Unlexicalized Dependency Parser for Variable Word Order Languages based on Local Contextual Pattern

Hoojung Chung and Hae-Chang Rim

Department of Computer Science
Korea University, Seoul 136-701 Korea
{hjchung, rim}@nlp.korea.ac.kr

Abstract. We investigate the effect of unlexicalization in a dependency parser for variable word order languages and propose an unlexicalized parser which can utilize some contextual information in order to achieve performance comparable to that of lexicalized parsers. Unlexicalization of an early dependency parser makes performance decrease by 3.6%. However, when we modify the unlexicalized parser into the one which can consider additional contextual information, the parser performs better than some lexicalized dependency parsers, while it requires simpler smoothing processes, less time and space for parsing.

1 Introduction

Lexical information has been widely used to achieve a high degree of parsing accuracy, and parsers with lexicalized language models [1–3] have shown the state-of-the-art performances in analyzing English. Most of parsers developed recently use lexical features for syntactic disambiguation, whether they use a phrase structure grammar or a dependency grammar, regardless of languages they deal with.

However, some researchers recently insisted that the lexicalization did not play a big role in parsing with probabilistic context-free grammars (PCFG). [4] showed that the lexical bigram information does not contribute to the performance improvement of a parser. [5] concluded that the fundamental sparseness of the lexical dependency information from parsed training corpora is not helpful to the lexicalized parser, and proposed an accurate unlexicalized parsing model.

This is the story of analyzing fixed word order languages, e.g. English, with a phrase structure grammar. What about parsing other type of languages with other type of grammars, without lexical dependency information? For instance, can an unlexicalized dependency parser for languages with variable word order achieve high accuracy as the unlexicalized PCFG parser for English does?

This paper investigates the effect of the unlexicalization in a dependency parser for variable word order languages and suggests a new unlexicalized parser which can solve the problems of the unlexicalized dependency parser.

Table 1. Effect of unlexicalization. The lexicalized parser uses Equation (2), while the unlexicalized parser uses Equation (3)

	Lexicalized (F_1-score)	Unlexicalized (F_1-score)
Training Set	0.996	0.801
Testing Set	0.837	0.801

2 The Effect of Unlexicalizing Dependency Parser

It seems that lexicalization may play more role in dependency parsing for variable word order languages than in parsing fixed word order languages with a phrase structure grammar. There are some reasons: dependency parsers cannot use information on constituents because the grammar is based on the word unit, not on the constituent. Therefore, the disambiguation depends more on the lexical dependency between words. Secondly, since the word order is variable, which means there is less restriction, it requires higher level information such as semantic constraint or lexical preference to offset inexistency of word order information.

To investigate the effect of unlexicalization, we implemented [6]-style parser, which uses bigram lexical dependencies and distance measure for syntactic disambiguation. The parsing model for a sentence S is :

$$P(t|S) \approx \prod_{i=1}^{|S|-1} P(dep_i = h(i)|S) \tag{1}$$

where $|S|$ is a number of words in S, $h(i)$ is the modifyee of w_i, the ith word of S, and dep_i is a dependency relation from w_i to $w_{h(i)}$. The modifyees for each word are stated in the tree t, which is a set of modifyees. The probability of each dependency relation is :

$$P(dep_i = h(i)|S) \approx P(link(i, h(i)) = Yes|w_i\ w_{h(i)}\ \Delta_{i,j}) \tag{2}$$

$$link(x, y) = \begin{cases} Yes & \text{if } w_x \text{ modifies } w_y \\ No & else. \end{cases}$$

where Δ is a number of features to consider the distance between the two depending words. The unlexicalized model is induced by substituting part-of-speech (POS) tags t for all lexical words w in (2) :

$$P(dep_i = h(i)|S) \approx P(link(i, h(i)) = Yes|t_i\ t_{h(i)}\ \Delta_{i,j}) \tag{3}$$

We trained both model (2) and (3) on about 27,000 sentences and tested their performance on held-out testing data. The result is on Table 1.

