyzed word-forms:       5
Recognized:                4 (80.00%)
Recognized by combi rules: 4 (80.00%)
Results per word-form:      1.25
```

The first line indicates the total number of word-forms analyzed, the second and third lines that of word-forms recognized, and the final line the results per word-form. As expected, the above result shows there is one word-form unknown and one word-form having two analyses. The cases of ambiguity will be presently dealt with.

6. Producing Parts-of-Speech Tagged Text

For some application such as spelling check or noun extraction, tagging text solely with parts of speech may be sufficient. In such a case, one can use the results of morphological analysis by Kormoran to tag each word-form in the preprocessed text with an appropriate category or part of speech.

6.1 Tagging Only

This process can easily be carried on Kormoran by activating an output filter rule, as shown below:

(20) Linearly Represented Tagging

```
1: "ᄆᆡ아가": "ᄆᆡ아_noun/가_particle"
2: "사과를": "사과_noun/를_particle"
3: "먹었다": "먹_verb/어_Bridge/ᄊᆞ_ending/다_ending"
```

By automatically editing the above result, we can also get the follwoing format with a new set of abbreviated tag names.

(21) Linearly Represented Tagging

```
ᄆᆡ아_n/가_p 사과_n/를_p 먹_v/어/ᄊᆞ_e/다_e
```

A linear representation with abbreviated tag names saves space and is useful for dealing with larger corpora.[13]

13. In its future version, Kormoran may easily be able to accommodate a standardized tag set for Korean text into its symbol set file *korean.sym* that lists all the category necessary names and other symbols including multi-symbols.

6.2 Treating Ambiguities

The resolution of ambiguities is a crucial issue in morphological analysis as well as in parsing. One of the main purposes of tagging is to resolve ambiguities. It can disambiguate by segmentation, by tracing base forms or by assigning appropriate POS names or their subcategories. For instance, the word form "가시는"(ka.si.nun) can be segmented into either "가/시/는" or "가시/는". The stem "가"(ka) in the segmented form "가/시/는" may have two different base forms: "가"(ka) and "갈"(kal). These base forms then can be tagged with two different POS names, namely *verb* and *adjective*.

Kormoran can properly handle such functions by yielding multiple tagging of the same surface forms, as shown below:

(22) Multiple Tagging

```
1: "가는": "가_verb/는_ending"

1: "가는": "갈_verb/는_ending"

1: "가는": "가늘_adj/ㄴ_ending"

2: "가시는": "가_verb/시_ending/는_ending"

2: "가시는": "갈_verb/시_ending/는_ending"

2: "가시는": "가시_noun/는_particle"

3: "걸었다": "걸_verb/었_Bridge/ㅆ_ending/다_ending"

3: "걸었다": "걸_verb/었_Bridge/ㅆ_ending/다_ending"

3: "걸었다": "걸_verb/었_Bridge/ㅆ_ending/다_ending"

3: "걸었다": "걸_verb/었_Bridge/ㅆ_ending/다_ending"

3: "걸었다": "걷_verb/었_Bridge/ㅆ_ending/다_ending"
```

The above analysis shows that many of the word forms may result in many different forms of tagging. The word form "가는"(ka.nun), for instance, has three different ways of tagging: the first two word forms are of the category verb, while the third one is of the category adjective.[14] The two verbs, however, have

14. The verb "갈"(kal) has several different senses: it can, for instance, mean 'to grind (flour)', 'to change (personnel)', or 'to plow (a field)'. With these different senses, the analysis of the word form "가는"(ka.nun) results in more different tagged representations.

different base forms: the first one is "가"(ka) and the second one, "갈"(kal). The other word forms like "가시는"(ka.si.nun) and "걸었다"(kel.ess.ta) again allow many different ways of tagging. The word form "가시는"(ka.si.nun) has three different tagging forms and the the word form "걸었다"(kel.ess.ta), at least five different tagging forms.

The ambiguities involving these word forms cannot be sufficiently resolved by segmentation and POS-tagging only. There are, however, two possible ways to resolve them further at this level of morphological analysis. One commonly accepted way is to link a morphological analyzer or tagger to well-balanced very large corpora and then to provide each analyzed form with statistical information, namely a confidence rate or weight, according to the frequency of its occurrence in the given corpora. Kormoran is theoretically designed to perform such a function by linking itself to a corpus and also to a very large lexicon generated from a small basic lexicon that lists all the possible allomorphic variants and derivational forms with fuller morpho-syntactic and semantic information.

(23) Confidence Rate

