Clustering for Obtaining Syntactic Classes of Words from Automatically Extracted LTAG Grammars

Tadayoshi Hara[†], Yusuke Miyao[†], Jun'ichi Tsujii[†‡]
†University of Tokyo ‡CREST, JST (Japan Science and Technology Corporation)

1. Introduction

We propose a method for obtaining syntactic classes of words from a lexicalized tree adjoining grammar (LTAG: Schabes, Abeillé and Joshi (1988)) automatically extracted from a corpus. Since *elementary trees* in LTAG grammars represent syntactic roles of a word, we can obtain syntactic classes by clustering words having the similar elementary trees. With our method, automatically extracted LTAG grammars will be arranged according to the syntactic classes of words, and the grammars can be improved from various points of view. For example, we can improve the coverage of the grammars by supplementing to a word the elementary trees of the syntactic class of the word.

An LTAG grammar consists of *elementary trees*, which determine the position where the word can be put in a sentence, that is, an elementary tree corresponds to a certain syntactic role. Hence, a syntactic class of a word is represented as a set of elementary trees assigned to the word. Since the words of the same syntactic class are expected to have similar elementary trees, we can obtain syntactic classes by clustering words having similar sets of elementary trees.

We applied our clustering algorithm to an LTAG grammar automatically extracted from sections 02–21 of the Penn Treebank (Marcus, Santorini and Marcinkiewicz (1994)), and investigated the obtained clusters with changing the number of clusters. We successfully obtained some of the clusters that correspond to certain syntactic classes. On the other hand, we could not obtain some clusters, such as the one for ditransitive verbs, and obtained the clusters that we could not associate clearly with syntactic classes. This is because our method was strongly affected by the difference in the number of words in each part-of-speech class. We concluded that, although our clustering method needs to be improved for practical use, it is effective to automatically obtain syntactic classes of words.

The XTAG English grammar (The XTAG Research Group (1995)) is a handmade LTAG grammar which is arranged according to syntactic classes of words, "*tree families*." Each tree family corresponds to a certain subcategorization frame, and determines elementary trees to be assigned to a word. Thanks to the tree families, the XTAG grammar is independent of a corpus. However, it needs considerable human effort to manually construct such a grammar.

Automatically extracted LTAG grammars are superior to manually developed grammars in the sense that it takes much less costs to construct the grammars. Chiang (2000) and Xia (1999) gave the methods of automatically extracting LTAG grammars from a bracketed corpus. They first decided a *trunk* path of the tree structure of a bracketed sentence, and the relationship (substitution or adjunction) between the trunk and branches. The methods then cut off the branches of them according to the relationship. Because the sentences used for extraction are in real-world texts, extracted grammars are practical for natural language processing. However, automatically extracted grammars are not systematically arranged according to syntactic classes their *anchors* belong to, like the XTAG grammar. Because of this, automatically extracted grammars tend to be strongly dependent on the corpus. This limitation can be a critical disadvantage of such extracted grammars when the grammars are used for various applications. Then, we want to arrange an extracted grammar according to the syntactic classes of words, without loosing the benefit for the cost.

Chen and Vijay-Shanker (2000) proposed the solution to the issue. To improve the coverage of an extracted LTAG grammar, they classified the extracted elementary trees according to the tree families in the XTAG English grammar. First, the method searches for a tree family that contains an *elementary tree template* of extracted elementary tree *et*. Next, the method collects other possible tree templates in the tree family and makes elementary trees with the anchor of *et* and the tree templates. By using tree families, the method can add only proper elementary trees that correspond to the syntactic class of anchors. Chen and Vijay-Shanker (2000) applied this method to an extracted LTAG grammar, and showed the improvement of the coverage of the grammar. Although their method showed the effectiveness of arranging a grammar according to syntactic classes, the method depends on

© 2002 Tadayoshi Hara, Yusuke Miyao, and Jun'ichi Tsujii. *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+6)*, pp. 227–233. Universitá di Venezia.

