Morphological Analysis and Synthesis by Automated Discovery and Acquisition of Linguistic Rules

Byoung-Tak ZHANG[1] and *Yung-Taek KIM*
Department of Computer Engineering
Seoul National University
Seoul 151742, South Korea

Abstract

This paper describes a rule-based machine learning approach to morphological processing in the system called XMAS. XMAS discovers and acquires linguistic rules from examples of morphological combinations and accomplishes the morphological analysis and synthesis by applying the rules. This approach is independent of languages, saves time and effort for development and maintenance, and takes small lexicon space. A Korean version of XMAS is effectively working in the English-Korean machine translation system KSHALT.

1 Introduction

From the knowledge engineering perspective in artificial intelligence, a natural language processing (NLP) system can be seen as a knowledge-based system in which major concerns are the discovery, acquisition and maintenance of knowledge or rules [2].

The main purpose of this paper is to present a method for facilitating the development and maintenance of NLP systems by automating the discovery and acquisition of linguistic rules in the domain of morphology. This method was implemented in a morphological processor called XMAS as a part of the English-Korean machine translation project KSHALT [4].

In Section 2, we give an overview of the XMAS system. Section 3 illustrates the representational formalism. In Section 4, the processes of discovery and acquisition of morphological rules are described. In Section 5, the procedures for morphological analysis and synthesis are briefly described and the application results are reported. Section 6 concludes this paper by comparing XMAS with other related works and by remarking on further work.

2 An Overview of XMAS

XMAS (Xpert in Morphological Analysis and Synthesis) is a learning system [12,13,14] which consists of a learning element (Meta-XMAS), a knowledge base (KB), and two inference engines of a morphological analyzer (MOA) and a morphological synthesizer (MOS). Figure 1 shows an overview of XMAS and its operational environment (parser and generator of natural language systems).

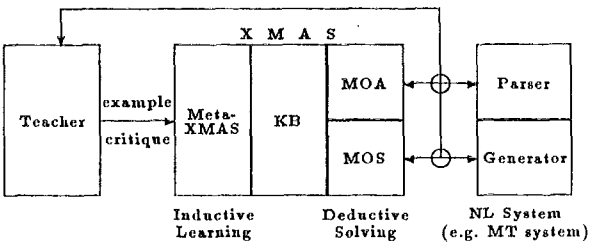


Figure 1: The overview of XMAS

[1]Present address: Artificial Intelligence Division, Institute for Computer Science, and Computational Linguistics Division, Institute for Communication Research and Phonetics (IKP), University of Bonn, D-5300, Bonn 1, West Germany, E-mail: ZHANG@DBNINF5.BITNET.

The knowledge base contains a lexicon and a rule base. The lexicon has entries only for word stems. The rule base contains the morphological rules which are learned from training examples. TEACHER, a human trainer, traces the execution of XMAS and provides training examples and critiques for Meta-XMAS. Meta-XMAS, a machine learner, acquires linguistic knowledge by discovering, formulating, generalizing and specializing grammatical rules. MOA solves the morphological analysis problems by applying the rules. MOS accomplishes the morphological synthesis by applying the rules.

3 Description Languages

3.1 Example Description Language

The rule discovery procedure is initiated by training examples. Two kinds of training examples are used: generalization examples and specialization examples. The generalization examples are used to maintain the completeness of the rule base and the specializaition examples are used to maintain the consistency of the rule base [3]. A training example consists of a *class name*, two strings of *before* and *after*, and a *critique*—an optional tag, indicating if the example is a specialization example. An instance of a generalization example is (PLURAL "baby" "babies").

3.2 Rule Description Language

The formalism for the representation of morphological rules is the string productions which are similar to if-then production rules [14]. A string production is a self-contained operator which consists of a left-hand side (LHS) and a right-hand side (RHS). It is symmetric and therefore applicable bidirectionally.

```
<string-production>  ::=  ( <LHS> ⟶ <RHS> )
<LHS>                ::=  ( ( <feature>* )  <pattern> )
<RHS>                ::=  ( <pattern> ( <feature>* ) )
<feature>            ::=  <category> | <conjugation> |
                          <prononciation> | <accent> | ···
<pattern>            ::=  ( <lwindow>  <main>  <rwindow> )
<category>           ::=  NOUN | VERB | ADJ | ADV | ···
```

The string production can accomodate not only graphemic patterns but also syntactic and phonological features [1]. A string production can have name(s) which is useful for direct access and maintenance of the rules.

