

NAIST-IS-DT9761017

Doctor's Thesis

**Japanese Dependency Structure Analysis based
on a Lexicalized Statistical Model**

Masakazu Fujio

February 7, 2000

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

Doctor's Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
DOCTOR of ENGINEERING

Masakazu Fujio

Thesis committee: Yuji Matsumoto, Professor
Kiyohiro Shikano, Professor
Hiroyuki Seki, Professor

Japanese Dependency Structure Analysis based on a Lexicalized Statistical Model *

Masakazu Fujio

Abstract

The claim of this dissertation is that statistics of surface features, such as part-of-speech tags and head words, extracted from a large corpus of parsed sentences, along with particular algorithm, can produce accurate parses.

In the literature of a syntactic analysis, statistical approach exhibited some degrees of success, and various statistical parsing models are proposed. But it is not enough for practical natural language applications (NLP). If we want to achieve a higher rate of accuracy, it is necessary to use much information. However, the more information we use in a statistical model, the more parameters we must estimate from a corpus, and the much more corpora we need (in other words, the effect of sparse-data problem becomes more serious). In some cases, all the parameters cannot be correctly estimated because of the limit of computational resources (memories and disks).

The basic statistical model is not so much complex as other statistical parsers in the literature of a computational statistical parser. We stick to a statistical model of simple setting aiming at an easy implementation and efficiency of parsing. Instead, we address the problem of subordinate clauses and coordinate structures, which are among the major causes of difficulty in the syntactic analysis of a Japanese sentence.

In the study of subordinate clauses, we propose the decision list model that considers “modify” and “beyond” relation between subordinate clauses. We show that this model contributes to improve the precision of dependency analysis of sentences.

* Doctor's Thesis, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DT9761017, February 7, 2000.

In the study of coordinate structures, we propose the method to treat basic dependency analysis and coordinate structure identification in a uniform way, and show the model improves the precision of dependency analysis of sentences.

To achieve higher precision under the available statistical model, we also propose statistical partial parsing (the method to achieve higher precision at the cost of lower recall) and redundant parsing methods (the method to achieve higher recall at the cost of lower precision).

Keywords:

parsing, statistic, dependency analysis, decision list, subordinate clauses, coordinate structures, partial parsing

Acknowledgments

Above all, I would like to thank my supervisor Professor Yuji Matsumoto for all the ways in which he has helped me, over the years. His comments on my research and the manuscript have been of great value. And all of this research would not have been possible without his guidance and encouragement.

I am also very grateful to my second supervisor Professor Shikano, and Professor Seki of Nara Institute of Science and Technology, for their comments on my work, and detailed correction of my writing in English.

I am indebted to Associate Professor Yashuharu Den, Dr. Takehito Usturo and Dr. Takashi Miyata for constructive and fruitful discussions and advises. Their suggestions and advises helped me a lot when I lost my way on my research.

Dr. Takehito Usturo taught the importance of the perspective of the research before actually beginning macroscale experiments. The study on the subordinate clauses owes greatest debts to his work. Dr. Takashi Miyata supported me a lot, especially for the trouble on the machine.

I am grateful to my colleagues in computational linguistics laboratory at Nara Institute of Science and Technology for many helpful and valuable suggestions. I could spend grateful and precious time with these members, both officially and privately. Special thanks are due to the system administrators for providing and maintaining of comfortable computer environment. Especially, Satoru Takabayasi taught me many useful tools and techniques, such as CVS, autoconf, ndtpd and so on, which improved the environment of the programming very much. His way of thinking about the manner of the programming teaches me a lot.

And I would like to thank Akira Kitauchi, Hiroaki Noguchi and Kou Kawabe, Shinya Takamura for their encouragements and supports both officially and privately.

Finally, I wish to thank my parents, Masatomo and Mutsuko Fujio, for their continuous encouragements and supports.