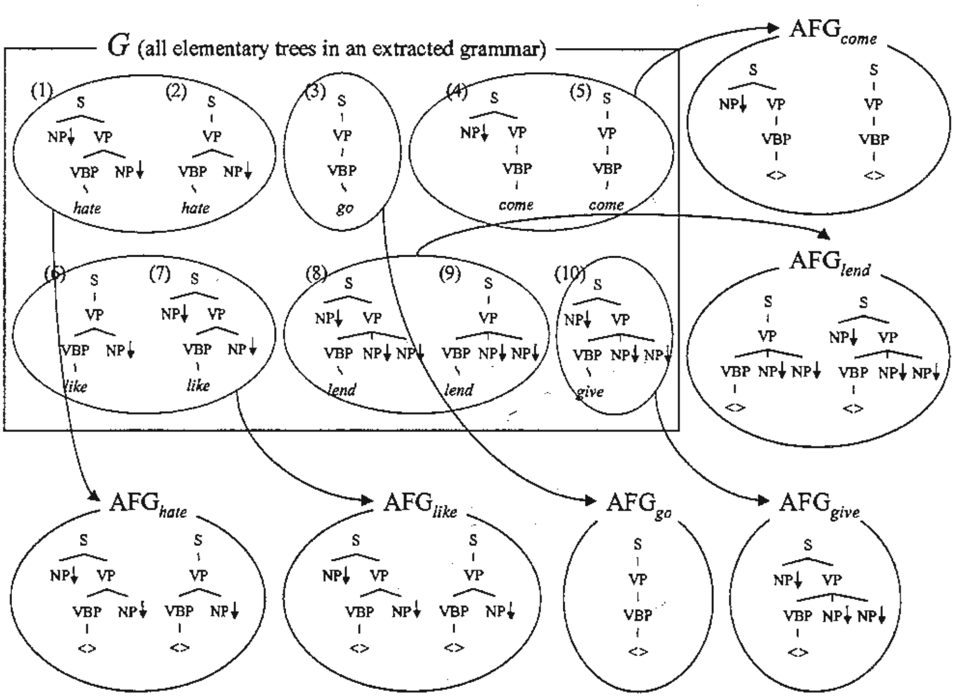


Figure 1: Obtaining AFGs from an extracted LTAG grammar

a manually developed grammar that takes considerable effort to construct. Since our method can automatically obtain syntactic classes, we can obtain this benefit without the considerable cost.

2. Methods

In this section, we give a method for automatically obtaining syntactic classes of words from an automatically extracted LTAG grammar. We first give the idea for our method, and then trace each step of our method with examples and formalization.

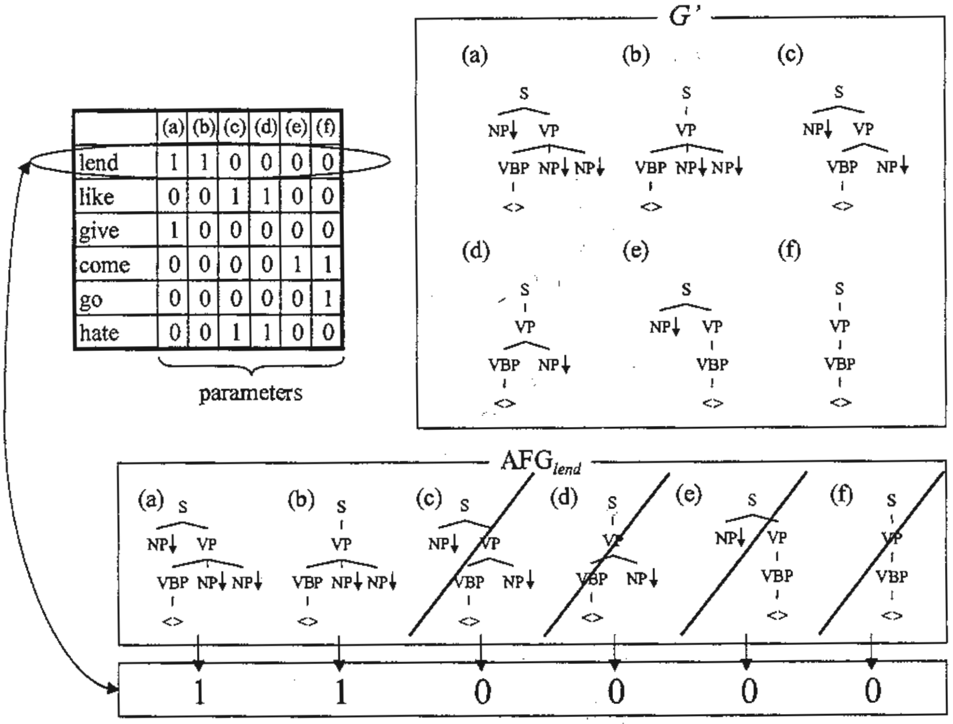
In LTAG grammars, syntactic roles are represented as elementary trees, and as in the XTAG grammar, a syntactic class of a word determines elementary trees to be assigned to the word. Given set W of words and set T of elementary tree templates, LTAG grammar G is defined as a subset of the product of them $G \subset T \times W$ that satisfies the following equation.