For example, the following rule is a string production

$$(((\text{NOUN SG})\ ((\text{b})\ (\text{y})\ (\%))) \longrightarrow (((=)\ (\text{i e s})\ (\%))\ (\text{NOUN PL})))$$

where =y/ies% can be assigned to the name of the rule whose class name is PLURAL.

4 Automated Learning of Morphological Rules

4.1 Learning Procedure

The procedure LEARNING (Figure 2) describes the top-level algorithm for rule learning in Meta-XMAS.

```
procedure LEARNING(classname, before, after, critique)
   features ← LEXICON(before)
   (mop, rulename) ← DISCOVER(before, after)
   H ← FORMULATE(classname, rulename, features, mop)
   matchset ← MATCH(rulename)
   if (matchset = {})
      then CREATE(H)        // a new rule H is generated //
      else
         R ← RESOLVE(matchset)     // select one rule //
         if positive(critique)
            then GENERALIZE(R,H)   // R is generalized //
            else SPECIALIZE(R,H)   // R is specialized //
end LEARNING
```

Figure 2: The learning procedure of Meta-XMAS

Given a training example, Meta-XMAS searches the lexicon for the features of the word stem *before*. By comparing the *before* with *after*, a micro-operator (mop) and its name is constructed (DISCOVER). The mop is a description of the transformation procedure from *before* to *after*. From these, a hypothetical rule H is generated (FORMULATE). Then it checks if other rules with the same name already exist (MATCH). If yes, then the most special rule R is selected (RESOLVE) and H and R are generalized or specialized according to the critique (GENERALIZE or SPECIALIZE). If no, then the rule H is appended to the present rule base with the new rule name (CREATE).

4.2 Discovery of Rules

The rule discovery procedure works as follows. First, the input strings *before* and *after* of the training example are compared and splitted by using the four pointers—*lb*, *rb*, *la* and *ra*. The pointers *lb* and *la* move one grapheme at a time from the left end to the right, while the pointers *rb* and *ra* move from the right end to the left, until *lb* and *la* (*rb* and *ra*, respectively) point different graphemes.

This results in dividing each string into the three regions (lwindow, main and rwindow) which represent the left context, main string, and the right context, respectively. The corresponding contexts of both strings are generalized to '=' symbol which means *remains unchanged.* The result is a micro-operator (mop). And then, a rule name is given to the mop by creating a symbol representing the transformation form. For the training example

(PLURAL "baby" "babies")

the mop is

((b) (y) (%)) ⟶ ((=) (i e s) (%))

whose name is =y/ies%.

Finally, a hypothetical rule for the given training example is generated by integrating the class name, the rule name, the features, and the mop. In our example, the following rule is generated (in a simplified form):

=y/ies%: (PLURAL (NOUN) ((b) (y) (%)) ⟶
((=) (i e s) (%)))

4.3 Acquisition of Rules

The acquisition process of rules is the iteration of the creation of new rules and the modification of existing rules. Modification is carried out by generalization on the one hand and specialization on the other hand. Meta-XMAS uses the four kinds of induction rules in generalization [10]:

1. *Variablization of constant*: replace a grapheme by =.
2. *Dropping AND conditions*: decrement the contextual window size.
3. *Adding OR conditions*: union the two grapheme sets.
4. *Climbing generalization trees*: climb the path to the root of the tree.

A concept is represented as a set of graphemes or features which can have an explicit name. The generalization tree [11] is a tree which describes the generalization relations between the concepts in the domain. Figure 3 shows a generalization tree for Korean graphemes.

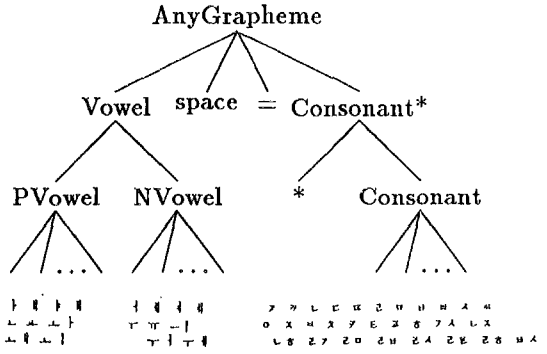


Figure 3: A generalization tree for Korean graphemes

Generalization using the tree is a process of following the path to the root on the basis of training examples. This is useful in concept learning in a well-established domain. In general, however, the generalization tree is not given *a priori.* In this case the Meta-XMAS builds the trees for itself (see Appendix A). The two generalization strategies have their strengths and limitations. The first strategy needs more *a priori* linguistic knowledge, but it is better in learning efficiency. The second strategy guarantees to find detailed rules, but is less efficient in learning speed.

