Journal of Liberal Arts and Sciences,
Korea Maritme University, Vol.1, 1993.

Korean Inflectional Morphology Parser (KIM Parser)*

Jeong-Ryeol Kim*

〈CONTENTS〉

Ⅰ. Introduction

Machine Translation (MT) is energetically being revived in various parts of the world such as EuroTra in European Community, ECS machine translation toolkit and other machine translation workbenches in U.S. Most language pairs under consideration in both Europe and US are typologically, genetically and even culturally similar to each other. However, since Japan started to participate actively in this ambitious phase, for example, Nihon Telephone and Telegraph project, the kinds of languages in MT have been diversified to include SOV type languages such as Japanese and Korean. One of the major characteristics of Japanese and Korean is the complexity of verbal morphology which naturally results in the complexity of designing and implementing verbal morphology parser.

The paper discusses the design and implementation issues laid behind the Korean verbal morphology. Considering the complexity of the verbal morphology and the scale of

*한국해양대학교 교양과정부 교수(언어학 전공)

implementation. I delimit my discussion to the matrix verbal morphology and its implementation. The parsing technique to be hired in the discussion will be Finite State Transition Network (FSTN, Aho and Ullman 1972, 1977; Langendeon and Langsam 1984, Pullman 1986).

FSTN has been criticized in not being powerful enough to cover certain linguistic aspect such as multiple center-embedding phenomenon, but it is simple and easy to implement, and efficient at doing what it is designed to do. (Gazdar, G. and C. Mellish 1989)

II. Theoretical Assumptions

Before proceeding any further, the distinction between token and type needs to be drawn. They are roughly equivalent to the distinction between allomorphs and morphs.[1)]

Type is a collective name which refers to a group of tokens. Token is a single occurrence of an element that satisfies the requirements for the membership of the category specified by the type. Taking an example from English plurals, "plural" is a type and each occurrence of plural, [-s], [-z] and [-Ez], is a token.

This parser is strictly based on the "context-freeness" of syntagmatic relationship of tokens. A syntagmatic relationship is context-free if the relationship does not require one of its sister tokens to be of certain type. In other words, a token does not refer to the further classification of its sister token, if it is context-free to another token.

The parser is modulaic and consists of two main parts: Exhaustive listing where all the syntagmatic relations of tokens are produced exhaustively, and Filter which suppresses an inadmissable combinations of tokens. Also, this parser is based on the compositionality of a grammatical category with its syntagmatic units. The compositionality in this paper is strictly restricted to "form", not "sense" which has been the typical use of compositionality. Two tokens are compositional if and only if the tokens make up a legitimate type. For example, Verb Stem forms a type with any other suffixes within verbal morphology with the obligatory presence of Sentential Marker, but Honorification cannot form a type with another token in the absence of Verb Stem, even though Sentential Marker exists. In other words, Verb Stem plus a suffix with Sentential Marker is compositional, but Honorification plus another suffix by themselves are not.

III. Description of Parser

3.1 An Exhaustive Listing of Syntagmatic Relations of Tokens

There are six types of verbal morphemes in Korean verbal inflectional morphology: from left to right, Verb Stem (VS), Honorification (HON), Negation (NEG), Tense (TNS), Formality (FML) and Sentential Marker (SM). A brief characterization of these types is made as follows:

VS has 3 tokens which will determine a group of following tokens, one token is VS ending with "ci" to form a negation with the following NEG "anh", another token is VS ending with a vowel (VSV) excluding "ci" ending, and the last one is VS ending with a consonant (VSC). VSV makes up an admissable inflectional morphology only with certain set of morphophonemically compatible tokens such as "si", "n", and "p" as opposed to the fact that VSC is morphophonemically compatible with "usi", "nun" and "sup". Another characteristic of VS is its obligatory presence of a real phonetic form. In other words, other types are in general optional in nature, but VS and SM are obligatory in nature.

HON marks the relative relationship between subject and speaker. HON includes 5 tokens: "si", "usi", "sy", "usy" and 0 (null). 0 (null) states that HON is optional, but that does not mean that the absence of HON in a real phonetic form constitutes the null value. The phonetic null specifies that the subject is NOT honorified.