$$G = \{(t, w) | w \in W, t \in F(s(w))\} \tag{1}$$

where function s gives a syntactic class of word w, and function F gives a set of tree templates allowed by the syntactic class. Handmade LTAG grammars follow this formalization, for example, in the XTAG grammar, syntactic classes are represented as “tree families.” However, automatically extracted LTAG grammars are not arranged according to syntactic classes, and lack elementary trees that do not appear in the training corpus. Therefore, we need to obtain s and F automatically.

The idea to achieve this task can be derived from the equation 1. From this formalization, we can see that words of the same syntactic class should have the same set of elementary trees. This indicates that, even when the grammar lacks some elementary trees, we can obtain syntactic classes by collecting words having the similar elementary trees. To achieve this, we apply a clustering method. First, we make *anchor's feature groups (AFGs)*, which represent possible syntactic roles of the word, and are objects for clustering. Next, we apply a clustering method to collect similar AFGs, and finally, we interpret obtained clusters.

The first step in our method is to make AFGs. An AFG for a word is a set of all tree templates assigned for the word. Figure 1 shows an example of AFGs. For example, an AFG for “*lend*” can be obtained by collecting all elementary trees for ”*lend*” in the grammar, that is (8) and (9), and removing the anchor ”*lend*” from them. The obtained tree templates correspond to the syntactic roles, for example, a declarative and an imperative for



	(a)	(b)	(c)	(d)	(e)	(f)
lend	1	1	0	0	0	0
like	0	0	1	1	0	0
give	1	0	0	0	0	0
come	0	0	0	0	1	1
go	0	0	0	0	0	1
hate	0	0	1	1	0	0

Figure 2: Parameters of AFGs for clustering

"ditransitive verbs." Formally, an AFG for word w is defined as follows.

$$A_G(w) = \{t | t \in T, (t, w) \in G\} \tag{2}$$

Suppose that we have "perfect" LTAG grammar G_P which consists of all elementary trees for composing syntactically correct sentences. From equation 1 and 2, we get

$$\forall w \in W \; F(s(w)) = A_{G_P}(w)$$

Therefore, we can obtain a syntactic class of word w from the AFG for w.

$$\forall w \in W \; s(w) = F^{-1}(A_{G_P}(w))$$

However, automatically extracted LTAG grammar G_E lacks some elementary trees, because a corpus for training does not contain all elementary trees of words[1]. Therefore, we cannot obtain syntactic classes of words just as above. However, we can assume that an extracted LTAG grammar is very similar to a perfect LTAG grammar, and an AFG for a word in G_E should be very similar to an AFG for the word in G_P, and not to other AFGs in G_P at all. Formally,

$$\operatorname*{argmin}_{A_{G_P}(w')} d(A_{G_E}(w), A_{G_P}(w')) = A_{G_P}(w) = F(s(w)) \tag{3}$$

Function d gives a distance measure that indicates how given two sets are different. In addition, we can see that the words of the same syntactic class should have similar AFGs. Therefore, we can find the syntactic class by collecting words having similar AFGs.