If Meta-XMAS applies only the generalization schemes, the rules learned can be overgeneralized. So at any point (e.g. in case of false output) the rules should be specialized to avoid the overgeneralization or to handle the exceptions. Meta-XMAS accomplishes this automatically with the aid of specialization examples. The specialization process is a kind of rule creation procedure in which overgeneralization is checked and the overly generalized rule is recovered—by going down the generalization tree to the leaves or eliminating a grapheme from a concept, if necessary.

5 Morphological Analysis and Synthesis

XMAS has two inference engines (or rule-interpreters) of MOA and MOS. MOA, the morphological analyzer, solves the morphological analysis problms by applying the rules in the RHS-driven forward chaining manner [14]. MOS, the morphological synthesizer, solves the morphological synthesis problems by applying the rules in the LHS-driven forward chaining

manner (see Appendix B). More than one morphemes can be synthesized and analyzed by an instruction. If more than one rule is applicable in rule selection, then MOA/MOS selects the most special rule. So the rules in the rule base need not be ordered, making the maintenance of the rule base very easy.

XMAS is implemented on a SUN workstation in a Common-LISP program. A variety of experiments were carried out with XMAS on derivational and inflectional phenomena in almost all of the Korean language and part of the English and the German languages [17]. The results were successful in the sense that Meta-XMAS learns the grammatical rules and MOA/MOS solves correctly the morphological analysis and synthesis problems by applying the rules. Part of the rules learned by XMAS are shown in Appendix C. A more efficient Korean version of XMAS, called MASK [19], was applied to the generation of Korean language in the English-Korean machine translation system KSHALT [4] and MASK is effectively working now.

6 Related Work and Concluding Remarks

In summary, XMAS, including Meta-XMAS as a linguistic knowledge acquisition tool, is characterized by the following capabilities and properties:

1. using graphemic, phonological, and syntactic information
2. analysis as well as synthesis
3. language independent
4. capable of treating infix inflections as well as prefix and suffix
5. simple and small lexicon
6. rules need not be orderd
7. automatic discovery and acquisition of linguistic knowledge by inductive machine learning
8. easily maintainable
9. working in a machine translation system.

The properties (2), (3), and (4) are the same as in Kaplan et al. [5], but XMAS has in addition the properties (5)-(9). XMAS shares the properties (3) and (7) with Wothke [15], but is improved by the capabilities (1), (2), (4) and (6). The advantage (8) comes especially from (6) and (7). The property (9) is an indirect demonstration of XMAS approach as a practical approach.

As for all rule-based systems, XMAS is less efficient in run time in comparison with procedural systems [6]. There is a way to improve the run-time efficiency. One can construct a rule compiler as the transducer in KIMMO systems [5,7].

But XMAS is better in effectiveness in development and maintenance. In addition to the run-time efficiency, these factors should also be considered because, as was mentioned at the beginning, a natural language processing system is a complex knowledge-based system which requires a long period of design, implementation, test, debugging, and extention time.

In addition to being a practical knowledge acquisition tool, Meta-XMAS in isolation can also be used as a linguists' tool for scientific discovery [8] which aids linguists in the discovery and test of grammatical rules in morphology.

Acknowledgements

This work was supported in part by IBM Korea, Korea Science Foundation, and Department of Science and Technology of Korean Government. We would like to thank Hyuck-Cheol Kwon, Young-Hoon Seo, Duck-Ho Yun, Kwang-Seob Shim, Seung-Sik Kang, Se-Jung Kim, and the other members of the project KSHALT for their discussions and useful comments. The first author thanks also Prof. Dr. Winfried Lenders and Hooshang Mehrjerdian for careful reading and suggestions.

References

[1] Bear, J., A morphological recognizer with syntactic and phonological rules, *Proc. of COLING-86*, pp. 272-275, Bonn, 1986.

[2] Hayes-Roth, F., Waterman, D.A., and Lenat, D.B., *Building Expert Systems*, Addison-Wesley, 1983.

[3] Jansen-Winkeln, R.M., Induktives Lernen von Grammatikregeln aus ausgewaehlten Beispielen, In: Savory, S. (ed.), *Kuenstliche Intelligenz und Expertensysteme*, pp. 211-223, Oldenbourg, 1985.

