Korean Verbal Morphological Analyzer

by

Eric H. Davis
11-712 Natural Language Processing Lab
Carnegie Mellon University
Language Technologies Institute
Professors Shuly Wintner & Kemal Oflazer
May 7, 2007
1. Introduction

The goal of this term project was to build a morphological analyzer for Korean verbs with the best coverage possible given the time constraints of a semester. I used the Xerox Finite State toolkit to design a transducer that analyzed Korean verbal morphology. I will begin by providing some background on the Korean language. Then, I will give a brief overview of Korean morphology. Next, I will discuss the key components of the system, including the Romanization system, the re-write rules, and the ASCII to Korean syllable mappings. After that, I will exemplify some issues with Korean morphology and how this system handles those issues. Finally, I will discuss the impact of this work, what I have learned, and any future direction to be taken.

1.2 Korean Background

Linguists have classified Korean as a member of the Altaic language family, which groups it with Turkish and Mongolian. King Sejong invented the writing system in the 15th Century, largely to address the large segment of the population that could not read or write due to the complicated nature of the Chinese characters in use at that time. The writing system is based on syllables, and each syllable must have at least one consonant and a vowel. Most textbooks will claim that Korean has 10 vowels and 14 consonants. These consonants and vowels can be combined systematically in numerous ways into syllables producing an estimated 11,000 possible syllables. Not all combinations are possible and constraints apply. For example, it is not possible for consonants to cluster or liquids to occur in the word-initial position. The system, called Hangul, is simple yet comprehensive and caught on quickly. This systematic nature makes Hangul one of the most scientific writing systems in the world (http://www.korea.net/korea/kor_loca.asp?code=A020301).

1.3 Korean Morphology and Phonology

Korean like many other Altaic languages possesses a high degree of vowel harmony. The rules for vowel harmony were rigidly followed in old Korean. However, the rules are significantly more lax in modern Korean and many contexts where vowel harmony would have been mandatory in old Korean are now optional in modern Korean (Lee 5-12). For example, 'thank' has two distinct, valid forms: one with vowel harmony (komawa) and one without (komaweo) (Romanization will be discussed below). The former exhibits vowel harmony, and the 'a' in the stem harmonized with the 'a' in the suffix. The latter, on the other hand, does not. It is interesting to note, that in modern Korean, the latter form is much more commonplace. Nevertheless, vowel harmony is still an integral part of the onomatopoeic words that are quite plentiful in Korean (http://www.korea.net/korea/kor_loca.asp?code=A020301).

Korean, like Turkish, is an agglutinative language that adds derivational and inflectional suffixes to nominal and verbal stems. It is possible to "stack" numerous suffixes on a single stem, and suffixes join together "like beads on a string." Suffixes agglutinate one after another and indicate different styles of speech, different moods, aspects, tense, conjunctions, and various other functions. Despite this rich system of suffixes, verbs do not inflect for number (at least verbs), gender, or agreement, and only mood is truly mandatory. A verb stem cannot stand alone and must have at least one suffix (Lee 12-18).

The typical characterization of Korean morphology is that of a template with slots for one of five possible suffixes:

Stem (Suff1) (Suff2) (Suff3) Suff4 (Suff5)
The stem and suffix number four are mandatory, but everything else is optional. Suffix one represents the causative or passive and must occur before the second, third, fourth, and fifth suffixes. Suffix two represents the honorific and must occur adjacent to the stem or possibly immediately after the causative or passive. The third suffix is tense and may or may not be a part of a fully-inflected verb. The fourth suffix is mood and represents the only mandatory suffix. The final optional suffix is continuation and can be a conjunction or reason, or any indication that the verb form is not final (http://web.khu.ac.kr/~jongbok/research/final-papers/projecdtion.pdf). More details on all of the above will occur below. However, this project did not include continuation because it only focused on final endings. Suffix five exhibits similar phonological and morphological alternations to the other suffixes and would thus not add anything too interesting to the project other than more endings and more forms.