Overfitting causes the lexicalized parser performs extremely well in the training data. On the testing set, unlexicalization hurts the parsing performance by

3.6% absolute, which is a sharper drop than the decrease by the unlexicalization of PCFG parser for English, which was reported in [4].

Despite the poor performance of the unlexicalized dependency parser for variable word order language, using unlexicalized parser have considerable advantages. First, we can simplify smoothing processes for estimating probabilities that are designed to alleviate lexical data sparseness problems. Consequently, it increases parsing speed. And eliminating lexical data reduces the space complexity.

So we designed a new unlexicalized parser that considers more contexts for syntactic disambiguation, yet can parse more accurately.

3 Revising the Unlexicalized Parser

We observed that even a variable word order language generates some fixed POS tag sequence pattern in a local context. Based on this observation, we use local contexts of modifier and modifyee in estimating the likelihood of dependency between the two POS tags, and the likelihood of a length of modification relation from the modifier. We use the Korean language[1], which allows variable word order, for explaining our ideas.

3.1 Word Dependency Probability with Local Context

In the research on a phrase structural parsing model for Korean [7], the outer contexts of constituents were found to be useful for syntactic disambiguation. We use similar method for our dependency parser. In other words, we consider outer contexts of a dependency relation, instead of the constituent, when we estimate the *word dependency probability*:

$$P(link(i,j) = Yes|w_i\ w_j\ \Phi_i \Phi_j) \approx P(link(i,j) = Yes|t_i\ t_j\ \Phi_i\ \Phi_j) \quad (4)$$

$$\approx P(link(i,j) = Yes|t_i\ t_j\ t_{i-1}\ t_{j+1}) \quad (5)$$

where Φ_i is contextual information of w_i. According to [7], considering two POS tags, one at the left and one at the right of a constituent, was sufficient for improving parsing performance. So we substitute a single POS tag for each context Φ. i.e. (5).

3.2 Modification Distance Probability based on Local Contextual Pattern

We observed that a word has the tendency to have a fixed modification distance in a certain context. Let's see an example with Figure 1. The word *na-ui*

[1] Readers who are unfamiliar with the Korean syntax may refer Appendix at the end of this paper for a brief introduction to the Korean syntax.

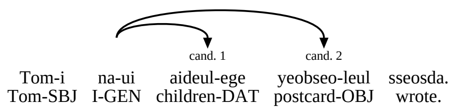


Fig. 1. The sentence means *Tom wrote a postcard to my children.* The word *na-ui* has two alternative modifyee candidates

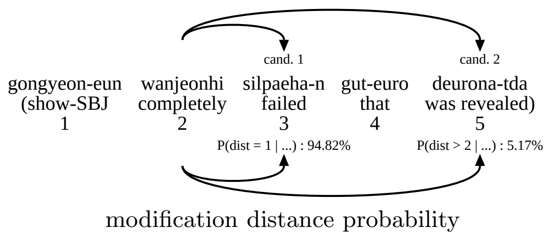


Fig. 2. The sentence interpreted as *It was revealed that the show was completely failed.* in English. The arcs at the bottom of the sentence show modification distance probability, which is proposed in this paper

(I-GEN[2]), which is a noun modifier, has two alternative noun modifyee candidates: *aideul-ege* (children-DAT) and *yeobseo-leul* (postcard-OBJ). Here, the first candidate is the correct modifyee for the modifier. It is well known to Korean users that the word ends with the morpheme *-ui* (genitive postposition) usually modifies the right next word. In other words, the word ends with the genitive marker *-ui* prefers modification distance of 1 in general context. Some rule-based or heuristic-based parsers encoded this preference into a rule for syntactic disambiguation.

Let's see another similar, but more complex example in Figure 2. The adverb *wanjeonhi* (completely) has two alternative modifyee candidates in this sentence. They are *silpaeha-n* (failed) and *derona-tda* (was revealed), and the former is the correct modifyee of the adverb. Finding the correct modifyee is tough in this case, even though we consider lexical or semantic information, because the lexical or semantic preference of the adverb *wanjeonhi* to both modifyee candidates are similar.