In order to collect similar AFGs, we next apply a clustering method, *the K-means* (MacQueen (1967)), which groups "*objects*" close to each other. The distance between two objects is given by Euclidean distance between the two "*parameters*" of the objects. In our method, the "*object*" for clustering is a word characterized by a certain

[1] Actually, an extracted LTAG grammar may contain some improper elementary trees which can not make a syntactically correct sentence. But in this discussion, we consider that there are not any such elementary trees in the grammar.

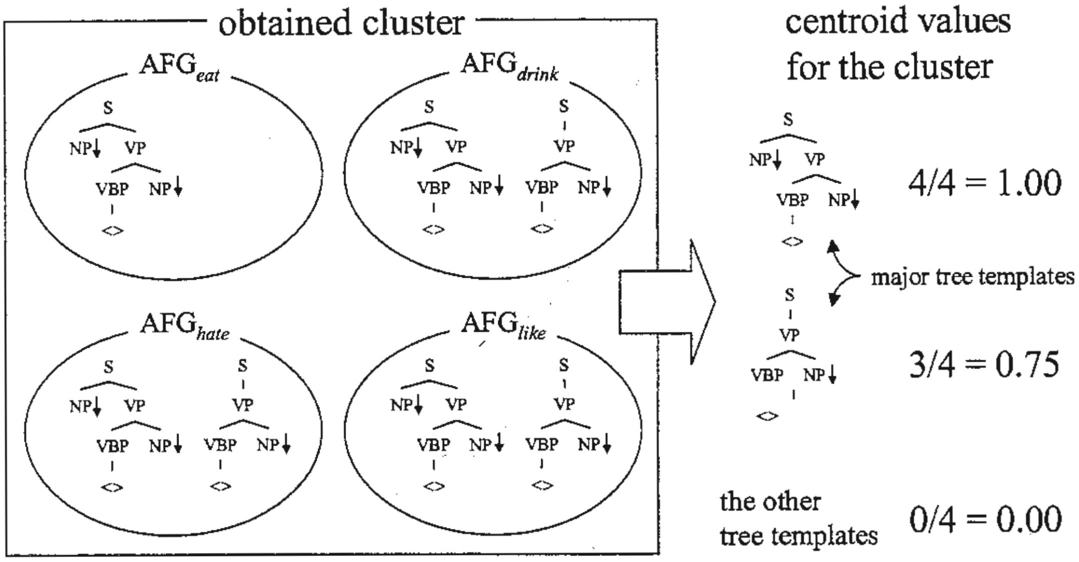


Figure 3: Centroid values for an obtained cluster

AFG, and the "*parameters*" for it are binary values for whether each tree template in the grammar is in the AFG or not. A table in Figure 2 shows example parameters. The grammar is the same as the example in Figure 1. A sequence of 0s and 1s in each row is a set of parameters for an AFG. For example, the AFG for "*lend*" contains tree templates (a) and (b), and therefore parameters for the two tree templates are both 1. The other parameters for AFG "*lend*" are all 0, because it does not contain the other tree templates in the extracted LTAG grammar.