2. The System

The first step was to find a systematic Romanization system for Korean. As mentioned, Korean is syllable based, but the syllable does not necessarily have to have a left-to-right reading. Certain consonants and vowels occur under other consonants and vowels. I am much more familiar with an ASCII representation and left-to-right ordering of characters and chose to Romanize the Hangul characters to make writing the rules easier and more intuitive. After a bit of research, I decided to use a modified Yale system for Korean. This system is popular in the United States and other Western countries, but it is a bit outdated. I kept most of the romanizations the same, but I chose to update several to make them closer to the sounds that they truly represent and coincide better with the system that I learned in Korea (http://www.glossika.com/en/dict/Korpin.pdf).

<table>
<thead>
<tr>
<th>Korean</th>
<th>Yale</th>
<th>Me</th>
<th>Korean</th>
<th>Yale</th>
<th>Me</th>
</tr>
</thead>
<tbody>
<tr>
<td>가</td>
<td>k</td>
<td>k</td>
<td>ㅏ</td>
<td>e</td>
<td>eo</td>
</tr>
<tr>
<td>고</td>
<td>kk</td>
<td>kk</td>
<td>ㅗ</td>
<td>o</td>
<td>o</td>
</tr>
<tr>
<td>나</td>
<td>n</td>
<td>n</td>
<td>ㅜ</td>
<td>wu</td>
<td>u</td>
</tr>
<tr>
<td>도</td>
<td>t</td>
<td>t</td>
<td>ㅡ</td>
<td>u</td>
<td>eu</td>
</tr>
<tr>
<td>떨</td>
<td>tt</td>
<td>tt</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>련</td>
<td>l</td>
<td>l</td>
<td>ㅐ</td>
<td>ay</td>
<td>ae</td>
</tr>
<tr>
<td>렴</td>
<td>m</td>
<td>m</td>
<td>ㅔ</td>
<td>ey</td>
<td>e</td>
</tr>
<tr>
<td>렺</td>
<td>p</td>
<td>p</td>
<td>ㅟ</td>
<td>oy</td>
<td>oi</td>
</tr>
<tr>
<td>령</td>
<td>pp</td>
<td>pp</td>
<td>ㅢ</td>
<td>wi</td>
<td>wi</td>
</tr>
<tr>
<td>렼</td>
<td>s</td>
<td>s</td>
<td>ㅣ</td>
<td>ya</td>
<td>ya</td>
</tr>
<tr>
<td>렽</td>
<td>ss</td>
<td>ss</td>
<td>ㅑ</td>
<td>ye</td>
<td>yeo</td>
</tr>
<tr>
<td>렶</td>
<td>ng</td>
<td>ng</td>
<td>ㅢ</td>
<td>yo</td>
<td>yo</td>
</tr>
<tr>
<td>렷</td>
<td>c</td>
<td>j</td>
<td>ㅠ</td>
<td>yu</td>
<td>yu</td>
</tr>
<tr>
<td>렸</td>
<td>cc</td>
<td>jj</td>
<td>ㅐ</td>
<td>yay</td>
<td>yae</td>
</tr>
<tr>
<td>령</td>
<td>ch</td>
<td>ch</td>
<td>ㅐ</td>
<td>yey</td>
<td>ye</td>
</tr>
<tr>
<td>렻</td>
<td>kh</td>
<td>kh</td>
<td>ㅏ</td>
<td>wa</td>
<td>wa</td>
</tr>
<tr>
<td>렼</td>
<td>th</td>
<td>th</td>
<td>ㅢ</td>
<td>way</td>
<td>wae</td>
</tr>
<tr>
<td>렻</td>
<td>ph</td>
<td>ph</td>
<td>ㅢ</td>
<td>we</td>
<td>weo</td>
</tr>
<tr>
<td>령</td>
<td>h</td>
<td>h</td>
<td>ㅢ</td>
<td>wey</td>
<td>we</td>
</tr>
<tr>
<td>렺</td>
<td>a</td>
<td>a</td>
<td>ㅣ</td>
<td>uy</td>
<td>eui</td>
</tr>
</tbody>
</table>