We define a *modification distance probability* to solve the problem. It is the likelihood of the preferred length of a modification relation for a certain modifier in a certain context, which reflects the following two preferences:

1. Whether a modifier prefers long distance modification or local (short distance) modification.

[2] SBJ, GEN, DAT, and OBJ stand for a subjective, genitive, dative and objective case marker, respectively.

2. If a modifier prefers local modification, which word in the local context is preferred as its modifyee.

The probability of a certain modification distance x for the given modifier word w_i and its surrounding context Φ_i is :

$$\begin{aligned} P(len = x \mid w_i\ \Phi_i) &\approx P(len = x \mid w_{i-m}\ \cdots\ w_{i+n}) \\ &\approx P(len = x \mid t_{i-m}\ \cdots\ t_{i+n}) \end{aligned} \tag{6}$$

where the constants m and n are empirically determined. The length of the modification relation x is calculated with the function $\Psi(ld)$, that is

$$\Psi(ld) = \begin{cases} ld & \text{if} \quad ld < k \\ long & \text{if} \quad ld \geq k \end{cases}, \qquad \Psi(ld) \in Dist = \{1, \cdots, k-1, long\}.$$

when ld is a linear distance between two depending words. A constant k is the yardstick to decide whether a dependency relation is short or long. We named (6) the modification distant probability and used this probability instead of using the distance measure as in (3).

To see an example that uses this probability, revisit Figure 2 which is showing the probabilities for each modification distance[3]. The probabilities are calculated with the modification distance probability. [4].

$$\begin{aligned} P(len = 1 | mag\ pvg\text{-}etm\ nbn\text{-}jca) &= 94.82 \\ P(len = 2 | mag\ pvg\text{-}etm\ nbn\text{-}jca) &= 0 \\ P(len = long | mag\ pvg\text{-}etm\ nbn\text{-}jca) &= 5.17 \end{aligned}$$

3.3 The Probabilistic Dependency Parsing Model

A dependency parsing model estimates the probability of a parsing tree t for a given sentence S.

$$P(t|S) \approx \prod_{i<|S|} P(dep_i = h(i)|S) \tag{7}$$

We assume a dependency relation depends only on the two words that is linked by the relation and their local context. This makes (7) become (8).

$$P(dep_i = h(i)|S) \approx \frac{P(link(i,h(i)) = Yes\ len = \Psi(h(i)-i)|w_i \Phi_i w_{h(i)} \Phi_{h(i)})}{\sum_{k>i, x \in \{Yes, No\}, y \in Dist} P(link(i,k) = x\ len = y | w_i \Phi_i w_{h(i)} \Phi_{h(i)})} \tag{8}$$

$$\begin{aligned} &= P(link(i,h(i)) = Yes | w_i\ \Phi_i\ w_{h(i)}\ \Phi_{h(i)}) \\ &\quad \cdot P(len = \Psi(h(i)-i) | link(i,h(i)) = Yes\ w_i\ \Phi_i\ w_{h(i)}\ \Phi_{h(i)}) \end{aligned} \tag{9}$$

[3] The probabilities for distance 2 is not shown in the figure, and the probabilities for *long* distance modification is marked as *dist* > 2.

[4] *m*,*n*, and *k* are 0, 2, and 3 here. *mag*, *pvg-etm* and *nbn-jca* are POS tags for w_1, w_2, and w_3 in Figure 2.

Since the denominator of (8) is constant, and by a using chain rule, we can get (9). The latter term of (9) is a probability of a certain modification length. Since we assume that the modification length only depends on a modifier and its context, and since we exclude all lexical information from the model, the whole parsing model becomes as :

$$\begin{aligned} P(dep_i = h(i)|S) &\approx P(link(i, h(i)) = Yes|w_i\ \Phi_i\ w_{h(i)}\ \Phi_{h(i)}) \\ &\quad \cdot P(len = \Psi(h(i) - i)|w_i\ \Phi_i) \\ &\approx P(link(i, h(i)) = Yes|t_i\ \Phi_i\ t_{h(i)}\ \Phi_{h(i)}) \cdot P(len = \Psi(h(i) - i)|t_i\ \Phi_i) \end{aligned}$$

As you see, it becomes a product of the probability of word dependency between modifier and modifyee, and the probability of length of modification relation for the modifier based on the local contextual pattern of it.

