Learning Cooccurrences by using a parser

Kazunori Matsumoto Hiroshi Sakaki Shingo Kuroiwa
KDD Kamifukuoka R&D Labs.
Saitama, Japan

ABSTRACT

This paper describes two methods for the acquisition and utilization of lexical cooccurrence relationships. Under these method, cooccurrence relationships are obtained from two kinds of inputs : example sentences and the corresponding correct syntactic structure. The first of the two methods treats a set of governors each element of which is bound to a element of sister nodes set in a syntactic structure under consideration, as a cooccurrence relationship. In the second method, a cooccurrence relationship name and affiliated attribute names are manually given in the description of augmented rewriting rules. Both methods discriminate correctness of cooccurrence by the use of the correct syntactic structure mentioned above.

Experiment is made for both methods to find if thus obtained cooccurrence relationship is useful for the correct analysis.

1. Introduction

Much attention should be paid for the role of minutely described grammar and real world knowledge in order to improve natural language analysis performance. In this respect, the authors have tried to acquire and use coocurrence data for the improvement of analysis performance. By combining a parser and an acquisition mechanism, we implemented a learning program of lexical cooccurrence data. The program has two kinds of inputs, example sentences and the corresponding correct structures. The related study of learning grammar from sentences and their semantic structure is conducted in LAS[1] (Language Learning System) by Anderson. He is of the opinion that most of grammars are derived from semantic structures. We advocate the use of syntactic structures, because information such as cooccurrence is a reflection of the real world and is easily derived from syntactic structure. Furthermore we implemented a parser to utilize the acquired lexical data.

This paper describes above mentioned two methods for acquiring lexical cooccurrences and also describes the experiment results of the methods.

The result of the experiments shows (1) a reduction of the number of alternative analysis trees (2) the increase on probability of selecting a correct analysis tree. The experiments might be influenced by the used sentences and the nature of the used grammar. However we believe that our methods proposed here is one of the promising ways to reflect real world knowledge to sentence analysis.

At first, we explain the parser we use. This parser is based on augmented CFG. And the parser produces a forest (multiple analysis trees), and selects a single structure from the forest.

In section 3, the first of the above methods is showed. The method has two features : (1) Comparing generated analysis structures with the correct structure which should be generated by a parser for a treated sentence, each sequence of the governors on sister nodes is judged into two cases, correct case or wrong one. (2) The sequence which is always judged as a wrong case through the period of acquisition, is utilized for reducing analysis trees generated by the parser.

Experiments are made to measure effects of the second feature above. The result shows : when the set of example sentences are equivalent to the set of analyzed sentences, very few ambiguous analysis trees are generated. Almost all the selections of generated trees, then, are successful. However when the set of examples are not equivalent to the set of analyzed sentences, only one third of ambiguous trees are eliminated and probability of selection decreases a little in comparison with a original (no action) case.

In section 4, several problems of the first method are described. And in the successive section, a modified implementation is showed. We explain three modifications. The first modification is to uses the information of any proper attributes on the node. This information is manually described in augmented rewriting-rules. The information consists of the names of relations and the calculation of arguments for the relations. The second modification is to raises the priority of the structure which appears the cooccurrences judged solely as correct all through the period of acquisition. The third is to collect cooccurrence data on two phases.

In section 6, we show the analysis performance of the modified version on our experiment. The results show that modified version shows better performances than the previous version, when relatively small number of acquired data is utilized. Furthermore we show another experiment which measures the appearance rate of acquired cooccurrences data in each parsed text with the measurement of an analysis performance in each text. By this measurement, we can confirm that texts having high appearance ratio are analyzed more accurately than texts having low appearance ratio.

2. Features of the utilized parser

In our method, cooccurrence data are collected with a parser. Here, we utilize a parser of a English-to-Japanese machine translation system named KATE. The analysis technique for a English sentence is based on augmented context free grammar like LINGOL. Cook-Kasami-Younger algorithm and Early algorithm are implemented with some fast parsing techniques[2] in this parser. Other features of the parser are :

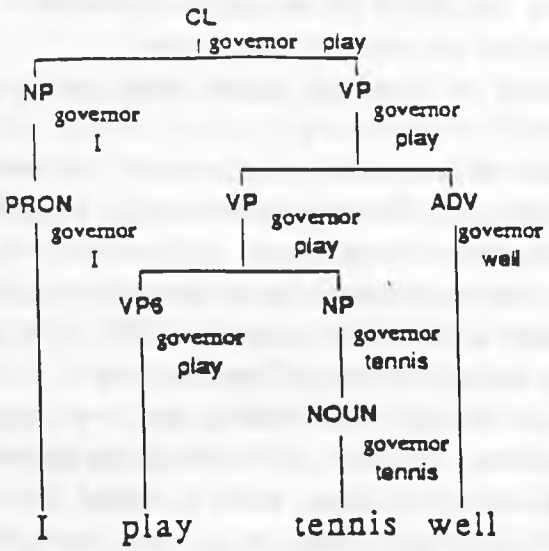
(1) Each node of syntactic trees generated by the parser has attributes information which is the meaning representation of the sub-tree governed by the node.