[4] Kang, S.S., Shim, K.S., Zhang, B.T., Kwon, H.C., Woo, C.S. and Kim, Y.T., An English-Korean system for human-assisted language translation, *Proc. of TENCON-87*, IEEE, pp. 550-555, 1987.

[5] Kaplan, R. and Karttunen, L., *Computational Morphology*, CSLI Report No. LI 283, June 1987.

[6] Kim, S.Y., Choi, K.S., and Kim, K.C., A Korean morphological analyser using tabular parsing and connection information, *Proc. of 87 Spring Conf. on Arti. Intell.*, Korea Info. Sci. Soc., pp. 133-147, 1987, (in Korean).

[7] Koskenniemi, K., Two-level model for morphological analysis, *Proc. of IJCAI-85*, pp. 683-685, 1985.

[8] Langley, P., Zytkow, J.M., Bradshaw, G.L., and Simon, H.A., Three facets of scientific discovery, *Proc. of IJCAI-83*, pp. 465-468, 1983.

[9] Lenders, W. and Willée, G., *Linguistische Datenverarbeitung: Ein Lehrbuch*, Westdeutscher Verlag, 1986.

[10] Michalski, R.S., A theory and methodology of inductive learning, In: *Machine Learning*, Michalski et al. (eds.), Tioga press, pp. 83-134, 1983.

[11] Mitchell, T.M., Utgoff, P.E. and Banerji, R., Learning by experimentation: acquiring and refining problem-solving heuristics, In: *Machine Learning*, Michalski et al. (eds.), Tioga press, pp. 163-190, 1983.

[12] Smith, R.G., Mitchell, T.M., Chestek, R.A., and Buchanan, B.G., A model for learning systems, *Proc. of IJCAI-77*, pp. 338-343, 1977.

[13] Veenker, G. (ed.), *Automatisches Lernen*, Arti. Intell. Seminar, Inst. for Comp. Sci., Univ. of Bonn, 1990.

[14] Waterman, D.A. and Hayes-Roth, F., An overview of pattern-directed inference systems, In: *Pattern-Directed Inference Systems*, Waterman, D.A. and Hayes-Roth, F. (eds.), Academic press, pp. 3-22, 1978.

[15] Wothke, K., Machine learning of morphological rules by generalization and analogy, *Proc. of COLING-86*, pp. 289-293, Bonn, 1986.

[16] Wothke, K., *Maschinelle Erlernung und Simulation morphologischer Ableitungsregeln*, Dissertation, IKP, Univ. of Bonn, 1985.

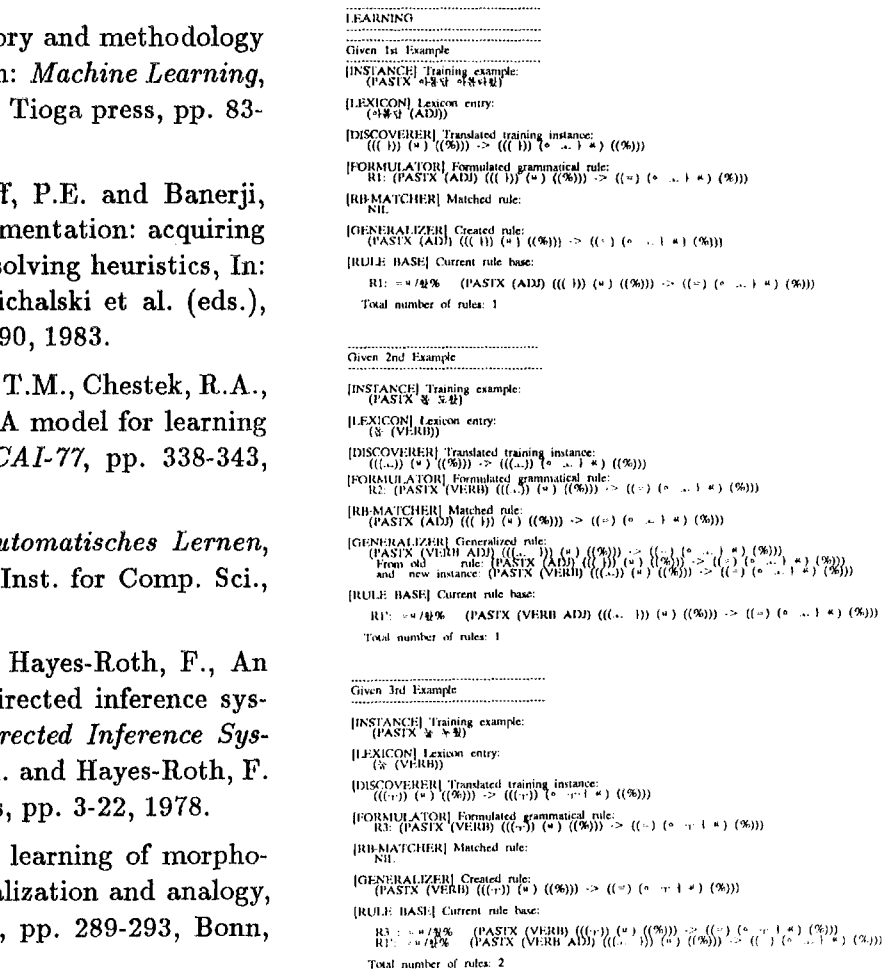
[17] Zhang, B.T. and Kim, Y.T., Machine learning of linguistic rules for multilingual morphological analysis and synthesis, *Journal of Korea Info. Sci. Soc.*, in press.