```
1: "가는": "가_verb/는_ending"_5.0

1: "가는": "갈_verb/는_ending"_3.3

1: "가는": "가늘_adj/ㄴ_ending"_2.7
```

The second way of resolving ambiguities is to provide each analyzed item with fuller information. Such information is provided in Kormoran by assigning a feature structure to each analyzed item that represents not only morpho-syntactic information, but also semantic content. In other words, Kormoran can either switch on or off an output filter that allows only tagged representations or fuller information in feature structures.

By switching on the output filter, for instance, Kormoran has just produced five different tagged forms of one surface form "걸었다"(kel.ess.ta). Their stems are all of the category *verb* and, out of the five, all the four stems have the base forms "걸"(kel), while the only one has the base form "걷"(ket). Since these four stems have the same base form of the same category, they need to be further analyzed. But since their differences are due to their different senses, their ambiguity cannot be resolved by tagging only. Kormoran with its output filter switched off, however, shows fuller information in feature structure. The following is an example:

(24) Analysis of "걸었다"(kel.ess.ta)

```
3: "걸었다": [Phon: "걸었다", Segmentation: "걸/어/쓰/다",
              Cat: verb,
```

```
                SEM: [Content: <"(옷을) 걸다_hang">,
                              SLv: plain, Tense: past],
                Baseform: "걸다", Val: <<nom, acc>>,
                STyp: declarative, VForm: terminal,
                Tagged: "걸_verb/어_Bridge/써_ending/다_ending"]

3: "걸었다": [Phon: "걸었다", Segmentation: "걸/어/써/다",
                Cat: verb,
                SEM: [Content: <"(발을) 걸다_trap">,
                              SLv: plain, Tense: past],
                Baseform: "걸다", Val: <<nom, acc>>,
                STyp: declarative, VForm: terminal,
                Tagged: "걸_verb/어_Bridge/써_ending/다_ending"]

3: "걸었다": [Phon: "걸었다", Segmentation: "걸/어/써/다",
                Cat: verb,
                SEM: [Content: <"(내기를) 걸다_bet">,
                              SLv: plain, Tense: past],
                Baseform: "걸다", Val: <<nom, acc>>,
                STyp: declarative, VForm: terminal,
                Tagged: "걸_verb/어_Bridge/써_ending/다_ending"]

3: "걸었다": [Phon: "걸었다", Segmentation: "걸/어/써/다",
                Cat: verb,
                SEM: [Content: <"(전화를) 걸다_ring">,
                              SLv: plain, Tense: past],
                Baseform: "걸다", Val: <<nom, acc>>,
                STyp: declarative, VForm: terminal,
                Tagged: "걸_verb/어_Bridge/써_ending/다_ending"]