(2) On each node, a governor (the word which represents the phrase) is given as one of attributes.

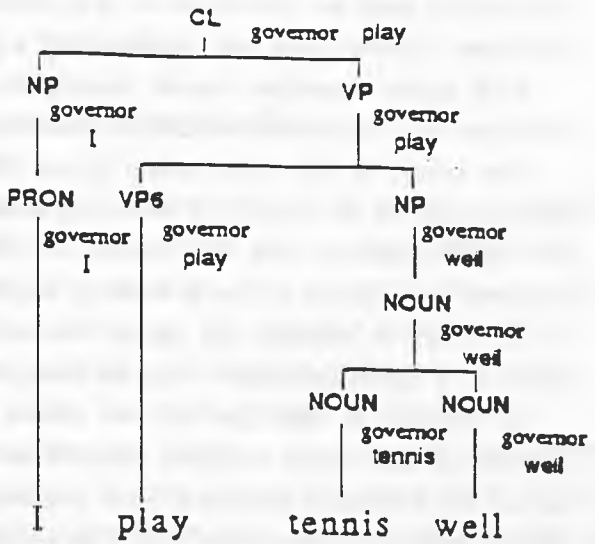
(3) We can register partial patterns of possible syntactic trees, and when a rule generates such a pattern on parsing stage, then the application of the rule is inhibited. These inhibiting patterns are used for the suppression of ambiguous trees.

Examples of generated trees and governors are showed bellow. [Fig 1, 2]

[Figure 1 An example of an analysis tree]



[Figure 2 An example of an analysis tree]



3. Acquisition and usage of relationships between governors in a simple version

Details of the first method are explained here. We call the program for this method a simple version. This version is more easily implemented than the modified version described in section 5, but lacks the accuracy in collecting cooccurrence data, We show this method for explanation purposes.

3.1 Discrimination procedure of a cooccurrence and maintenance of stored cooccurrence-data

Insufficient semantic analysis causes the generation of unproper syntactic trees, like one in Fig.2. Our program compares each generated tree with the correct tree of corresponding sentence, and classifies the sequence of governors appearing on sister nodes into two classes for each rewriting-rule. When the sequence of the governors occurs on the following two conditions, the program judges the sequence as a correct cooccurrence data, otherwise judges as a wrong cooccurrence data.

(1) the same rule which fields the remarked sentence is applied in the correct syntactic tree ;

(2) In each sub-node of the applied rule, the terminal words sequence rewritten is the same as the terminal words sequence rewritten by correct applications in the correct tree.

If we assume the tree in Fig.1 is a correct syntactic tree we obtain, from the trees in Fig.1 and Fig.2, we obtain following correct cooccurrence data and wrong cooccurrence data.

Correct cooccurrence data from Fig.1 & Fig.2

≪	I	play	≫	for	CL	→	NP VP
≪		I	≫	for	NP	→	PRON
≪	play	well	≫	for	VP	→	VP ADV
≪	play	tennis	≫	for	VP	→	VP6 NP
≪		tennis	≫	for	NP	→	NOUN

Wrong cooccurrence data from Fig.1 & Fig.2

≪	play	well	≫	for	VP	→	VP6 NP
≪		well	≫	for	NP	→	NOUN
≪	tennis	well	≫	for	NOUN	→	NOUN NOUN

In accordance with this discrimination procedure, the sequence of governors may be judged as correct cooccurrence data in one example sentence and be judged as wrong cooccurrence data in another. So the program stores the sequence of the governors into three categories. First is the set of sequences being always judged as correct cooccurrence data by the discrimination procedure. The second is the set of sequences being always judged as wrong cooccurrence data. And the last is the set of sequences being judged as correct cooccurrence data in one or more cases and judged as wrong in one or more cases. Our learning program maintains these three categories through the period when example sentences and their correct structures are inputted. In this section, we simply call the sequence of governors as cooccurrence data.

3.2 Experiment for acquiring cooccurrence data

We make an experiment for acquiring cooccurrence data with the use of the above mentioned learning program. About 3,200 example sentences are collected from a English grammar text[3] and example sentences in a dictionary. We assume each example sentence has a situation free interpretation, so if semantics analysis is successful, very few ambiguous analysis trees are generated.

We measure the number of cooccurrence data in each category at every 50 inputted pairs of sentences and correct structures. We observe that :

(1) Each number of acquired cooccurrence data increases monotonously.

(2) Finally, from 3,200 sentences, the program acquires about 10,0000 kinds of cooccurrence data belonging to the first category, about 5,000 kinds and 4,000 kinds respectively belonging to the second and the third.

However, our detailed observation finds a part of acquired cooccurrence data purposeless or mischievous. This problem is described later in section 4.