[18] Zhang, B.T., Yoo, S.I, and Kim, Y.T., XMAS: An eXpert in Morphological Analysis and Synthesis, *Proc. of '88 Spring Conf. on Arti. Intell.*, Korea Info. Sci. Soc., pp. 179-188, March 1988.

[19] Zhang, B.T. and Kim, Y.T., Morphological synthesis and analysis of Korean: a machine learning approach, *Proc. of 87 Automn Conf. of Korea Info. Sci. Soc.*, pp. 573-576, Sept. 1987.

Appendix

A Learning



B Analysis and Synthesis

```
----------------------------------------------------
SYNTHESIS
----------------------------------------------------

[INSTANCE] Synthesis problem:
   (PASTX 새롭)

[LEXICON] Lexicon entry:
   (새롭 (ADJ) ㅂ)

[TRANSLATOR] Translated problem instance:
   (PASTX (ADJ) (% ㅅ ㅐ ㄹ ㅗ ㅂ %) -> ?)

[MOS-MATCHER] Conflict set:
   ((PASTX (VERB ADJ) (((ㅗ ㅏ)) (ㅂ) ((%))) -> ((=) (ㅇ ㅗ ㅏ ㅆ) (%))))

[MOS-RESOLVER] Selected rule:
   (PASTX (VERB ADJ) (((ㅗ ㅏ)) (ㅂ) ((%))) -> ((=) (ㅇ ㅗ ㅏ ㅆ) (%)))

[GENERATOR] Synthesized output:
   새롭 + PASTX --> 새로왔


----------------------------------------------------
ANALYSIS
----------------------------------------------------

[INSTANCE] Analysis problem:
   (새로왔)

[TRANSLATOR] Translated problem instance:
   (? -> (% ㅅ ㅐ ㄹ ㅗ ㅇ ㅗ ㅏ ㅆ %))

[MOA-MATCHER] Conflict set:
   (((PASTX (VERB ADJ) (((ㅗ ㅏ)) (ㅂ) ((%))) -> ((=) (ㅇ ㅗ ㅏ ㅆ) (%))) (ㅅ ㅐ ㄹ ㅗ ㅂ)))

[MOA-RESOLVER] Selected rule(s):
   (((PASTX (VERB ADJ) (((ㅗ ㅏ)) (ㅂ) ((%))) -> ((=) (ㅇ ㅗ ㅏ ㅆ) (%))) 새롭 ((ADJ))))

[RECOGNIZER] Analysis output:
   새로왔 --> 새롭((ADJ)) + PASTX
```