This system has a one-to-one mapping from ASCII to Korean. The one exception is 'ng,' which represents to distinct consonants in Korean. At the start of a syllable, 'ng' marks the syllable begins
with a vowel and only exists to maintain the rule that all syllables must have at least one consonant. On the other hand, 'ng' at the end of a syllable is equivalent to a the 'ng' in 'sing.' To maintain this distinction and allow for transliteration from ASCII to Korean, I use 'ng' in both syllable initial and syllable final positions; even if it is just marking a syllable that begins with a vowel. I found that the typical 10 vowels and 14 consonants did not provide a fine enough distinction, so my system has 13 vowels and 19 consonants. The extra consonants come from including aspirated consonants and 'ng.' 'ng' is ambiguous and not typically included as a consonant, but it acts like a consonant in its place-holder role and is a consonant as 'ng,' so I included it as a consonant. Many Korean texts do not consider these base consonants (typically non-aspirated consonants). However, aspirated consonants occur on the surface and there are minimal pairs distinguishing them from base consonants (such as thang 'land' and tang 'sugar'), so I chose to include them as consonants.

Abstract consonants also play a key role in the system. Such consonants have a certain base underlying form and varying surface forms. First, ^P can become 'w,' 'u,' or 'p' on the surface depending on the context. Next, ^T can become 'l' or 't.' Moreover, ^S is either deleted or becomes 's,' and ^H is either deleted, becomes 'ae' or 'h.' Finally, ^L is deleted or becomes 'l,' and ^R geminates into 'll' or becomes 'l.'

Similar reasoning can be employed to explain the extra vowels. Base vowels can be combined to form diphthongs. These diphthongs play a key role in my system, but no re-write rule relies on them, and they can be derived from other vowels, so there was no need to encode them as vowels. Usually, 'ya,' 'yoe,' and 'ye' are not considered base vowels either. However, these vowels cannot be derived from other vowels, so I chose to encode them as base vowels as well.

The system also makes use of one abstract vowel. ^E has two surface forms: 'eo' and 'a.' 'eo' is the base form, and 'a' is derived from vowel harmony rules.

As mentioned, Hangul is based on syllables. I thus, needed a syllable boundary marker and chose '|' to mark syllable boundaries. This symbol was easier to see than the typical '.', and would allow the later transliteration of the output of my system from ASCII to Hangul. The symbol boundary also made the system output easier to read and divided a word into easy to read syllabified phonological units.

### 2.1 Stems and Endings

I chose 20 representative Korean verbs that would exhibit interesting phenomena. The 20 forms can be divided into main verbs, light verbs, and stative verbs. The labels Main and Light distinguish verb stems with identical forms such as 'ha,' which can be both main and light verbs. Main verbs license all endings but may not license the passive. The one light verb, 'ha' is a part of the a causative suffix, cannot stand alone, and inflects to a certain extent. Certain suffixes, such as imperative and propositive, are not possible, but the light verb inflects for tense and the other moods. Finally, the stative verbs behave more like adjectives and do not license the imperative or propositive suffixes.

#### 2.1.2 Suffix 1

I followed the template from above and defined 4 suffixes, excluding the continuation suffix for reasons listed above. The first suffix is the optional causative and passive. As with most if not all languages, the two cannot co-occur. The passive has five possible suffixes, whereas the causative has three possible suffixes. It is not possible to derive from rules which passive or causative applies to
which verb, so I created dummy variables to link certain passive and causative suffixes to certain verbs. For example, the definition for 'ka' is k a [ %+NoPass | %+CausIII ] %| | #go. This says that 'ka' does not take a passive suffix and takes the third type of causative. Then the definition for CausIII is as as follows: [ %+CausIII .x. k e %| h a %+Light %| ]. The system then matches the dummy variable attached to the stem 'ka' to the label 'CausIII' in the definition of Suffix one ensuring that the proper causative suffix attaches to 'ka.' The other examples for causative and passive follow the same process. The dummy variable and labels in Suff1 do not ensure that the correct suffix attaches to the correct verb, and the process to filter out invalid input will be discussed below.