3: "걸었다": [Phon: "걸었다", Segmentation: "걸/어/써/다",
                Cat: verb, SEM: [Content: <"걸다_walk">,
                               SLv: plain, Tense: past],
                Baseform: "걸다", Val: <<nom>>, STyp: declarative,
                VForm: terminal,
                Tagged: "걸_verb/어_Bridge/써_ending/다_ending"]
```

note again that the first four analyses have the same base forms but with different senses. The last one only has a different base form "걷"(ket, 'to walk'), one of the *t*-type semi-regular verbs, that has the allomorphic form "걸"(kel).

7. Storing Tagged Corpora

As shown in section 4 a corpus can be stored in form of a list of unique word-forms with an index – a so-called *inverted corpus*. This can also be done for a tagged (or morphologically analyzed) corpus as shown by (Künneth 2001). Using an inverted corpus serves two main purposes: First, it greatly compresses the size of the corpus, with higher compression rates for larger corpora. Secondly the access time for searches by word form, POS-tag, stem, base form or the like is greatly increased.[15]

Furthermore this method of storing an analyzed corpus makes it possible to use an off-the-shelf relational database as the basis for storage and retrieval (also see Künneth 2001), thereby eliminating the need to develop a proprietary solution.

8. Application

A tagged corpus stored in a data format that allows efficient search and access methods can be used for a variety of purposes both academic and commercial as described further below. All the applications described there rely on (some of) the following key-features a full morphologic analysis provides:

8.1 Segmentation

Segmentation is the first step of morphological analysis. It splits the input surface into allomorphs that are then further analyzed. It not only plays a crucial role in analysis, it also allows to break up composite word-forms into its different stems.

These can then be searched for: A search for e.g. the stem "집"(cip, 'house') will also find the composite forms "술/집"(swul.cip,'drinking house') or "기와/집"(ki.wa.cip, 'tile-roofed house'), but not any word-forms which contain the sequence "집"(cip) as part of their surface but not as a stem –as e.g. "고집"(ko.cip,'obstinacy') or "계집"(kyey.cip), an derogatory term for 'woman'. As illustrated by the last example, this search for stems has a higher precision than a simple substring search on an un-analyzed corpus.

8.2 Lemmatization

Lemmatization or reduction to base forms as part of morphological analysis find the base form to an input word-form, also reducing allomorphs to their stems. This allows easy searching for all inflectional forms of a stem (or word) with a higher recall than a string search on an un-analyzed corpus or a stemmer.[16] If the corpus for instance contains the adnominal form "아름다운"(arum.ta.wu/n,

15. See (Künneth 2001) for a discussion of the access times.
16. The term *stemmer* refers to a method of reducing word-forms to their base forms that is based on heuristics in contrast to the full model of a rule-based morphology.

'beautiful'), the conjunctive adjectival form "아름답고"(arum.tap.ko,'be beautiful and') and the declarative terminal adjective and "아름다웠다"(arum.ta.wu/ess.ta 'was/were beautiful'), then they will all be recognized as forms of the lemma "아름답다"(arum.tap.ta,'to be beautiful').

8.3 Categorization

Categorization as the final output of morphological analysis is especially useful to retrieve not only isolated word-forms, but word-forms occurring in a specified context or syntactic structure, which is virtually impossible in an un-analyzed corpus.

8.4 Full-text information retrieval

For any kind of large scale full-text retrieval system as the world wide web or a large database of e.g. medical articles, morphological analysis can be applied for several purposes:

Improving recall and precision: As Piotrowski (1998) showed, lemmatization and decomposition improve the precision of full-text retrieval even at high recall levels.

Reducing the index: Using base forms instead of full forms for building the search index results in a smaller and thus more efficient index.

8.5 Corpus-based linguistic research

There are at least two possible areas in corpus-based linguistic research:

Finding real-world examples: The monolingual Collin's *COBUILD English Dictionary* has been the first dictionary to use only examples taken from real-world language data. Without an analyzed corpus it would be virtually impossible to find examples for words in given contexts in reasonable time.

Exploring language-data: Corpus-research has become more and more widespread in linguistics during the last decade: Research of vocabulary as well as of syntactic constructions both require a detailed morphologic analysis.

9. Evaluation

Kormoran was not origianally built for the purpose of POS-tagging. It was primarily built as a morphological analyzer that generates a lexical basis which provides all the necessary morpho-syntactic information for syntactic as well as semantic processing. Most of the preexisting so-called morphological analyzers