Contents

1. Introduction	1
1.1 The Motivation for Parsing	1
1.2 Syntactic Ambiguity	2
1.3 Phrase Structure Analysis and Dependency Analysis	6
1.3.1 Phrase Structure Analysis	6
1.3.2 Dependency Analysis	7
1.3.3 Correspondence between CFG and Dependency Analyses	9
1.4 Rule Based Approach v.s. Statistical Approach	9
1.4.1 Rule Based Approach	9
1.4.2 Statistical Approach	10
1.5 Dependency Structure Representation	13
1.5.1 Free Word Order Language	13
1.5.2 Topicalization	14
1.5.3 Lexical Preference	15
1.5.4 Adaptability	15
1.6 Organization of this Dissertation	15
2. Related Work	18
2.1 Statistical Dependency Structure Analysis	18
2.1.1 Lexical Preference and Structural Preference	18
2.1.2 Grammar Extension Approach	19
2.1.3 Lexicalization in Various Formalisms	22
2.1.4 Direct Use of Words Associations	26
2.1.5 Decision Tree Model	27
2.1.6 Maximum Entropy Approach	28
2.2 Statistical Partial Parser	28
2.3 Coordinate Structures	29
2.4 Subordinate Clauses	31
2.4.1 Scope Embedding Preference	31
2.4.2 Applying Scope Embedding Preference to Dependency Preference	32
2.5 Topicalization	34

2.6	Ellipsis	35
3.	Basic Dependency Analysis Model	36
3.1	Constraints	36
3.2	Definition of a Segment	36
3.2.1	Composition of a Segment	36
3.2.2	Definition of Segment Features	37
3.3	Statistical Model	39
3.3.1	Training of Head Collocation Probability	41
3.3.2	Training of distance probability	44
3.3.3	The Collins' Model	44
3.4	Parsing Algorithm	46
3.4.1	Parsing Procedure	46
3.4.2	Back-Off Method for Head Collocation Probability	46
3.4.3	Back-Off Method for Distance Probability	49
3.4.4	CKY Algorithm	49
3.5	Training and Test Corpora	50
3.5.1	EDR Corpus	50
3.6	Evaluation	52
3.6.1	Evaluation Measure	53
3.6.2	Results of Segment Level Precision	53
3.6.3	Precision of Correct Dependency Structures	56
3.6.4	Evaluation for Dependency Relations	57
4.	Application of Dependency Probability	61
4.1	Partial Parsing	61
4.1.1	Algorithm	63
4.1.2	Evaluation Measure	64
4.1.3	Results	64
4.2	Redundant Parsing	66
4.2.1	Algorithm	66
4.2.2	Evaluation Measure	66
4.2.3	Results	67

5. Statistical Model for Subordinate Clauses	69
5.1 Definition of Subordinate Clauses	69
5.2 Learning Dependency Preference	70
5.2.1 Decision List Learning	70
5.2.2 Model for Dependency Preference of Subordinate Clauses .	71
5.3 Calculation of Preference Value for Subordinate Clause Dependencies	72
5.4 Experiments and Evaluation	75
5.4.1 Test Data	75
5.4.2 Evaluation Method	76
5.4.3 Results	76
5.4.4 Comparison between Related Works	77
5.5 Conclusion	81
6. Statistical Model for Coordinate Structure	83
6.1 Definition of Coordinate Structures	84
6.2 How to Learn from Coordinate Structures	84
6.3 Parsing Algorithm	87
6.3.1 Calculation of the Dependency Probability	91
6.4 Training and Test Corpora	95
6.4.1 Kyoto University Corpus	95
6.5 Experiment	97
6.5.1 Test Data	97
6.5.2 Evaluation Measure	97
6.5.3 Results	99
6.6 Conclusion	99
7. Conclusion	101
7.1 Summary	101
7.2 Future Work	102
References	103
Appendix	109

List of Figures

1	Alignment of English and Japanese text.	3
2	An example of CFG	6
3	Derivation step of HGB.	20
4	Partially-grown decision tree for part-of-speech-tagging.	21
5	Tree representation of SPATTER.	22
6	Example of a tree adjoining step.	23
7	A link grammar parse.	24
8	Parse tree of a simple sentence.	25
9	Each constituent with n children (in this case $n = 3$) contributes $n - 1$ dependencies.	27
10	Embedded structures of Japanese clauses.	32
11	An example of BaseNP and dependency analysis	45
12	CKY algorithm	49
13	Example of EDR bracketed sentence.	51
14	Extraction of a dependency relation.	52
15	Precision of correct parses within n-best parses.	56
16	Measures to select plausible dependency relations	62
17	Relationship between precision and coverage.	65
18	Relationship between recall and redundancy rate.	67
19	Changes of segment level precisions of dependencies of a whole sentence	78
20	Changes of sentence level precisions of dependencies of a whole sentence	79
21	Modifier has a coordinate structure.	86
22	Modifiee has a coordinate structure.	86
23	Both modifier and modifiee have coordinate structures.	87
24	Normal construction of a new candidate partial parse in CKY al- gorithm.	88
25	Coordinate structure construction in CKY algorithm.	89
26	Multiple coordinate structure construction in CKY algorithm.	90
27	Modifier is a coordinate structure.	92
28	Modifiee is a coordinate structure.	92