2.1.3 Suffix 2

Suffix two is the optional honorific. This suffix must come either adjacent to the stem or immediately after the passive or causative suffix. The honorific has two forms on the surface 'ngeu|si' and 'si.' I chose 'ngeusi' as the underlying form, as it was easier to create rules that deleted the syllable before '|' (the 'ng eu') then to derive it via rules.

2.1.3 Suffix 3

Suffix three is the optional tense marker and has four possible endings. This suffix must come after the preceding two suffixes if they exist or can come next to the stem if neither suffix one nor suffix two exist. The system has four tenses: past, past perfect, future, and present. Present tense is severely limited as it is implied in many of the suffixes in suffix four. As such, present tense can only co-occur with Decl Low from suffix four. Using the abstract vowel was necessary for both the past tense suffix and past perfect tense suffix. The underlying vowel in these suffixes alternates based on context and has two surface forms 'a' and 'eo.'

2.1.4 Suffix 4

Suffix four, mood, is the only mandatory suffix and has 34 different endings. No verb form is valid if one of the suffixes from suffix 4 is not attached to the verb stem. Korean society is based on respect and a hierarchical ordering, and this system is encoded in the inflections. Each mood has four different levels of respect depending on who the speaker is addressing. Low is the least formal and would be used if an adult addressed a child. Middle is still quite informal and would be used between two friends. Moreover, High is formal and would be used between two acquaintances that respect each other. Finally, Highest is the most formal and would be used by a student addressing a teacher.

There are five different moods representing various types of utterances. Foremost, Decl represents a typical declarative utterance. Next interrogative represents a question. Then, Imper is the imperative form such as 'do the laundry.' Also, Propos represents propositive, where one speaker suggests an action to another speaker. This is equivalent to the 'let's...' in English. Finally, Exclamatory is used when the speaker is excited and wants to share his or her excitement with someone else.

2.2 Filters

The above templatic approach greatly over-generates. To resolve this issue, I defined 24 filters that constrain what can co-occur with what, what can come before and after what, and what verb forms license what endings. For instance, define Filter1 [ ∼ $[ %+Pres ] | $[ %+Pres %+Decl %+Low ] ] maintains that the present tense cannot occur alone and if it does occur must be followed immediately
by the declarative low mood suffix. The passive and causative also over-generated because the definitions mentioned above did not have any constraints. That meant that although it was not valid, a verb with the NoPass dummy variable could combine with PassI, PassII or any of the passive suffixes. I have 12 filters to handle any combination of passive and causative suffixes and only allow valid combinations. define Filter12 [ \$[ %+NoCaus %| %+CausI | %+PassI | %+CausII | %+PassII | %+CausIII | %+PassIII | %+PassIV | %+PassV | %+NoPass ] ] -> [~[?*] ]] maps any form that contains a NoPass dummy variable and is followed by any passive or causative to the null language. The imperative and propositive moods cannot follow the passive or causative suffix either. I defined another filter to resolve this issue. Finally, certain verbs do not license certain suffixes. An example is the stative verbs which do not allow imperative or propositive endings. The filter, define Filter18 [ \$[ m eo %^L ?* [ %+Pres | %+Propos | %+Imper ] ] -> [~[?*] ] ], filters out such possibilities. Filtering greatly reduces the size of the network. Without filtering, the network has 128,520 paths, but after filtering, it has 19,772 paths.

2.2 Re-Write Rules

The good news is that Korean is a highly regular language. The number of possible endings and the combinations of these endings may be exponential but only a few rules are necessary to derive the surface forms. The aforementioned filters aided in limiting the number of possibilities and ensured that the strict templatic ordering of suffixes was not violated. Hence, the re-write rules focus more on alternations of underlying forms based on the verb stem and other morphemes.