The relation between obtained clusters and syntactic classes can be formalized as the following. The equation 3 shows our assumption that leads to the clustering method. We assume that the distance measure d of two sets, $T', T'' \subset T$ can be given as follows:

$$d(T', T'') = ||\vec{p}(T') - \vec{p}(T'')||$$

$$\vec{p}(T') = (x_i) \text{ where } x_i = \begin{cases} 1 & if\ t_i \in T' \\ 0 & otherwise \end{cases}$$

The vector $\vec{p}(T')$ is a parameter for AFG T'. According to the definition of d, equation 3 can be rewritten as follows.

$$\underset{A_{G_P}(w')}{\text{argmin}}\ ||\vec{p}(A_{G_E}(w)) - \vec{p}(A_{G_P}(w'))|| = A_{G_P}(w) \tag{4}$$

On the other hand, by clustering AFGs with parameters as defined above, the K-means algorithm makes clusters that satisfy the following equation:

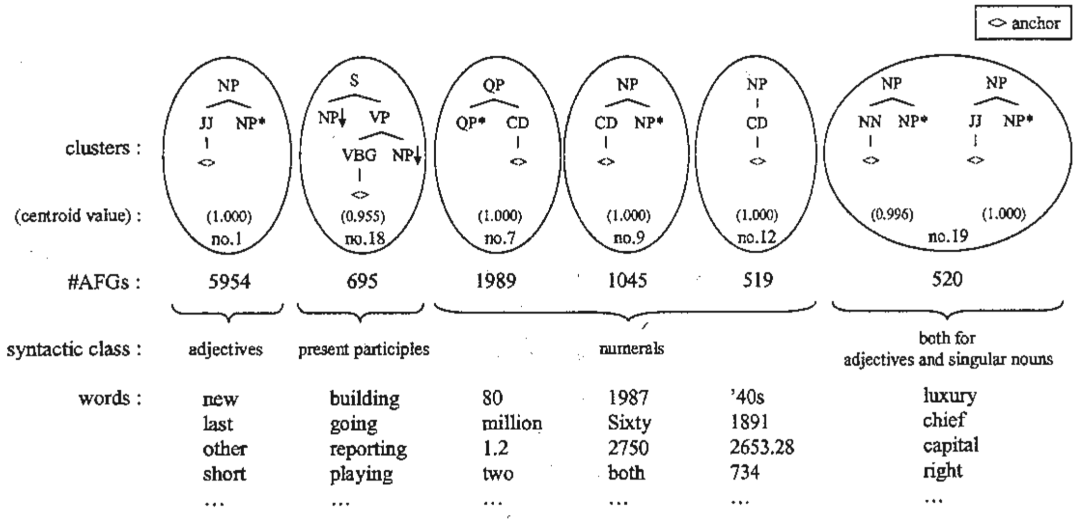
$$\underset{centroid(c')\ \text{s.t.}\ c' \in C}{\text{argmin}}\ ||\vec{p}(A_{G_E}(w)) - \vec{p}(centroid(c'))|| = centroid(c) \text{ s.t. } A_{G_E}(w) \in c \tag{5}$$

where C is the set of all obtained clusters. *centroid*(c) is a "*centroid*" of the cluster c, which is an imaginary object with the average of the parameters for objects in c:

$$\vec{p}(centroid(c)) = \frac{\sum_{A_{G_E}(w) \in c} \vec{p}(A_{G_E}(w))}{|c|}$$

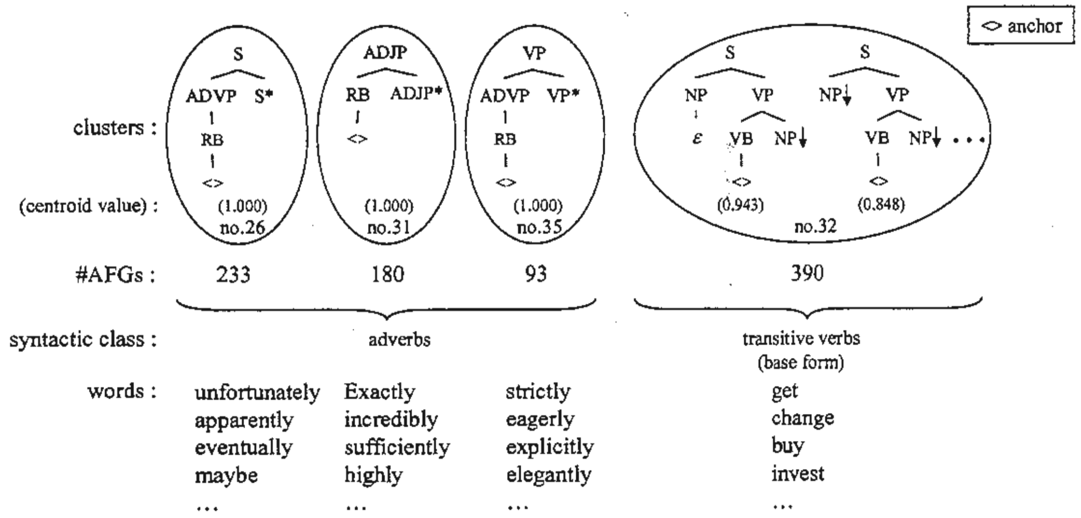
Comparing the equations 4 and 5, we can consider that *centroid*(c) corresponds to $A_{G_P}(w)$. This suggests that the obtained clusters of words correspond to sets of words which are in the same syntactic class.