NEG includes two tokens: "anh" and 0 (null). 0 (null) describes that the absence of a real phonetic form means the opposite value of NEG, that is, affirmative.

TNS covers 3 morphophonemically determined tokens for present tense, 2 tokens for past tense and 1 token for future.[2)]

FML indicates the relationship between speaker and hearer. It includes 3 tokens: "supni", "pni" and 0 (null). The first two tokens are morphophonologically conditioned depending on the preceding morpheme (either consonant-ending or vowel-ending), and the 0 (null) token is in complementary distribution with the present tokens of TNS.

SM covers 5 tokens of 3 subtypes. The 3 subtypes are interrogative, declarative and

imperative. Each subtype is realized as the following phonetic tokens: "kka" and "ekka" (interrogative), "ta" (declarative), and "la" and "ela" (imperative).

The exhaustive combination of these tokens is a cartesian product of the tokens keeping the given order of type crucially. Consider the figure 1 and the following description of its cartesian product:

fg1

TYPE	VS	HON	NEG	TNS	FML	SM
# of TOKENS	3	5	2	6	3	5
TOKENS	VSN	0	0	0	0	kka
	VSV	usi	anh	nun	supni	ekka
	VSC	usy	n	pni	ta	
		si	keyss	la		
		sy	ess	ela		
			ss			

The possible combinatorial rules are 3 * 5 * 2 * 6 * 3 * 5 (=2700) rules, but of course, some of these combinations are impossible due to the nature of complementary distribution of the syntagmatic tokens under each type.

3.2 Suppressing Rules

As noted in the discussion of the cartesian product of the tokens, the finite state machine depending only on the syntagmatic relationship of tokens is too powerful to block inadmissible forms. The suppressing rules filter out inadmissible forms. It serves to suppress any combination which is contradictory to pragmatic grounds, possibly specific to Korean, for example, the combination of FML token and an IMP token is not found in the language. It also serves to suppress inadmissible combinations resulting from morphophonological

constraints such as *VSC + si (HON) + ta (SM).

The rules stating combinations of tokens are checked to see whether or not the output of the rule is grammatical in both pragmatic and morphophonological grounds. If the output of certain rule is ungrammatical, then the rule is to be suppressed and checked out to spare processing time. Using the same example of FML+IMP combination, any finite state linking these two tokens is suppressed and eliminated.

This way of approaching the problem is somewhat related to the issue of whether one can construct a less powerful grammar and find some grammatical string which does not fit it, then make the grammar only powerful enough to explain the grammatical string, or make a grammar as powerful as one can, then reduce the power down to eliminate the ungrammatical string. As one can notice from the term "filter" or "suppressing rules", this parser takes the latter position. How I justify this position against the general belief that it is harder to disapprove any part of grammar because of difficulty of finding out an ungrammatical string rather than a grammatical string. This position does not necessarily hold if the set of all the possible combinations of tokens resulting from grammar is finite. In other words, if anyone's grammar is computationally precise and definite, then there is always a set of data which can be explained or described by the grammar, and another set of data which cannot be explained or described by the grammar. This will show us that as long as this is clear it should not be the case that finding out a grammatical string is easier than finding out an ungrammatical string or vice versa. Consequently, the issue of disaprovability does not depend on the direction of building up grammar, but it crucially depends on the precise definiteness of the grammar.

The problem of the definiteness of a grammar allows us to look into some more fundamental problems about the nature of the complexity which intrinsically exists within the structure of a grammar. It is generally known that a finite state language is computationally concise and definite with relation to its learnability and parsability as compared to a context-free language, and the context-free language is so as compared to a context-sensitive language, and again the context-sensitive language is so as compared to a recursively enumerable language.

However, there are some opposing views to this hierarchy, for example, Barton, Berwick and Ristad (1987:24) claim that some context-free languages are generated only by very large context-free grammars, and the size of the grammar will affect the parsing time for all well-

known general parsing algorithms. Additionally, a finite-state language does not demonstrate anything about the nature of underlying constraints on natural languages and a natural language is just accidental if a finite mechanism can resolve the complexity. Despite the counter-arguments by Barton et al. Aho and Ullman (1972) fairly well explored this problem in a compiler design, claiming it is grossly more difficult to do for a context-sensitive language than for a context-free language. Also, Pullum and Gazdar (1982) show effectively that the issue on context-free languages has been unfairly dismissed in the modern history of linguistics by arguing against those who have proposed what context-free machinery is insufficient to deal with natural languages.[3)]

Gazdar (1982:132-33) states that the context-free language is favored over the context-sensitive language in terms of the complexity of machine processing of natural languages as well as learnability and parsability.