Overall, the system has 45 rules handle these alternations and clean-up dummy variables and abstract vowels and consonants to which the re-write rules do not apply. The first two rules clean up the dummy variables used to ensure that the correct passives and causatives attached to the correct verbs and to distinguish the main verb 'ha' from the light verb 'ha.'

The next set of rules cover vowel harmony. ^E changes to 'a' if the verb stem contains an 'a' or 'o' and becomes 'eo' otherwise. Vowel harmony is local in Korean, and these rules ensure that ^E alternates only if the preceding syllable or two contain an 'a' or 'o.' The local harmonization means that a form like 'jap|ass|ta' 'grabbed' is valid but 'jap|hi|assta' 'was grabbed' is not because the correct context is no longer valid in the second example because of the intervening syllable. Many of the later rules rely on the surface form of ^E, so I placed these rules near the top to ensure that the later rules would have the correct surface form of the abstract vowel. define Rule3 [ %^E -> a || a (Cons|AbstrC) %| ng _ ] is one such rule.

Three of the underlying consonants are very interesting. First ^H becomes the vowel 'ae' before the past, past perfect, middle declarative, high declarative, low imperative but is deleted before 'eu.' Thus, a verb such as 'nola^H' can take on a wide variety of surface forms depending on which verb endings are present. define Rule7 [ %^H -> %| ae || _ %| [ End3 | End4 ]]. ^R is also quite interesting. It can become 'll,' 'l,' or get deleted depending on what morphemes follow. define Rule13 [ %| %^R -> 1%| 1 || (AbstrV | Vowel) %| [ 1 | End3 | End4 ]]. This rule maintains that ^R geminates when it is possibly followed by a vowel, a syllable boundary, and 'l' or the same endings mentioned above. This effects 'mo|^Reu' which becomes 'mol|la|yo' on the surface because of vowel harmony and gemination of ^R. On the other hand, ^R gets deleted when the following syllable begins with a morpheme that begins with an n such as 'neun' or 'na,' and it is optionally deleted when the following syllable begins with 's.' In all other cases, ^R becomes 'l.' Finally, ^P also has a variety of surface forms. ^P alternates between 'w,' 'u,' and 'p.' In a familiar trend, ^P becomes 'w' in the same context where ^R becomes 'll' and ^H
becomes 'ae.' This is more than likely for ease of articulation, as the 'w' and following vowel (either 'eo' or 'a') are then merged to create a diphthong. Hence, a verb like 'to^P' becomes 'to|wa|yo' thanks to vowel harmony, the ^P changing to a 'w' and the merge of 'w' and 'a' to create the diphthong 'wa.' ^P changes to 'u' before the honorific and morphemes beginning with 'eu' such as 'eul|kka' also for ease of articulation. This creates surface forms such as 'chu|ngu|se|ngyo' instead of the harder to say 'chup|ngeo|se|ngyo.' define Rule20 [ %^P %| -> %| w || _ ] [ End3 | End4 ]. These rules must be placed before some of the latter rules, which delete certain suffix vowels such as 'eu,' since if they came after, 'eu' and other suffix vowels would not be deleted because the context would not match because ^P and other underlying consonants would not have become derived vowels yet.

Despite its scientific nature, Korean still has a few irregular verbs. One such example is 'ha.' This is the only example of a verb in Korean (that I know of) where the derived 'a' from vowel harmony changes to 'ae,' and the stem vowel is then deleted. In all other cases, such as 'ka,' the derived 'a' from vowel harmony merges with the vowel from the stem, creating a surface form of 'ka.' However, this change to 'ae' is optional, and the derived 'a' can also take on the form 'yeo.' The rules for deriving the proper stem and suffix vowels for 'ha' depend only on the vowel harmony rules, and as such, I have placed them almost directly following the vowel harmony rules as rules 8 and 9. define Rule8 [ a (->) ae || h a %| ng _ ] and define Rule9 [ a -> ng yeo || h a %| ng _ ] respectively.