At the last, we need to interpret obtained clusters. After the clustering, the method obtains words that have similar sets of tree templates, and the tree templates will indicate a syntactic class of the words. In particular, the set of tree templates common to most of those words are certain to correspond to the syntactic class of the words. Such



<> anchor
clusters :
(centroid value) :
#AFGs :
syntactic class :
words :
NP JJ NP* <> (1.000) no.1
S NP↓ VP VBG NP↓ <> (0.955) no.18
QP QP* CD <> (1.000) no.7
NP CD NP* <> (1.000) no.9
NP CD <> (1.000) no.12
NP NN NP* <> (0.996) NP JJ NP* <> (1.000) no.19
5954 695 1989 1045 519 520
adjectives
present participles
numerals
both for adjectives and singular nouns
new last other short ...
building going reporting playing ...
80 million 1.2 two ...
1987 Sixty 2750 both ...
'40s 1891 2653.28 734 ...
luxury chief capital right ...

Figure 4: Clusters generated in the case of 20 clusters



<> anchor
clusters :
(centroid value) :
#AFGs :
syntactic class :
words :
S ADVP S* RB <> (1.000) no.26
ADJP RB ADJP* <> (1.000) no.31
VP ADVP VP* RB <> (1.000) no.35
S NP VP ε VB NP↓ <> (0.943) S NP↓ VP VB NP↓ <> (0.848) ... no.32
233 180 93 390
adverbs
transitive verbs (base form)
unfortunately apparently eventually maybe ...
Exactly incredibly sufficiently highly ...
strictly eagerly explicitly elegantly ...
get change buy invest ...

Figure 5: Newly classified syntactic classes in the case of 40 clusters

tree templates can be obtained by interpreting each element of the parameter for the centroid, a *centroid value*, as the probability that each tree template corresponds to a part of a syntactic class. We call such tree templates "*major tree templates*", and use them to associate clusters to syntactic classes of words. The left-hand side of Figure 3 shows an example of an obtained cluster. In this cluster, major tree templates represent the syntactic roles of a "transitive verb," and therefore, we can interpret the cluster as a "transitive verb" class.

3. Experiments

Our clustering algorithm was applied to an LTAG grammar automatically extracted from sections 02–21 of the Penn Treebank (Marcus, Santorini and Marcinkiewicz (1994)). The grammar is extracted by an algorithm similar to the one in Xia (1999). From 39,598 sentences, 2,571 elementary tree templates are extracted for 43,030 words. Accordingly 43,030 AFGs were obtained by our method. We then classified the obtained AFGs into clusters. Since we have no knowledge how many clusters are suitable for this task, we varied the number of clusters over 20, 40, 60, 80, and 100.