4 Related Works

There has been little work done on unlexicalizing the parsing model. Instead, many studies tried to combine various features including lexicalized information. The distance measure is one of widely used feature in dependency parsing. As shown earlier, [6] proposed a statistical parsing model for English, based on bigram lexical dependencies and distance between the two depending words. [8–10] proposed similar models for parsing Korean and Japanese.

However, using the distance features in the conditional part of the probability equation[5] as [6] assumes that the dependency relation of a certain length is different from dependency relations with different lengths. This assumption may cause sparse data problem in estimating word dependencies for the languages allowing variable word order. The sparseness would be more serious for the model that uses lexical dependencies, such as [9, 10].

There were another approaches that used modification distance as we do, but in a different way. [11, 12] utilized handcrafted HPSG for dependency analysis of Japanese. HPSG is used to find three alternative modifyee candidates: the nearest, the second nearest, and the farthest candidates from a certain modifier. Then, the probabilistic models choose an appropriate modifyee among three candidates. These models seem to work well for Japanese, however, it is doubtful that the parsing models can be applied to other languages well. The parsing models are restricted to consider only three head candidates at most, based on the statistics from Japanese corpora. So they may fit for Japanese parsing but would cause problems for parsing other languages. And these approaches require handcrafted grammars which usually demands excessive manual labors. These features can be obstacles when someone uses these models to develop a new parser for other languages.

[5] See Equation (2)

In contrast, our model splits probability of a dependency relation into the word dependency probability and the modifying distance probability to alleviate sparse data problem. And proposed model does not depend on language specific features, and does not require any language-specific manual rules – such as heuristic constraints or HPSG – either. So it can be adapted to other languages easily. Of course, our model does not ignore any grammatically correct modifyee candidates at all, while [11] and [12] ignore less likely grammatical modifyee candidates.

5 Experimental Results

We implemented a probabilistic parser that uses the proposed parsing model and performed some experiments to evaluate our method empirically. We used a backward beam search algorithm, which was originally designed to analyze Japanese with dependency probabilities[13].

The parser was trained on 27,694 sentences and tested on heldout 3,386 sentences of dependency tagged sections of KAIST LANGUAGE RESOURCES[14]. All sentences are POS tagged, and this information was used as an input for the parser.

5.1 Deciding length of modification relation with the Modification distance probability

First of all, we evaluate our assumption, that is *length of a modification relation can be determined by a modifier and its local contextual pattern.* To do this, we made a classifier using the modification distance probability that models our assumption statistically. The classifier decide the length of a modification relation for the given modifier t_i and its context Φ_i.

$$\begin{aligned}\text{modification distance} &= argmax_{d \in Dist} P(len = d \mid t_i \; \Phi_i) \\ &= argmax_{d \in Dist} P(len = d \mid t_{i-n} \cdots t_i \cdots t_{i+m})\end{aligned} \tag{10}$$

We experiment with changing n and m from 0 to 2, while k changes from 1 to 3. We used F_1 measure for evaluating the classifier. The experimental result is on Table 2. It tells that considering wider context does not always induce more accurate classification. The best result is acquired when m and n are 0 and 2. This means the left context hardly affect the performance of deciding modification distances[6]. Right context size bigger than 3 does not help the classification too.

Meanwhile, the performance of the classifier increases as the value of k decreases. It is because a smaller k decreases the number of distance class, which is k, and classification becomes easier for smaller and more generic class. We selected the values for m, and n as 0 and 2 through this experiment, but could not decide the value for k. Although the performance of classifier with bigger k

[6] [15] reported that similar characteristic is observed for Japanese too. Based on his experiment with humans, it is true more than 90% of the time for Japanese.