Also interestingly, the stem vowel 'o' can optionally change to 'w.' This occurs when 'o' ends a syllable, and the next syllable begins the vowel marker and 'a.' The derived 'w' and 'a' then merge to create the diphthong 'wa' much as they did when ^P turned into 'w' above. However, this rule is optional, so both 'po|ass|ta' (saw) and 'pwass|ta' (saw) are valid. Rule 26 handles this situation: define Rule26 [ o %| ng (->) w || _ a ].

Most of the alternations in Korean depend on whether the stem ends in a vowel or consonant. Typically, if a stem ends in a vowel, the underlying vowel is deleted. For example 'ka|ngeup|si|ta' becomes 'kap|si|ta' because 'ka' ends with a vowel. Rules to handle such phenomena come near the end of the main rules (the final rules are cleanup rules) so that all the surface forms have been derived and the vowel merge rules can apply in the proper contexts. One such rule is define Rule28 [ ng eu -> [..] || [ Vowel | AbstrV ] (\%) \_ (p | l) %| [ s | n | kk ] ( i | e | a | yeo ) ], which derives the above example with 'ka.' The vowel merge rules must come close to last because the underlying forms may end in an abstract consonant that becomes a vowel on the surface. Forms such as 'chu|^P+eo' -> 'chweo' would be missed if the vowel merge rules came too early.

After deriving the correct surface form for the underlying consonants and deleting unnecessary syllable initial vowels, it is possible to merge remaining vowels. For instance, if '(ngeu)|si' and 'nge|ngyo' occur next to each other, the 'i' merges with the 'e,' and the surface form is 'se|ngyo.' 'si' is also completely consumed by 'se|ngyo' producing the surface form of 'se|ngyo.' This means that one of the high imperative forms is indistinguishable from one of the high interrogative forms on the surface.

Glide formation is optional and turns an 'i' the ends a syllable into 'y' before 'ng eo' thereby creating 'ngyeo' or 'i|ngeo.' Rule 33 handles optional glide formation: define Rule33 [ i %| ng (->) [..] || j _ yo ]. It is key that this rule comes before the next rule for ^S. ^S is deleted before 'eo,' 'a,' and 'eu.' If the rule for ^S came before Rule 33, a verb like 'ji|^S' would become 'jyeo' thanks to the optional glide formation. This form is not valid. As such, if the rule for ^S occurs after the rule for glide formation, glide formation is blocked, and only valid forms of 'ji|^S' are produced.
The final interesting rule is a rule that only applies to a certain form. The 'oi' in 'toi' can optionally become a 'wae' before endings beginning with 'ngeo' and 'nga.' The typical form is actually the optional 'wae' derivation, but 'oi' occurs as well, so the only option is to make this rule optional. Finally, Rule 44 cleans up the derived form for transliteration purposes. Diphthongs are merged into single unit characters to agree with standard Korean orthography. That means, 'w eo' becomes 'weo,' and a similar transformation occurs for the other diphthongs in the rule. 'y eo' has two rules (%) ng y eo -> %| ng yeo, y eo -> y eo) because it can either occur after a consonant such as 'hyeo' or at the beginning of a syllable such as 'ngyeo.'

2.3 Transliteration

The final step was to map the ASCII characters to Korean syllables. Mapping the ASCII to syllables is necessary because the Korean writing system is not always right-to-left and because Korean words are syllable based. The mappings work as follows: First, I defined variables to take the place of ASCII characters used in the system. For example, define K k. I then used these variables in sequences and mapped these to a given Korean syllable. I followed Korean alphabetical order when deriving these syllables. For instance, K A %| -> "о." I created classes of such syllables (based on first character in Korean) and composed the rules within each of the 18 classes. After composing the mapping rules, I defined a variable to save each class. I then composed the variables for each of the 18 classes at the end creating a giant mapping network. I chose to break the syllables down into 18 different classes because XFST threw a "too many labels" error when I attempted to compose everything at once. The left side of each mapping ends with a syllable boundary to ensure that only the longest match is found and no ambiguous mappings occur. Some syllables can occur in the middle of a word or at the end. If they occur at the end, there is no syllable boundary. To handle this issue, I had two mappings for such syllables: one with a syllable boundary and one without. The rule without the syllable boundary occurred after all similar syllables to ensure that it would only match as a last resort.