for Korean are POS-taggers that provide information on segmentation and POS-classification without being able to provide complex information that are necessary for syntactic or semantic processing. Hence, the main purpose of this paper must be understood as showing how the complex rule-based system Kormoran that was primarily built for syntactic and semantic processing can also be simplified into a POS-tagger. If this perspective is lost, then Kormoran as described here has very little value.

Unlike most of the taggers, the construction of Kormoran as a rule-based analyzer is not based on corpora or any raw data. It is rather based on preestablished Korean morphological descriptions as well as preexisting dictionaries like those compiled by Keum-Seong or the National Korean Language Research Center. It accommodates not only the so-called inflectional variations dealing with regular, semi-regular, semi-irregular or irregular allmorphs, but also with derivational forms like the derivation of "ha"-verbs from nouns or adjectives.[17] But to test its efficiency and other requirements of adequacy, the use of Kormoran should be subjected to very large corpora. This task is, however, left as part of the future work.

10. Conclusion

A variety of efficient techniques for tagging Korean corpora have already been developed and are in good use for many practical applications.[18] Nevertheless, some tasks require information that only full, rule-based morphological analysis can give.[19] One prominent example is the improvement of both precision and recall for full-text search in large textual databases as in the world wide web. Apart from this technological use, full morphological analysis also forms the basis for rule-based syntactic and semantic analysis, providing further grammatical insight into language and its use which in turn will lead to further technological development. The morphological analyzer Kormoran presented in this paper provides the basis for both technological applications and further linguistic research by allowing to systematically analyze, store and access large corpora.

17. In the current version of Kormoran, all the inflectional cases are accommodated, but many of the derivational forms are left untreated because these forms fail to be described without regular distributions.
18. See Kim and Kang (1996) for their discussion of techniques of building Korean corpus and also Yoon and Choi (1999) for their method of POS-tagging Korean corpus based on a morphological analyzer. Besides them, most of Korean computational linguistics or specialists in natural language processing devised their own rule-based, statistical or hybrid techniques of building Korean corpora and tagging them. But none has aimed at building a comprehensive system that can cover from morphology to syntax to semantics.
19. Feng (2001) proposes hybrid (rule-based and statistical) approaches for automatic segmentation and annotation of Chinese corpus. But his rules are not linguistically motivated morphological rules: He uses statistically trained rules for transforming bracket chunking based on the Brill tagger.

References

Beutel, Björn. 2002. A Manual for Malaga 6.7. Department of Computational Linguistics, Friedrich-Alexander University of Erlangen-Nuremberg.

Brill, Eric 1993. Automatic Grammar Induction and Parsing Free Text: A Transformation-Based Approach. *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics.*

Feng, Zhiwei. 2001. Hybrid Approaches for Automatic Segmentation and Annotation of Chinese Text Corpus. *International Journal of Corpus Linguistics*, 6 (Special issue) [in print] (John Benjamin Publishing Co. Amsterdam, The Netherlands).

Hausser, Roland. 2001. *Foundations of Computational Linguistics: Human-Computer Communication in Natural Language* 2nd edition, revised and extended. Springer, Berlin.

Kim, Heung-Gyu and Beom-Mo Kang. 1996. Korean-1 Corpus: Design and Composition [written in Korean]. *Korean Linguistics* 3, 233–258.

Kim, Young-Taek, et al. 2001. *Natural Language Processing.* Life and Power Press, Seoul.

Künneth, Thomas. 2001. Storing Annoatated Corpora in a Relational Database Management System. *Proceedings of the COMPLEX 2001 6th Conference on Computational Lexicography and Corpus Research*, 95-102. University of Birmingham.

Lee, Kiyong. 1999. *Korean Computational Morphology*[written in Korean]. Korea University Press, Seoul.

Lee, Kiyong. 2002. *Kormoran-6.6.* Department of Computational Linguistics, Friedrich-Alexander University of Erlangen-Nuremberg.

Palmer, David D. 1994. *SATZ - An Adaptive Sentence Segmentation System*, Report No. UCB/CSD-94-846. Computer Science Division University of California, Berkeley.

Piotrowski, Michael. 1998. NLP-Supported Full-Text Retrieval. *CLUE-Technical Reports* No. 3, Department of Computational Linguistics, Friedrich-Alexander University of Erlangen-Nuremberg.

Yoon, Jun-Tai and Key-Sun Choi. 1999. Study on POS tagged Corpus for Korean. CS-TR-99-138 KORTERM-TR-99-01. KAIST, Daejoen, Korea.

Submitted on: October 21, 2002
Accepted on: November 20, 2002