Table 2. Experimental result (in F_1-score) for the modification distance classifier, with various m (left context size) , n (right context size) , and k (class size) values

context size		*Dist*		
m	n	{1, long} (k=2)	{1,2, long} (k=3)	{1,2,3, long} (k=4)
1	0	0.916	0.750	0.722
2	0	0.787	0.747	0.721
0	1	0.916	0.855	0.817
1	1	0.898	0.835	0.799
2	1	0.847	0.793	0.761
0	2	0.926	0.879	0.838
1	2	0.894	0.851	0.814
2	2	0.816	0.775	0.748
0	3	0.872	0.831	0.800
1	3	0.831	0.793	0.770
2	3	0.794	0.755	0.731

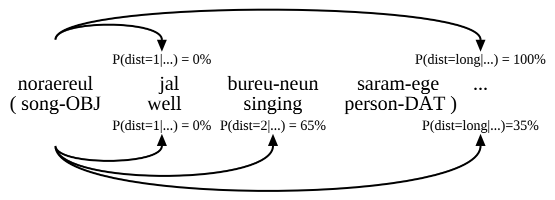
is worse, it might be more helpful for the parser to have probabilities for more subdivided distances.

Figure 3 is an example that shows the effect of different k makes. The upper arcs show modification distance probability when k is 2. The lower arcs show the probability when k is 3. When k is 2, the only information we can get from the modification distance probability is that the modifier *norae-reul* (song-OBJ) does not modify the next word *jal* (well) . However, this independency can be known by simple dependency rule probability because an object noun never modifies an adverb. So the modification distance probability is not helpful when k is 2. However, when the value of k is 3, the modification distance probability assigns higher probability for length 2 modification relation, which cannot be considered with the simple dependency probability. So we will not determine the value k here, but use all k for the following experiments.

5.2 Experiment with Richer Context

We used modification distance probability and added a little more context information to the bare word dependency probability to achieve higher parser performance. Here, we are going to evaluate the effect of information we have added to the vanilla dependency probability.

To evaluate the performance of the parser, we used arc-based F_1 measure and sentence-based exact matching rate. The results are shown in Table 3. It shows both additional contextual information (OC & MDP) contribute to the parser performance. Interesting point here is the increase of parser performance as k gets bigger. In the previous experiments, classifier performs worse for bigger k values. This change is due to the effect of larger k, which is more helpful for a parser to decide appropriate modifyee as we discussed in the previous experiment with Figure 3. The table also shows that using the modification distance probability when $k = 2 : Dist = \{1, long\}$



when $k = 3 : Dist = \{1, 2, long\}$

Fig. 3. Comparison of the modification distance probability from the word *norae-reul*, when $k = 2$ and $k = 3$. As k gets bigger, the modification distance probability may be more helpful.

Table 3. Effect of considering outer context and modification distance. BM stands for the unlexicalized parser with the model (3). Δ, OC, and MDP stands for distance measure used in [6], outer context, and modification distance probability

	Measures	BM	BM	BM-Δ	BM-Δ+MDP			BM-Δ+OC+MDP		
			-Δ	+OC	k=2	k=3	k=4	k=2	k=3	k=4
Training Set	Arc Prec.	0.801	0.748	0.783	0.812	0.824	0.831	0.854	0.862	0.865
	Exact Match	0.212	0.181	0.242	0.246	0.273	0.291	0.336	0.350	0.354
Testing Set	Arc Prec.	0.801	0.747	0.762	0.807	0.819	0.826	0.842	0.853	0.856
	Exact Match	0.200	0.178	0.223	0.225	0.246	0.266	0.310	0.321	0.322

(BM-Δ+MDP) is better than using the distance measure as in [6] (BM). This means using the distance measure as in our paper is better than method in others.

5.3 Comparison with Other Parsers

We compared our parser with some lexicalized parsers. They are parsers from [6] and [11] [7]. The results are shown in Table 4.