Some concrete comparison between context-free and context-sensitive ways of constructing Korean Inflectional Morphology Parser is shown in Kim (1987). I just want to mention the schematic provision in constructing the parser when we use the context-free grammar without going through all the details in comparing two different ways of constructing a grammar.

When we see a grammar as a well-sorted combination of tokens on a context-free frame, we can easily imagine that a phonological component, morphological component and syntactic component can be integrated into one progressive chain of token-type relations, even though the parser I constructed here readily leaves conceivable traces.

One way of doing this is to require all the instances of a morpheme on the surface level to be tokens of morpheme types and all the instances of morphemes of a word to constitute tokens of word types and all the instances of a phrase to constitute tokens of phrase types, and so on.

3.3 Minor Adjustment

The input for the compiler requires a strictly formulated expression; the KIM parser hires this syntax because any other form of compiler is not ready yet. The script for using this compiler is generously provided by Dr. Greg Lee.

The input for compiler has two major components, one of which is a set of grammar rules and the other of which is a set of lexical rules.

Also, each grammar rule is divided into 3 parts, one is the mother node on the lefthand side of the arrow, another is the sister nodes on the righthand side of the arrow and yet another is the interpretation rule on the further left after an left-angled bracket "〈". The lexical rules consist only of types and their tokens separated by arrows.

3.4 Compiler

The compiler this parser uses for this job is a combination of yacc and lex. Yacc is a compiler taking a regular expression and returning an execution program which we test a parser (or analyzer), and it can be used with lex which is good at returning what it receives.

IV. Theoretical Consequences

In this parser the minimal unit which the parser will accept is a string of phonemes in the traditional sense, which constitute a token in a type. When we talk about underlying phonemes based on one-to-one correspondences between morpheme and phonological representations, we take the position that the in-between phonemic level traditionally widely identified should be abandoned. However, we can see various examples showing in fact there is a very crucial level between underlying representation and phonetic representation.

For example, in a secret language like "ab" language (David Stampe, personal communication) or "babibu" language (Haraguchi 1977)[4] when people pronounce "kisses" and its secret language for "kabIssabEz". If the phonological conditioning is the single factor to determine the output of the underlying phoneme /kIsz/ and its secret version we can expect [kabIsz] to be the output, but in fact [kabIsabEz] is the output.

One may object to this conclusion by saying that the rule ordering is in fact crucial here in such a way that the E-insertion applies first and the ab-insertion later. If this is the case then we can predict that "latter" and "ladder" will have the same form "labaeDabEr" since the ab-insertion will apply after all the regular phonological rules have applied, but in fact "latter" is "labaetabEr" and "ladder" is "labaed[~D]abEr".

One more example from babibu language is that "anai" in "kak-anai" (write-not) is a variant of "nai" (not) in "mi-nai" (see-not) and the conditioning factor is known to be whether the verb stem is consonant-ending or vowel-ending will determine the nai or anai-insertion will

apply to the case which two inadmissable consonant cluster comes adjacently in an example / kak-nai/ -> [kaka-nai] (write-not). Analogously "noN" (drink) will have the babibu language form no-bo-N-bu and this will convert "noN-" into a verb stem ending with a vowel and then the choice of negative suffix will be "-nai" instead of -anai. The whole string of this in babibu language should be no-bo-bu-na-ba-i-bi but in fact the correct form is no-bo-ma-ba-na-ba-i-i which it takes -anai form. This again proves that there is some level between underlying representation and phonetic representation.

More examples can be drawn from a reversed pronunciation which clearly distinguishes the phonemic neutralization from the allophonic convergence in terms of recoverability. In Korean [kwut][i] 'firm + ly' becomes [cuci] and the reversed pronunciation of this is [cigu], while [cikwu] 'earth' becomes [cigu] and its reversed pronunciation is [kuji].