C Part of rules learned by XMAS

C.1 English Plural

```
=/s%    (PLURAL (N) (((o f y w p t n m th c d k r a)) NIL ((%))) -> ((=) (s) (%)))
=/es%    (PLURAL (N) (((o ch sh x z s)) NIL ((%))) -> ((=) (e s) (%)))
=f/ves%    (PLURAL (N) (((a e l)) (f) ((%))) -> ((=) (v e s) (%)))
=fe/ves%    (PLURAL (N) (((i)) (f e) ((%))) -> ((=) (v e s) (%)))
=y/ies%    (PLURAL (N) (((p d b t)) (y) ((%))) -> ((=) (i e s) (%)))
=/x%    (PLURAL (N) (((u)) NIL ((%))) -> ((=) (x) (%)))
=on/a%    (PLURAL (N) (((n)) (o n) ((%))) -> ((=) (a) (%)))
=i/e=    (PLURAL (N) (((s)) (i) ((s))) -> ((=) (e) (=)))
=/e%    (PLURAL (N) (((a)) NIL ((%))) -> ((=) (e) (%)))
=us/i%    (PLURAL (N) (((g l e)) (u s) ((%))) -> ((=) (i) (%)))
=um/a%    (PLURAL (N) (((d t)) (u m) ((%))) -> ((=) (a) (%)))
=/ren%    (PLURAL (N) (((d)) NIL ((%))) -> ((=) (r e n) (%)))
=/en%    (PLURAL (N) (((x)) NIL ((%))) -> ((=) (e n) (%)))
=a/e=    (PLURAL (N) (((m)) (a) ((n))) -> ((=) (e) (=)))
=oo/ee=    (PLURAL (N) (((t g f)) (o o) ((s t))) -> ((=) (e e) (=)))
```

C.2 Past Tense of Korean

```
==ㅏㅎ/ㅐㅆ%    (PASTX (ADJ) (((ㅏㅗ) (ㄹㄱㄹ)) (ㅏㅎ) ((%))) -> ((= =) (ㅐㅆ) (%)))
==/ㅆ%    (PASTX (VERB) (((ㅅㄱ) (ㅏ)) NIL ((%))) -> ((= =) (ㅆ) (%)))
%=ㅗ/ㅗㅏㅆ%    (PASTX (VERB) (((%) (ㅂㅇ)) (ㅗ) ((%))) -> ((% =) (ㅗㅏㅆ) (%)))
==ㄷ/ㄹ았%    (PASTX (VERB) (((ㄷ) (ㅏ)) (ㄷ) ((%))) -> ((= =) (ㄹ ㅇ ㅏㅆ) (%)))
==ㄷ/ㄹ었%    (PASTX (VERB) (((ㅋ ㄷ) (ㅓ ㅡ)) (ㄷ) ((%))) -> ((= =) (ㄹ ㅇ ㅓㅆ) (%)))
==/렀%    (PASTX (VERB ADJ) (((ㄹ) (ㅡ)) NIL ((%))) -> ((= =) (ㄹ ㅓㅆ) (%)))
==ㅡ/랐%    (PASTX (VERB ADJ) (((ㅗ ㅏ) (ㄹ)) (ㅡ) ((%))) -> ((= =) (ㄹ ㅏㅆ) (%)))
==ㅡ/렸%    (PASTX (VERB) (((ㅡ) (ㄹ)) (ㅡ) ((%))) -> ((= =) (ㄹ ㅓㅆ) (%)))
==ㅂ/았%    (PASTX (VERB ADJ) (((ㄷㅈ) (ㅏㅗ)) (ㅂ) ((%))) -> ((= =) (ㅇ ㅗㅏㅆ) (%)))
==ㅂ/웠%    (PASTX (VERB ADJ) (((ㅈ ㄴ ㄷ ㅇ) (ㅜ ㅓ ㅕ)) (ㅂ) ((%))) -> ((= =) (ㅇ ㅜㅓㅆ) (%)))
==ㅅ/았%    (PASTX (VERB ADJ) (((ㅈ ㄴ) (ㅏ)) (ㅅ) ((%))) -> ((= =) (ㅇ ㅏㅆ) (%)))
==ㅅ/었%    (PASTX (VERB) (((ㅂ ㅈ) (ㅜ ㅡ)) (ㅅ) ((%))) -> ((= =) (ㅇ ㅓㅆ) (%)))
==/였%    (PASTX (VERB) (((ㅎ) (ㅏ)) NIL ((%))) -> ((= =) (ㅇ ㅕㅆ) (%)))
%=ㅜ/ㅕㅆ%    (PASTX (VERB) (((%) (ㅍ)) (ㅜ) ((%))) -> ((% =) (ㅕㅆ) (%)))
%=ㅡ/ㅕㅆ%    (PASTX (VERB) (((%) (ㅆ)) (ㅡ) ((%))) -> ((% =) (ㅕㅆ) (%)))
==/았%    (PASTX (VERB ADJ) (((ㅗ ㅏ) (ㅂㅅㄹㄱㄹㄱ)) NIL ((%))) -> ((= =) (ㅇ ㅏㅆ) (%)))
==/었%    (PASTX (VERB) (((ㅓ ㅣ ㅜ) (ㄷㅅㄹ)) NIL ((%))) -> ((= =) (ㅇ ㅓㅆ) (%)))
```