3.3 Filtering technique based on the cooccurrence data

We implemented the parser which utilizes acquired cooccurrence data. When the sequence of the governors appearing on a rule application belongs to the set of acquired cooccurrence judged as to be always wrong, the parser doesn't apply the rule. This paradigm suppresses the excessive application of rules and reduces generated trees. So the probability of selecting proper analysis tree may increase. We call this paradigm 'Filtering based on cooccurrence (judged as to be always wrong).'

And we measure the transition of following three values as the analysis performance of the parser, with the amount of increasing inputted pairs as a parameter.

(a) average of the number of generated trees per a sentence

(b) probability of generating a correct analysis tree

(c) probability of selecting a correct analysis tree

We made two experiments to measure above values.

One is measured in the condition that the set of sentences for the acquisition program is equivalent to the set of sentences analyzed by the parser. Actually we can't make the set of inputted pairs equivalent to the set of model sentences in a practical occasion. Because of the monotonous increase of acquired cooccurrence data in each category, however, we consider the result of this experiment gives a prospective view of the effect of the filtering. We observe following results. [Fig.3]

(a) With the increase of inputted pairs, average of trees decreases almost monotonously. Finally, the average becomes approximate 1.0 starting from 2.5 at the beginning. (A in Fig.3 shows the reduction of trees)

(b) Probability of generating a correct tree severely goes down, when amount of inputted pairs are few. And finally the probability, of course, becomes equal to the probability initial.

(c) Probability of selecting a correct tree also goes down, when inputted pairs are few, and after number of inputted pairs exceeds one third of the number of final inputted pairs, the probability becomes better than that of the beginning. (C in Fig.3 shows the improvement)

The second experiment is made in the condition that the set of 2,400 sentences inputted for the acquisition program is not equivalent to the set of 800 sentences parsed. Following observation is made.

(a) With the increase of inputted pairs, the average of trees decrease with a somewhat nonmonotonic. (A in Fig.4 shows the reduction of number of trees)

(b) As in the case of the previous experiment, probability of generating a correct tree goes down severely at the beginning but does not resume the initial state. (B in Fig.4)

(c) Probability of selecting a correct tree also goes down at the beginning and, what is worse, the probability finally becomes lower than that of the initial state, in spite of the assisting effect of reducing ambiguities. (C in Fig.4)

4. Problems in the simple version

This section explains eight problems of the previous simple implementation.

Problem [1] : Meaningless and purposeless data acquired.

Because the previous version discriminates and classifies all the sequences of governors appearing in all the rewriting rules, the learning program acquires purposeless cooccurrence data from the governors which represents no cooccurrence relationships. For example, in the case of the rule TEXT → CL END, which means a clause and a end-mark make a sentence, the previous program obtains the sequence of governors of CL and END. However, this sequence is useless to be utilized for parsing.

Problem [2] : Cooccurrence data judged as to be always wrong but easily revised in the future

In accordance with the increase of inputted pairs for the leaning program, the sequence of governors judged as to be always wrong so far may encounter a case where the sequence is judged as to be correct. Probability of reclassification for acquired cooccurrence data varies with the rewriting-rule related to the acquired data. For instance, in Fig.2, a sequence ≪well≫ for the rule NP → NOUN is the sequence judged as wrong. If the discrimination for this sequence doesn't contradict any discrimination caused by inputted data for the learning program, this sequence is judged as to be always wrong and used for the filtering. However, we can easily mention the example where this filtering works adversely.

Problem [3] : There exists the governor which is independent of a cooccurrence.

In the case of the rule CL → NP ADV VP, which means a noun phrase and adverb and a verb phrase make a clause, the governors of NP and VP have a cooccurrence relationship. But the governor of ADV is almost independent of this relationship.

Problem [4] : The same relationship of cooccurrence in different rewriting rules can't be dealt with.

For example, the cooccurrence relationship between NP and VP for a rule CL → NP VP and the cooccurrence relationship between NP and VP for a rule CL → NP ADV VP are identified as different relations by the previous version. However, dealing with both relationships as the same will be more advisable for the utilization of cooccurrence.

Problem [5] : There exists cooccurrence relationships which can't be represented with the sequence of governors on sister nodes.