Increasing the number of clusters, we could observe how the classes of syntactic roles were being obtained in a more detailed way. In the case of 20 clusters shown in Figure 4, for example, the cluster No. 1 was for the

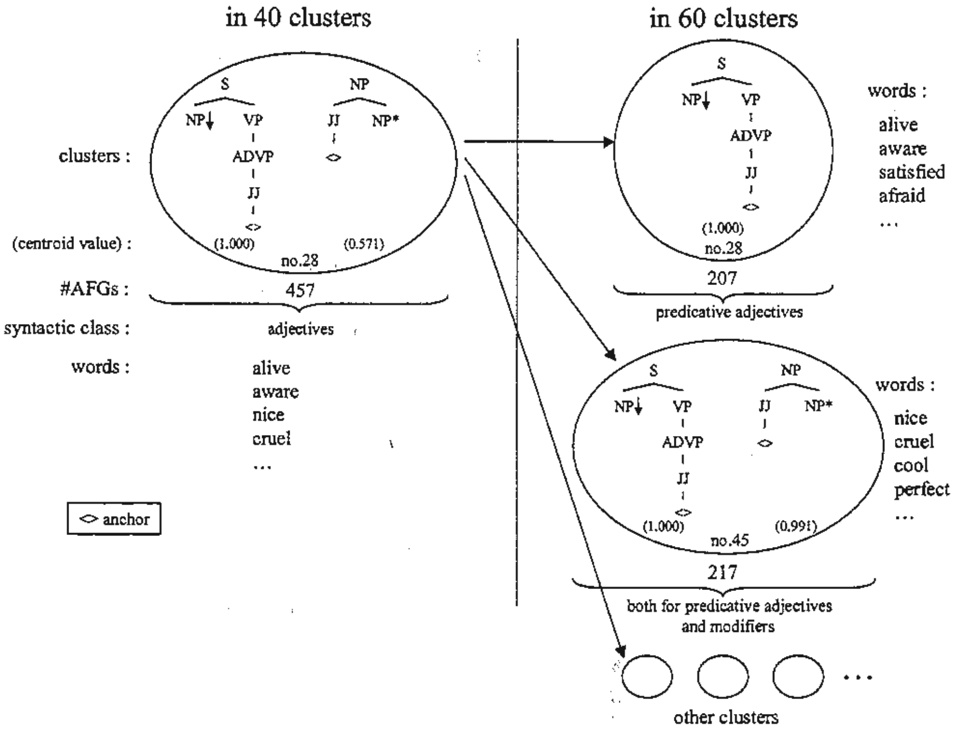


Figure 6: A divided class of syntactic roles

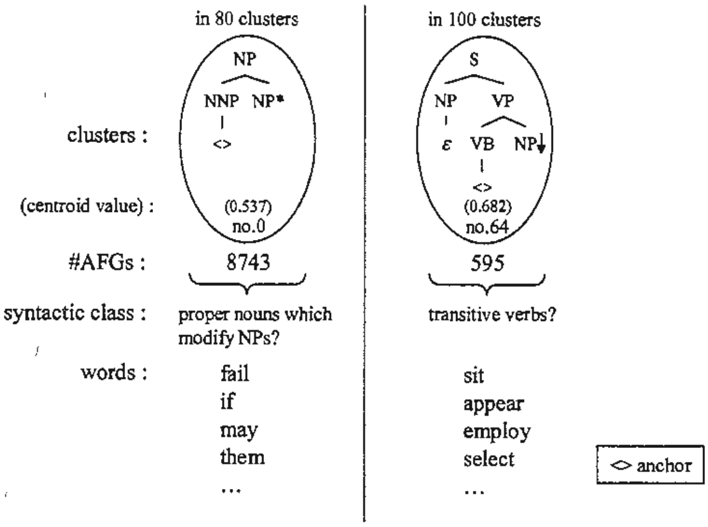


Figure 7: Ambiguous clusters

AFGs for adjectives; the cluster No. 18 was for present participles; the cluster No. 7, No. 9 and No. 12 were for numerals; the cluster No. 19 was for the words which can be both adjectives and singular nouns. In the case of 40 clusters, the clusters for more detailed syntactic classes were generated. For example, the clusters No. 26, 31, and 35 shown in Figure 5 were for adverbs that mainly modify a sentence, an adjective, a verb, respectively; the cluster No. 32 was for base form transitive verbs.