This shows that the phonemic change is not recovered in a reversed pronunciation, but the phonetic change is readily recovered. Also, we can quote some examples in rhyming which show in fact the in-between phonemic level is the rhyming unit. Why do "matter" and " ladder" rhyme?

Yet another case showing the level between underlying representation and phonetic level is the alphabetic orthographic system of many languages. For example, there is a famous triplets "p", "ph" and "pp" in Korean, when a Korean transcribes "Paris" pronounced by a English speaker and by a French speaker, the former will be /phali/ and the latter will be /ppali/. The basic unit for the orthographic system is the phonemic level, not underlying representation nor phonetic representation. Also, Mohanan (1982) shows the importance of this level in a non-sense word test between English and Malayalam. In Malayalam there is a distinction between the regular /n/ and the dental /n/ distinction while in English there is not. Consequently the Malayalam native speaker distinctively perceive and produce the two different /n/s while English speaker cannot, even a trained linguist has a difficulty in discerning the dental /n/ from the regular /n/.[5)]

The rather lengthy explanation above hopefully justifies the consideration of a phonemic unit as a token.

The assumption which I mentioned earlier, "context-freeness", implies a rather important consequence of its own. The context-free approach to the tokens described above shows the

same weak generative capacity as the context-sensitive approach to the same problem. In spite of the same weak generative capacity, the context-free grammar uses a much simpler apparatus than the one which the context-sensitive grammar might use for the same job. Also, with the context-free grammar, phonology, morphology and syntax can be integrated into one assembled system sharing a large portion of the devices and criteria instead of having to be considered distinct systems.

Compositionality can provide a systematic device to suppress a grammar, directing it from the finite-state language toward the context-free language. The example above in which the formality particle inadmissibly combines with the imperative particle indicates a rather substantial incompatibility of formalness with imperativeness. The other sort of example due to complementary distribution will be the case which one can claim that the systematic substance comes from some physiological constraints on human speech organs.

V. Conclusion

The main goal of this project is in constructing a Korean Inflectional Morphology (KIM) Parser. The characteristics of this parser are context-free, compositional, modulaic and token-type progressive.

As a by-product of working on this job I realized that the preference of certain working paradigm can be tested based on the complexity of each theory in terms of computation. The wild conclusion, since this is still on-going study which has a potential revision, is that the context-sensitive grammar is more complex than the token-type progression based on context-free grammar.[6] In this paper the integrity of the apparatus used in parts of the grammar shows that we can count more on the token-type progressive chain than on the context-sensitive grammar. Also, the underlying representation cannot be the basic token unit because of its dubious characteristic in the perception and production of speech, that is, there is nothing we can look into but highly formulaic archiphoneme.

The author is aware that there is a movement in generative phonology to constrain the underlying representation, which seems to be a natural direction of change as far as it is the goal of linguistics to construct a realistic phonology. Meanwhile, the traditional phonemic level is a psychologically perceivable unit in various tests, and thus that is how I adopted the level as the starting point.

Finally, the author wants to briefly mention something about what KIM Parser does in the context of machine translation. The recognition of verbal morphemes is very crucial especially if the translation goes from Korean to English, since it will be the case where some linguistic device in a source language will not have the corresponding value in a target language, specifically Honorification and Formality in Korean. If translation proceeds the other way around, that is, from English to Korean, it will require a much more pragmatic knowledge about Honorification and Formality in the level of accuracy.

The KIM Parser has also some potential to be applicable to a language analyzer which is required in a language game such as "doctor" and speech recognition.

Notes

* I thank Dr. George Grace for his insightful concept of token-type distinction which becomes the corner stone of the paper and Dr. Gregory Lee for his generous offer of his compiler without which the project could not have been finished. However, any remaining errors are of my own.

1. This distinction between type and token was introduced by Dr. George race in a class discussion.

2. The nature of TNS type with reference to tense, modality and aspect is still under discussion. The major points brought up are

(1) the future tense marker "kess" is not semantically appropriate for the future tense marker since it can refer to a probable on-going action. In this regard, it is closer to the probability modality "may" in English.

(2) "ess" marks a past-completed event, for example, "nay-ka ka-ss-ul ttay, ku-nun imi ka-ss-ta." (When I went, he had already gone.) "ess" indicates the event happened before the past event referred to in the dependent clause occurred.