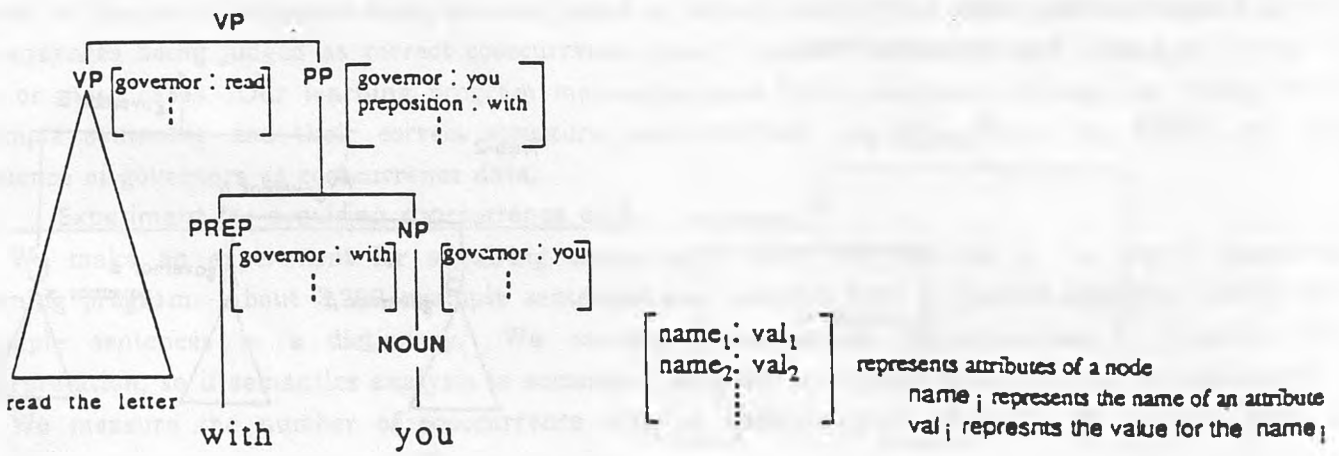
This problem is considerably affected by the grammar used. For the case of Fig.5, we explain this problem. From a rule VP → VP PP (which means a noun phrase and a prepositional phrase make a noun phrase), the sequence ≪read you≫ for the rule VP → VP PP is judged as correct, if the structure of Fig.5 is a correct structure. However, if the following sentence :

I read the letter from you.

is included in inputted pairs, a contradiction may occur. A prepositional phrase occurred in this sentence can modify a noun phrase. And if a rule VP → VP PP is applied wrongly, the sequence ≪read you≫ for the rule VP → VP PP is judged as wrong. Here, the acquired sequence becomes purposeless for to be utilized.

In this case, cooccurrence data for the rule VP → VP PP should be represented as the relation between the governor of VP, the preposition of PP, and the governor of PP.

[Figure 5 Part of a structre for "I read the letter with you"]



Problem [6] : In the previous version, information of cooccurrence data judged as to be always correct is not utilized.

Suppress of generated trees affects the selection of trees. Moreover information of correct cooccurrence data can improve the selection of a correct tree by the parser.

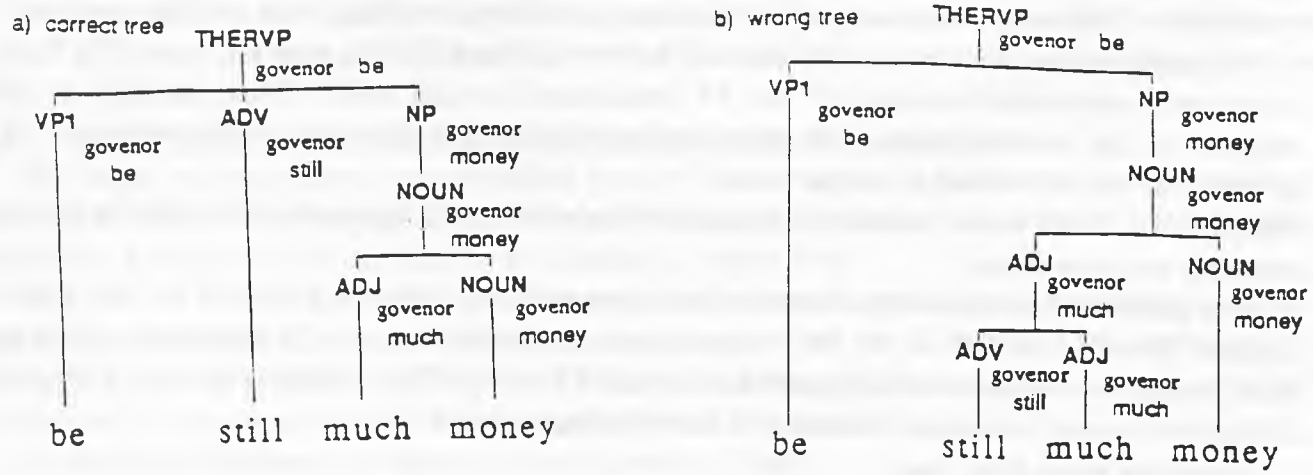
In the use of information of correct cooccurrence, however, following two problems become important.

Problem [7] : When a rule is applied at the occasion of an unproper application on lower level, the cooccurrence data which should be judged as correct may be judged as wrong.

We explain this problem with using Fig.6. In Fig.6, two analysis trees are generated. From a correct structure, the sequence ≪much money≫ for a rule NOUN → ADJ NOUN is judged as correct. And from a wrong structure, the same sequence is judged as wrong. So the data of this sequence becomes purposeless. If the application of a rule ADJ → ADV ADJ in the wrong structure fails, the sequence ≪much money≫ is only judged as correct in this sentence. We should not acquire

cooccurrence data from this wrong tree, if we consider ≪still much≫ for a rule ADJ → ADV ADJ is judged as to be always wrong.