2.4 Sample Derivation

The verb mo^Reu means 'not know.' This verb undergoes vowel harmony, geminification, and merging of 'eu' with 'a' on its way to the final surface form listed in step 6. The second 'eo' does not undergo vowel harmony because it is not in the correct local context for vowel harmony to occur.

1) mo^Reu+NoPass|+PastPerf+Interrog+Highest
2) mo^Reu+ng^Ess|eoss+seup|ni|kka
3) mo^Reu+ngass|eoss+seup|ni|kka (Rule 4)
4) mol|leu+ngass|eoss+seup|ni|kka (Rules 13, 14)
5) mol|lass|eoss+seup|ni|kka (Rule 25)
6) mol|lass|eoss+seup|ni|kka

The verb to^P means 'help.' The causative form produces a surface 'p' since neither of the ^P rules apply. However, vowel harmony and vowel merge both apply to 'ha+Light,' as represented by the optional rule 8.

1) to^P+CausIII|+CausIII+Past+Decl+High
2) to^P+ke+ha+Light+ng^Ess+ng^E|ngyo / ne|ngyo / ji|ngyo
3) to^P+ke+ha+ass+ngeo|ngyo / ne|ngyo / ji|ngyo (Rule 3)
4) to^P+ke+ha+aess+ngeo|ngyo / ne|ngyo / ji|ngyo (Rule 8)
4a) to^P+ke+ha+yeoss+ngeo|ngyo / ne|ngyo / ji|ngyo (Rule 9)
5) to^P+ke+ha+ess+ngeo|ngyo / ne|ngyo / ji|ngyo (Rules 10, 11)
5a) to^P+ke+ha+ngyeoss+ngeo|ngyo / ne|ngyo / ji|ngyo
6) top+ke+ha+ess+ngeo|ngyo / ne|ngyo / ji|ngyo (Rule 38)
6a) top+ke+ha+ngyeoss+ngeo|ngyo / ne|ngyo / ji|ngyo (Rule 38)
7) top|ke+ha+ess|ngeo|ngyo / ne|ngyo / ji|ngyo
7a) top|ke+ha+ngyeoss|ngeo|ngyo / ne|ngyo / ji|ngyo

3. Conclusion

The main contributions of the system are the mapping from the ASCII to Korean and the filters. The filters ensured that no invalid forms could be generated and that the proper order of suffixes was not violated. The filters also ensured that the correct passive and causative occurred with the correct verb and that certain verbs did not occur with certain suffixes. The 2000 plus mappings created a system that can be read by Koreans and Korean speakers and made the debugging that much easier. The mapping also checked that the Romanization was indeed systematic and there was a one-to-one mapping between ASCII characters and Korean characters.

Korean is very scientific and regular. The main complication is the sheer number of possible verb endings. However, with the filters, the number of possible combinations was greatly reduced. This meant that the re-write rules could focus on the morpho-phonological alternations. Rule ordering played a role, and I spent countless hours ensuring that the rules derived only valid surface forms and did not apply in inappropriate contexts. This led to numerous versions of the system before the ultimate final version.

The future is bright and complicated for Korean morphology. If I had more time, I would greatly expand the lexicon and make this a general purpose morphological analyzer for Korean verbs. I did not have time or space for many, many endings, but my project is representative of Korean verb final endings. It would still be possible to add the continuation suffix as well as some non-final endings. I leave this experiment for a future project. The final output when all possible combinations of suffixes were considered produced approximately 6400 forms. Naturally, this number would greatly expand if more endings were added. Korean morphology is not for the faint of heart, but this project proved to be an exciting challenge and a joy to implement.
Works Cited


http://www.korea.net/korea/kor_loca.asp?code=A020301