In the training set, the parser from [6] shows almost 100% F_1 score It is because the parser is highly lexicalized. The parser from [11] using triplet/quadruplet model assumes that a modifyee of a word is one among the nearest, second nearest, or the last modifyee candidate. Unfortunately, according to our investigation, only 91.48% of modifyees are among the three candidates in the training data. This restriction causes the poor performance in the training data even the model is lexicalized.

[7] [11] parser requires handcrafted grammar(HPSG). Instead of HPSG, we used a set of dependency rules whose frequency is more than one in the training corpus as the grammar. (e.g. *Treebank grammar* [16])

Table 4. Result of the comparison with other lexicalized models. Many statistical parsing models dealing with distance measure, such as [9, 10], resemble the model of [6]

	Measures	Parser from [6]	Parser from [11]	This Paper (k=4)
Training Set	Arc F_1 score	0.996	0.908	0.865
	Exact Matching	0.966	0.555	0.354
Testing Set	Arc F_1 score	0.837	0.843	0.856
	Exact Matching	0.256	0.303	0.322

In contrast, our model performs better than other lexicalized models in the experiment for the testing data. The improvements (+1.9% from [6]'s and +1.3% from [11]'s, absolute) in the arc-level performance are statistically meaningful.

This result is showing that the lexical dependency information may be useful for accurate parsing, but the proper use of other contextual information may be more helpful[8]. And it means the method we used to deal with the length of modification relation is effective for syntactic disambiguation.

6 Conclusion

We investigate the effect of unlexicalization of dependency parser for variable word order languages and propose a new parser, which is unlexicalized to keep the parser light, simple, and robust from the data sparseness problem. It utilizes some POS-level information to keep accuracy high as lexicalized parsers. In particular, we suggest using the modification distance probability to reflect the preference on a length of a modification relation for a given modifier and its contextual pattern. The experimental results show our model outperformed other lexicalized models for parsing Korean, which is a free word order language. Since it does not use any language specific predefined rules, the proposed parser can be easily adapted to other variable word order languages.

We don't say lexical information is unworthy. However, ignoring lexical information in a parser can give some advatages – simpler parser implementation, smaller disk space and shorter processing time – without sacrificing much accuracy, and this advatages may be useful for some cases, i.e. developing a parser for the system with limited memory size or processing speed.

We found out that the lexicalization plays a bigger role in parsing with probabilistic dependency grammar, but we haven't deeply investigated the cause of it yet. We will continue to investigate it. And there are some works that does

[8] In addition, our unlexicalized parser requires much smaller size of frequency data for estimating the probabilities. While the lexicalized parsers require 643M ([6]'s) and 540M, ([11]'s) bytes for storing the data, our parser uses only 18M bytes of data. We haven't trie to optimize the data structure. But taking that into account, the huge difference of the required resource size gives some ideas why unlexicalized parser is preferable.

not assume independency between dependency relations, such as [17]. We are going to reconstruct our parsing model without the independency assumption in the future.