[Figure 6 Partial trees for "There is stil much money"]

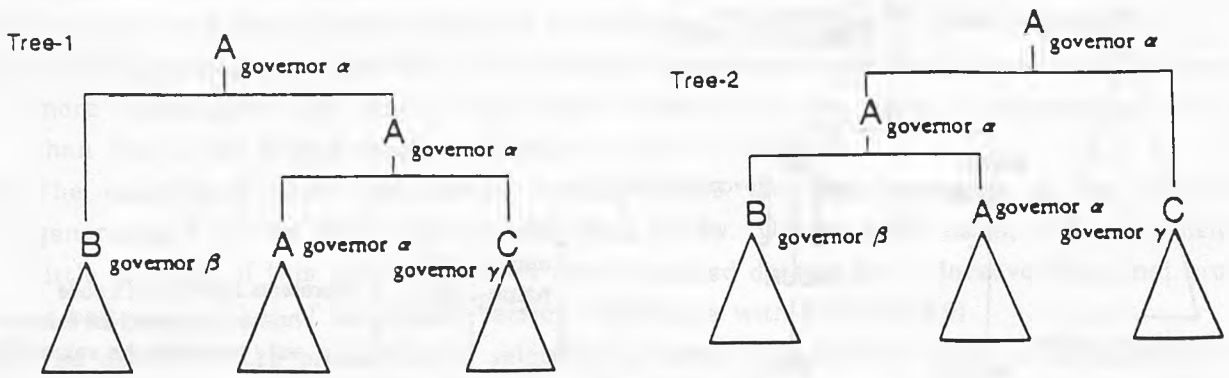


Problem [8] : Ambiguity in rule application orders causes the cooccurrence data which should be judged as correct to be judged as wrong.

We assume two rewriting rules, A → B A and → A C. If categories appear in the sequence of B A C and applications of each rule are successful, the parser generates two trees [Fig.7]. Appearance here of attributes related to the cooccurrence is assumed in Fig.7.

According to the discrimination procedure in section 3. 1, regardless of whether the tree-1 is correct or tree-2 is correct, both the cooccurrence ≪β α≫ for A → B A and the cooccurrence ≪γ α≫ for A → C A become purposeless.

[Figure 7 Two ambiguous trees]



5. Modified version of learning and usage

This section shows a modified version of the previous program. This modified version solves the problem [1]~[7] in the previous section. Only problem [8] is out of scope, but we have a basic idea to reduce this kind of ambiguity with the use of the inhibited pattern technique in section 2.

Three major modification is described bellow.

Modification [1] : Manual description of cooccurrence names and their attributes names in rewriting rules.

Rich input data is required by the system in order to determine what relations exists or what attributes are used in each relation. Therefore we consider that the kind of a cooccurrence relationship and the names of used attributes which appear in the cooccurrence should be described manually for the sake of effective learning by examples.

For this reason, we now extend the description method of the rewriting rules used in the former version. In this extended description, a cooccurrence relationship is depicted as a function of any

attributes in existing nodes and, moreover, these attributes used are depicted as functions of any attributes in all the nodes.

The program of modified version deals with cooccurrence data as bellow :

In the phase of acquisition, the program decide the name of cooccurrence and the names of used attributes in the cooccurrence, in accordance with the description of a rewriting rule.

Acquired cooccurrence data is judged similarly like in the previous method, and stored into three categories like in the simple version.

We show how the modified version solves the problems [1]~[5] mentioned in the previous section utilizing following examples.

Problem [1] : In the rules such as TEXT → CL END, which have no cooccurrence there should be no description of cooccurrence.

Problem [2] : In the rules such as NP → NOUN, which tend to be easily revised the cooccurrence should not be utilized.

Problem [3] : In the rule of CL → NP ADV VP, cooccurrence data should be described with an attribute of NP and an attribute of VP, because we consider cooccurrence relation exists between a governor of NP and a governor of VP.

Problem [4] : We should declare the same cooccurrence in both rules of CL → NP ADV VP and CL → NP VP.

Problem [5] : When we declare the cooccurrence in the rule VP → VP PP, we should choose the governor of VP, the preposition of PP, and the governor of the PP as the elements of the cooccurrence.

Modification [2] : Utilization of cooccurrence data judged as to be always correct in selection phase.

In problem [6] we pointed out the effect of using cooccurrence data judged as to be always correct. Hence, we implement next paradigm :

When a cooccurrence data judged as to be always correct occurs in a generated tree on the selection phase, the parser gives the tree a high priority for the selection purpose.

Modification [3] : Acquisition for cooccurrence data is executed in 2-passes.

To solve the problem [7], we modified the procedure of acquiring cooccurrence data. On the first pass of acquisition, the acquisition of cooccurrence is executed as in the previous version. After the end of the first pass, the modified program clear the both storages of 'cooccurrence judged as to be always correct' and 'cooccurrence judged as to be correct and wrong simultaneously.' This program executes the acquisition again from the beginning of inputted pairs with the filtering based on acquired cooccurrence.