Defining the exact nature of TNS is not the key issue of this paper, thus, TNS is used as a cover term (type) referring to these tokens. They are "n", "nun" and 0 (null) present, "ss" and "ess" for past and "keyss" for future.

3. First, Pullum & Gazdar (1982:477) argue against Chomsky (1956) who attempts to show

that natural language is not context-free. based on the non-identity in a comparative construction, claiming Chomsky's grammatical judgement about the sentence "That one is wider than this is s wide." is wrong. Second, against Elster (1978) who applies the pumping lemma directly to a set of English sentences about the decimal expansion of pi. Elster's mistake is to confuse grammaticality with felicity (ibid 481). Third, against Bar-Hillel (1953) who shows English is xx language with "respectively" construction. Pullum and Gazdar have a different grammaticality judgement about the example:

"John, Mary, David and Suzan are a widower, a widow, a widower and a widow respectively." in which they claim is ungrammatical. Fourth, the claim made in Huybregts (1985) that Dutch is not context-free collapses in the demonstration of the context-free rules treating the same construction (ibid 487). Fifth, although Postal (1969) argues that the interaction of the processes of nominalization and incorporation in Mohawk yields a property that places Mohawk outside the set of context-free languages they show the mathematical flaws in Postal's argument (ibid 491-7).

4. "ab" language is a widely known secret language among American kids. People insert " ab" between onset and rhyme, for example "department" will be "dabepabartmabent." " babibu" language is a secret language among Japanese junior high school girl students. It works in such a way that they insert "b" after a mora and copy the preceding vowel, so for example, "amega hurimasu" (it's raining.) will be "a-ba-e-be-ga-ba hu-bu-ri-bi-ma-ba-su-bu." Interestingly, if a syllable ends with a consonant like "heN-na" it will be he-be-n-bu-na-ba. Here we can see they insert the weak vowel when there is no vowel to copy in the preceding mora.

5. As shown in the main text, three different levels of nasal sounds are schematically distinguished as follows:

	English	Malayalam
underlying	m, n	m, n, nn
lexical	m, n, N	m, n, nn, n′, n′′, N′, N
phonetic	m, n, n′, n′′, N′, N	m, n, nn, n′, n′′, N′, N

[nn] is the /n/ in tenth, [n′] is the /n/ in Monroe, [n′′] is the [N] in lynch.

6. A more detailed report in which the preference to token-type progression is mathematically shown in Kim (1987).

References

Aho, A.V. and J.D. Ullman. 1972. The theory of parsing, translation and compiling. Prentice-Hall, Englewood Cliffs, NJ.

--------- 1977. Principles of compiler design. Addison-Wesley, Reading. MA.

Bar-Hillel, Y. 1953. A quasi-arithmetical notation for syntactic description. Language 29, 47-58.

Barton, Berwick and Ristad. 1987. Computational complexity and natural language. MIT Press, Cambridge, MA.

Chomsky, N. 1956. Three models for the description of language. IRE Trans. on information theory IT-2, No.3, 113-124.

Gazdar, G. and C. Mellish 1989 An introduction to computational linguistics, Addison-Wesley Publishing company, New York.

Haraguchi, S. 1977. The tone pattern of Japanese: an autosegmental theory of tonology. Kitakusha, Tokyo.

Huybregts, R. 1985. The weak inadequacy of context-free phrase structure grammar. In Van periferie naar kern (Gier de Haan, Mieke Trommelen and Wim Zonneveld, eds.) pp.81-89. Foris, Dordrecht.

Kim, J.R. 1987. Parsing the Korean verb. In Working papers in linguistics, University of Hawaii, Honolulu, HI.

Langendeon, D.T. and Y. Langsam. 1984. The representation of constituent structures for finite-state parsing. COLING-84, 24-7.

Mohanan, K.P. 1982. Lexical phonology. PhD. dissertation, MIT.

Postal, P. 1968. Anaphoric islands. CLS 5, 205-239.

Pullman, S. 1986. Grammars, parsers, and memory limitations. Language and cognitive processes, 1, 197-225.

Pullum, G. & G. Gazdar. 1982. Natural languages and context-free languages. Linguistics & philosophy, 4, 471-504.