In addition, we could observe that there were some clusters made by dividing one cluster in the case of less clusters, and in both of which syntactic classes could be identified. For example (Figure 6), the cluster No. 28 in the case of 40 clusters was mainly for adjectives. About a half of AFGs in it were only for predicative adjectives and the rest of them were not. In the case of 60 clusters, the two types of AFGs were divided almost into two clusters, No. 28 and No. 45. We could see many AFGs only for predicative adjectives, such as "aware," "glad," "alive," in the cluster No. 28. In the other cluster (No. 45), we could see many AFGs for adjectives which could be both predicative adjectives and modifiers, such as "wrong," "hot," "certain." These clusters were generated by classifying the AFGs for one syntactic class in a more detailed way by increasing the number of clusters.

In the case of 80 clusters, there were some clusters whose centroid values of tree templates were no more than 0.5 (Figure 7). This meant that there was no tree template which was common to all AFGs in such clusters, and we could not identify syntactic classes for them. This would suggest that the number of clusters exceeded the one suitable for clustering the AFGs, and in consequence the AFGs were classified too finely.

However, this would not indicate that there would be less than 80 classes of syntactic roles in the grammar. When we focused on nouns, we could find too fine classification for them. In the case of 100 clusters, the number of them was no less than 37. On the other hand, clusters for verbs were few. In the case of 100 clusters, the number of them was 20, and expected clusters such as one for ditransitive verbs were not in those 20 clusters. The reason for this would be as follows.

A noun class contains many words, and AFGs for nouns in the grammar reflected this fact; the AFGs for nouns occupied about $18,427/43,030$ of all the AFGs in the grammar. On the other hand, a verb class contains not so many words and AFGs for verbs occupied $1,190/43,030$. Our clustering method treated all words equally regardless of parts-of-speech, and as a result, words in the noun class would be classified too finely, and words in the verb class, too roughly.

4. Conclusion

We proposed the method for automatically obtaining syntactic classes of words from automatically extracted LTAG grammars. We supposed that the class of the syntactic roles of a word corresponded to the set of elementary trees for the word, and attempted to obtain syntactic classes by clustering words that have similar elementary trees. The experiments showed that our method could obtain some clusters each of which represents a certain syntactic class. However, it also showed that our method would not obtain all syntactic classes properly, because it was affected by the differences in the number of words in each part-of-speech class. We should consider this result, and build an algorithm that would not mix various parts-of-speech, but would cluster groups of elementary trees for them separately. If we can obtain proper classes of syntactic roles, we will be able to apply various methods that improve and make good use of the extracted grammar. For example, we will be able to properly predict elementary trees which are not in the extracted grammar, by giving major tree templates to all words in the cluster. Such predicted elementary trees will improve the coverage of the grammar in a syntactically proper way.

References

Chen, John and K. Vijay-Shanker. 2000. Automated Extraction of TAGs from the Penn Treebank. In *Proc. of the 6th IWPT.*

Chiang, David. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proc. of the 38th ACL*, pages 456–463, October.

MacQueen, J. B. 1967. Some methods of classification and analysis of multivariate observations. In *Proc. of the Fifth Berkeley Symposium on Mathemtical Statistics and Probability.*

Marcus, Mitchell, Beatrice Santorini and Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Schabes, Yves, Anne Abeillé and Aravind K. Joshi. 1988. Parsing strategies with 'Lexicalized' grammars: Application to Tree Adjoining Grammars. In *Proc. of the 12th COLING*, pages 578–583.

The XTAG Research Group. 1995. A Lexicalized Tree Adjoining Grammar for English. Technical Report IRCS 95-03, Institute for Research in Cognitive Science, University of Pennsylvania.

Xia, Fei. 1999. Extracting Tree Adjoining Grammars from Bracketed Corpora. In *Proc. of the 5th NLPRS.*