In the case of Fig.6, if the sequence ≪much time≫ is judged as to be always wrong at the end of the first pass of acquisition, a wrong tree in Fig.6 can't be generated by the parser on the second pass of acquisition. For this reason, the sequence ≪much time≫ is not judged as wrong in this sentence.

6. Acquisition and usage of cooccurrence data in the modified version

The result treated here is the one for the modified version. We make an experiment with the same example sentences as used for the simple version, but the used grammar is slightly different. The authors believe this slight difference is negligible for the comparison with the simple version and the modified version.

6.1 Experiment of acquiring cooccurrence data by the modified version

According to the same way of treatment in the simple version, we measure the number of each stored cooccurrence data for the modified version. The result shows each stored data increases monotonously with the increase of inputted pairs. [Fig.8]

The result is similar to that of the simple version in 3.1. More 'cooccurrence data judged as to be always correct' and more 'cooccurrence data judged as to be always wrong' are obtained in the modified version than in the simple version. This may be the reason why one or more cooccurrence relationships

are obtained in a single rewriting rule. Furthermore the result shows the number of 'cooccurrence data judged as to be correct and wrong simultaneously' is about one fourth of the simple version. This phenomenon is caused by manual descriptions for cooccurrence relationships, because these description suppress the acquisition of meaningless cooccurrence data and the acquisition of data easily reclassified.

We also examine the effect of the 2-pass acquisition. We observe that about 10% 'cooccurrence data judged as correct and wrong simultaneously' on the first phase are obtained as 'the data judged as to be always correct' on the second pass of acquisition.

6.2 Experiment of using acquiring cooccurrence data with modified version

We make two experiments with modified version like in section 3.3, in order to the transition of next three values : (a) average of the number of generated trees per a sentence (b) probability of generating a correct tree (c) probability of selecting a correct tree.

The first is under the condition that the set of sentences for acquisition is equivalent to the set of sentences for analysis. The second is for the condition that the set of sentences for acquisition is not equivalent to the set of analyzed sentences. We use the same set for acquisition and the same set for analysis as in experiments of the simple version on each two experiment.

At the first experiment we observe following results [Fig.9] :

(a) With the increase of inputted pairs, the average of generated trees decreases monotonously like in the experiment for the simple version. But at the final state, the effect of reducing the number of trees is less than that of the simple version. (Compare with **A** in Fig.3 ,8)

(b) When amount of inputted data are few, adverse effect of failing to generate a correct tree in the modified version is less than that in the simple version. Furthermore the range of fluctuation in the probability through this experiment is less than that in the simple version.

(c) When amount of inputted data is few, the probability of selecting a correct tree increases, which is differ from the simple version. The probability at the final state is lower than that of the simple version. (Compare with **C** in Fig.3,8)

And we observe following results [Fig.10] at the second experiment :

(a) With increase of inputted data, the average of generated trees also decreases. This decrease is more monotonous than that of the simple version, but the effect of suppressing trees is less than that of the simple version. (Compare with **A** in Fig.4,9)

(b) The experiment under the simple version shows the sever decrease of the probability of generating a correct tree, when inputted data is few. On the other hand, this experiment shows little decrease of this probability even when inputted data is few. Moreover the final probability is better than that of the simple version. (Compare with **B** in Fig.4,9)

(c) The decline of the probability of selecting a correct tree is very slight in comparison with the simple version, when inputted data are few. The final probability by this modified version slightly exceed that by the simple version. (Compare with **C** in Fig.4,9)

6.3 Performance analysis for the ratio of acquired cooccurrence data

We define the proportion of cooccurrence data obtained through the learning by examples to the cooccurrence data appearing in a parsed text as the ratio of acquired cooccurrence data. This section describe the experiment which treats the relation between analysis performance and the ratio of acquired cooccurrence data.

We choose 2,400 sentences for acquisition and six variations of sentence sets for analysis. Here, each of six sets is not equivalent to the set for acquisition. At first, we measure the ratios of acquired cooccurrence data for each of six sets, and measure performance for each of six sets with the use of acquired cooccurrence data. By these measurement we obtain following prospective view through the experiment.

When we compare, for each of those six sets, differences between the average of the number of generated trees by the parser without cooccurrence data and that with cooccurrence data, the difference

by the higher ratio text tends to be larger than a low ratio test [Fig.11]. And a higher ratio text tends to have less adverse effect on the probability of generating a correct tree than a lower ratio text [Fig.12]. Furthermore a higher ratio text is likely to have better prospect on the probability of selecting a correct tree than a low ratio text. [Fig.13]

[Figure 11 Difference of the average of generated trees]
[Figure 12 Difference of the probability of generating a correct tree]
[Figure 13 Difference of probability of selecting a correct tree]