References

1. Collins, M.: Head-Driven Statistical Models for Natural Language Parsing. PhD thesis, University of Pennsylvania (1999)
2. Charniak, E.: A maximum-entropy-inspired parser. Technical Report CS-99-12, Department of Computer Science, Brown University (1999)
3. Charniak, E.: Immediate-head parsing for language models. In: Meeting of the Association for Computational Linguistics. (2001) 116–123
4. Gildea, D.: Corpus variation and parser performance. In: Proceedings of Conference on Empirical Methods in Natural Language Processing. (2001)
5. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st Meeting of the Association for Computational Linguistics. (2003) 310–315
6. Collins, M.J.: A new statistical parser based on bigram lexical dependencies. In: Proceedings of the 34th Annual Meeting of the ACL. (1996)
7. Lee, K.J.: Probabilistic Parsing of Korean based on Language-Specific Properties. PhD thesis, Dept. of Computer Science. KAIST (1997)
8. Haruno, M., Shirai, S., Ooyama, Y.: Using decision trees to construct a practical parser. In: Proceedings of COLING-ACL 98. (1998) 505–512
9. Kim, H., Seo, J.: A statistical Korean parser based on lexical dependencies. In: Spring Proceedings of Conference on Korea AI Society. (1997)
10. Uchimoto, K., Sekine, S., Isahara, H.: Japanese dependency structure analysis based on maximum entropy models. In: Proceedings of 13th EACL. (1999) 196–203
11. Kanayama, H., Torisawa, K., Mitsuichi, Y., Tsujii, J.: Statistical dependency analysis with an HPSG-based Japanese grammar. In: Proceedings of 5th Natural Language Processing Pacific Rim Symposium. (1999) 138–143
12. Kanayama, H., Torisawa, K., Mitsuichi, Y., Tsujii, J.: A hybrid Japanese parser with an hand-crafted grammar and statistics. In: Proceedings of the COLING 2000. (2000) 411–417
13. Sekine, S., Uchimoto, K., Isahara, H.: Backward beam search algorithm for dependency analysis of japanese. In: Proceedings of the COLING 2000. (2000) 745–760
14. Choi, K.S.: KAIST Language Resources v.2001. Result of Core Software Project from Ministry of Science and Technology, Korea(http://kibs.kaist.ac.kr) (2001)
15. Sekine, S.: Japanese dependency analysis using a deterministic finite state transducer. In: Proceedings of the COLING 2000. (2000) 761–767
16. Charniak, E.: Tree-bank grammars. Technical Report CS-96-02, Department of Computer Science, Brown University (1996)
17. Seo, K.J., Nam, K.C., Choi, K.S.: A probabilistic model of the dependency parse for the variable-word-order languages by using ascending dependency. Computer Processing of Oriental Languages **12** (1999) 309–322

Appendix

Brief Introduction to Korean Syntax

Two prominent characteristics of Korean are agglutinative morphology, and rather free word order with explicit case marking [7].

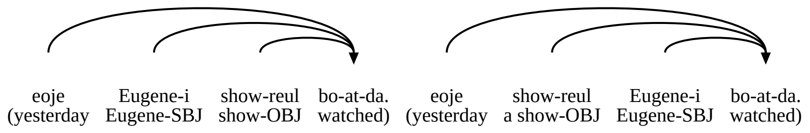


Fig. 4. Dependency trees for Korean sentences which have identical meaning, *Eugene watched a show yesterday.*

Korean is an agglutinative language, in which a word[9] is in a composition of more than one morpheme, in general. There are two types of morpheme: a content morpheme and a functional morpheme. A content morpheme contains the meaning of the word, while a functional morpheme plays a role as a grammatical information marker, which indicates a grammatical role, tense, modality, voice, etc. of the word.

The order of words is relatively weak in Korean compared to the fixed-order languages such as English. The grammatical information conveyed from a functional morpheme makes a word order be free. The following example is a Korean sentence consists of 4 words. Let's see a simple example[10] .

eoje	*Eugene-i*	*show-reul*	*bo-at-da.*
yesterday	Eugene-SBJ	a show-OBJ	watched

Eugene watched a show yesterday.

The second word in the sentence is *Eugene-i*. It consists of a content morpheme *Eugene* and a functional morpheme *i* which is a subject case marking postposition. The sentence can be rewritten as :

eoje	*show-reul*	*Eugene-i*	*bo-at-da.*
yesterday	a show-ACC	Eugene-NUM	watch-PAST-END

Eugene watched a show yesterday.

Though the subject and the object exchange their position, the two sentences have identical meaning. Because of this property of Korean, dependency grammar is widely used for analyzing syntactic structure of the Korean language.

The grammatical relation of a dependency relation can be specified by the functional morpheme of the modifier for the most case, selecting modifyee word for the modifier is the main concern for dependency parsing with Korean. Figure 4 shows dependency structure trees for the Korean sentences shown above.

[9] The exact term for the word is *eojeol*. However we use the term *word* for easier understanding.

[10] SBJ and OBJ stand for subjective and objective case.