Figure 11

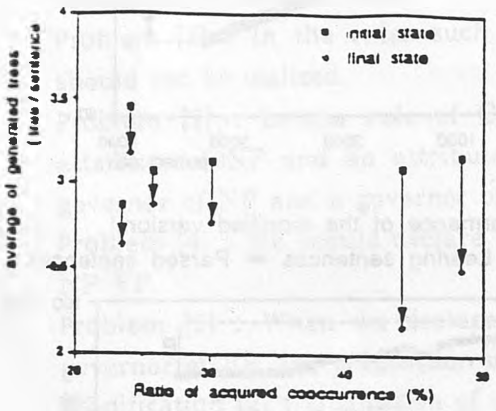


Figure 12

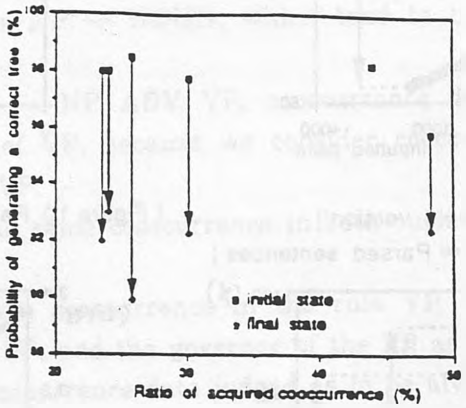
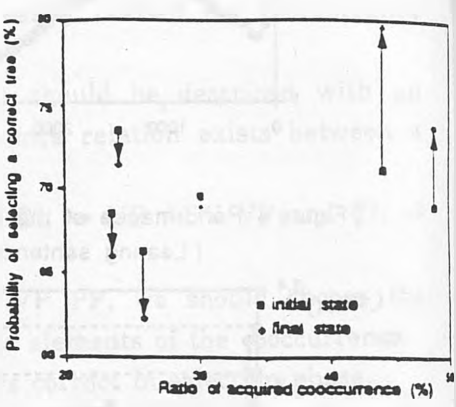


Figure 13



7. Conclusion

We observe cooccurrence data acquired by the modified version has less adverse effects on sentence analysis than by the simple version under the circumstance of relatively few acquired data. Though we consider sentences used in our experiments are basic and limited, we may conclude information of cooccurrence which human being has is very useful for acquiring cooccurrence relationships.

We conclude both of the simple version and the modified version are effective to suppress the generation of unproper tree structures by a parser and to raise the probability of selecting proper structures by a parser.

Authors believe in the modified version has more potential to learn cooccurrence by examples than the simple version.

ACKNOWLEDGEMENT

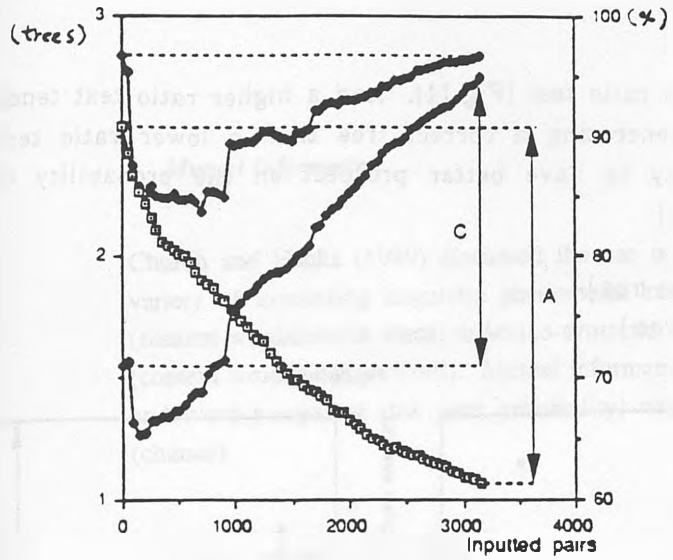
The authors wish to thank the members of AI laboratory in KDD Kamifukuoka R&D Labs, especially Tohru Asami, Kazuo Hashimoto, and Masami Suzuki for an earlier draft of this manuscript. We also thank Dr. Ono, director of KDD Kamifukuoka R&D Labs, and Dr. Urano, deputy director for giving a chance of our research and encouraging us.

REFERENCES

1. Anderson, J. : Introduction of Augmented Transition Networks, Cognitive Science, 1, pp. 125-157 (1977).
2. Sakaki, H. et.al : A Parsing method of Natural Language by Filtering Procedure, Transaction of the IECE of Japan, E69, pp, 1114-1124 (1986).

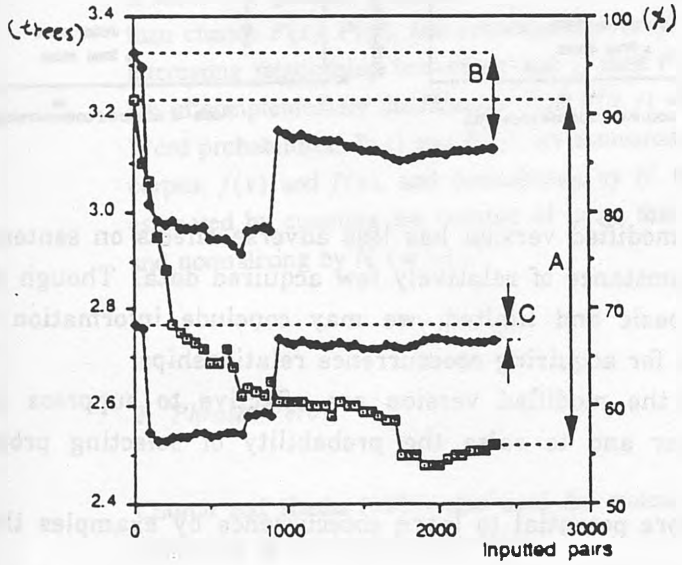
[Figure 3 Performance of the simple version]

(Learing sentences = Parsed sentences)



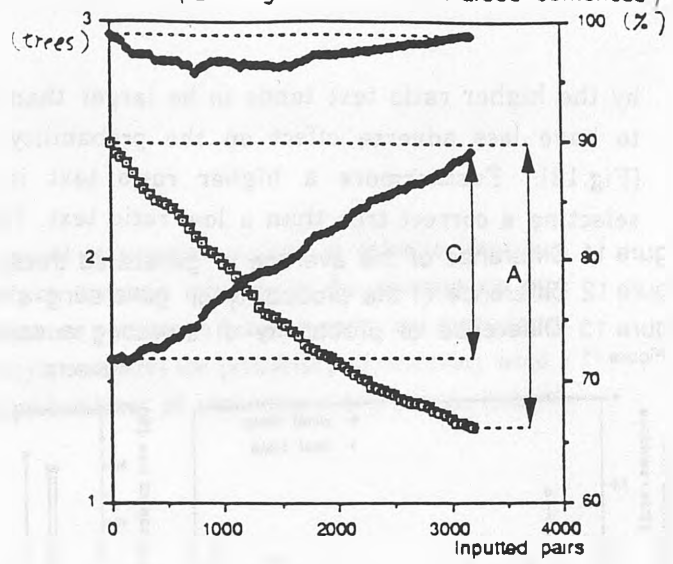
[Figure 4 Performance of the simple version]

(Learing sentences ≠ Parsed sentences)



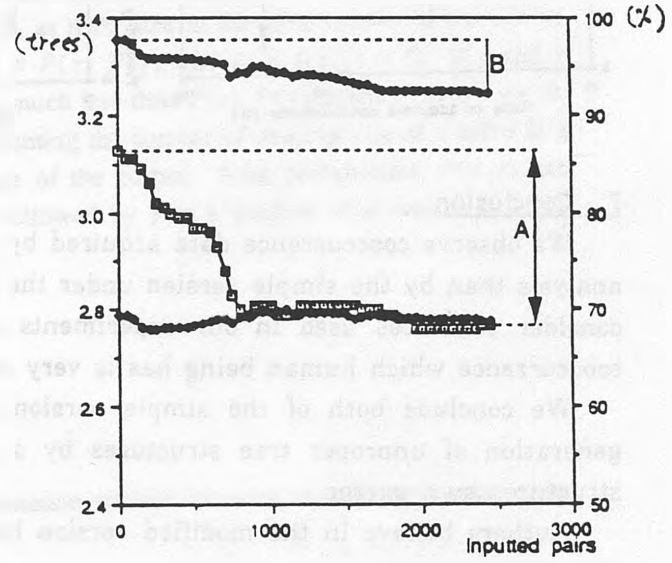
[Figure 9 Performance of the modified version]

(Learing sentences = Parsed sentences)

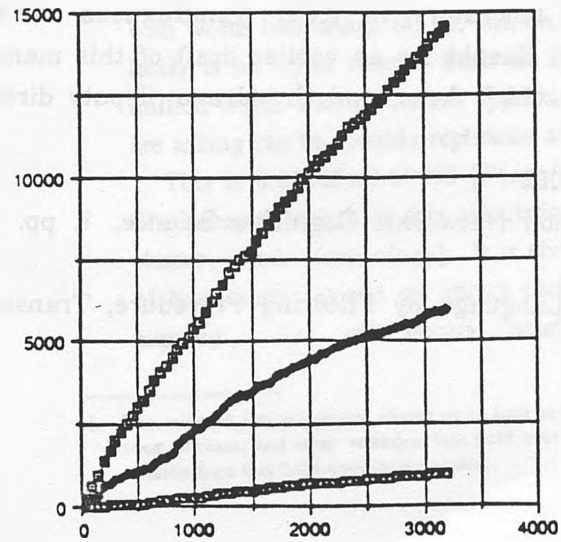


[Figure 10 Performance of the modified version]

(Learing sentences ≠ Parsed sentences)



[Figure 8 Number of the three kinds of cooccurrence data]



- cooccurrence data judged as to be always correct
- cooccurrence data judged as to be always wrong
- cooccurrence data judged as to be correct and wrong simultaneously

In Fig. 3 , 4 , 9 , 10

- Average of generated trees per a sentence
- Probability of generating a correct tree (%)
- Probability of selecting a correct tree (%)