29	Both modifier and modifiee are coordinate structures.	93
30	There are coordinate structures between a modifier and a modifiee.	93

List of Tables

1	Example of CFG (context-free grammar rules)	7
2	An example of word segmentation, POS tagging, and <i>bunsetsu</i> segmentation of a Japanese sentence	8
3	Example of PCFG (context-free grammar rules)	11
4	Keys to coordinate structures	30
5	Examples of segment features.	38
6	Variations of distance features.	39
7	Features used for estimating <i>head-collocation probability</i>	43
8	Features used for estimating Distance probability.	44
9	Phrase level precision under the our model	54
10	Phrase level precision under the Collins' model.	54
11	Precision examined by dependency types.	58
12	Features of Japanese subordinate clauses	73
13	Statistics of test sentences extracted from EDR corpus	75
14	Statistics of test sentences extracted from Kyoto University corpus	98
15	Segment level precision by the evaluation measure (1).	100
16	Segment level precision by the evaluation measure (2).	100

1. Introduction

The needs for practical natural language processing (NLP) is increasingly call for in real applications such as machine translation, information retrieval, computer-aided education, and text mining.

Parsing gives us fundamental and necessary information for understanding sentences. Parsing problems is to identify the hierarchical constituent structure of a sentence. Consider the following sentence.

- 東京と 大阪に行く。

(I go to Tokyo and Osaka.)

This sentence has at least two interpretations. One is more acceptable than the other. The plausible and acceptable interpretation is “a) someone goes to both Tokyo and Osaka”. The other interpretation, which is strange, is “B) someone goes to Osaka with the man named Tokyo”.

The difference between the two interpretation can be represented by the following syntactic structures:

- 1) [[東京と] [大阪に]] 行く。]
- 2) [東京と [[大阪に] 行く。]]

The syntactic structure 1) has a left branching structure, and corresponds to the interpretation a). The syntactic structure b) has a right branching structure, and corresponds to the interpretation b).

Although, a natural language sentence takes on different meanings, depending on its context, a syntactic structure gives us primary cues for later natural language processing. A number of potential applications described above would also benefit from highly accurate parsing of sentences.

1.1 The Motivation for Parsing

In this section, we show some examples of NLP applications that would benefit from highly accurate parsing of sentences.

Lexical pattern matching is not sufficient, to achieve high precision Information Retrieval. Consider the query like “retrieve all articles where the actress A appeared on some media.” Only with lexical pattern matching, you would retrieve all text where actress A and verbs such as “appear” or “was on TV” are in the same sentence, even though they have no relation to the appearance of actress A. This query can be implemented by retrieving of all documents where actress A is the subject of verbs such as “appear”. This example illustrates that high precision Information Retrieval requires highly accurate parsing of sentences.

In Information Extraction task, an NLP system should find facts from some group of documents. The properties or attributes of the noun phrase in a document can be inferred from the neighboring words and phrases, and from the syntactic roles it occupies. In the genre of Molecular Biology, user can know the category of the substance (such as protein, DNA or sugar) and its function from its neighboring verbs in a syntactic structure.

Machine Translation is another example. In machine translation, alignment is an important problem. Alignment is the description of how an expression in one language corresponds to an expression in another language. Word to word alignment does not always succeed, when aligning SVO language such as English, and SOV language such as Japanese. Consider the following case.

Given the pair of parsed trees, you can see “go” is the head word in the English sentence, and “行く” is the head word in the corresponding Japanese sentence. Furthermore, you can see the phrase “to Tokyo and Osaka” is an object of the verb “go”, by its position in the parse tree, and the phrase “東京と大阪に” is the object of the verb “行く”, by the case marker “に”. Parse trees allow you to correctly align sentences.

1.2 Syntactic Ambiguity

A major problem in parsing is ambiguity. Ambiguity can be roughly categorized into syntactic and semantic ones. In this dissertation, we address the problem of syntactic ambiguity. In this section, we give some examples of syntactic ambiguity that appears in natural language sentences.

The first examples are not ambiguous for humans.

- 1) 彼と大阪に行く

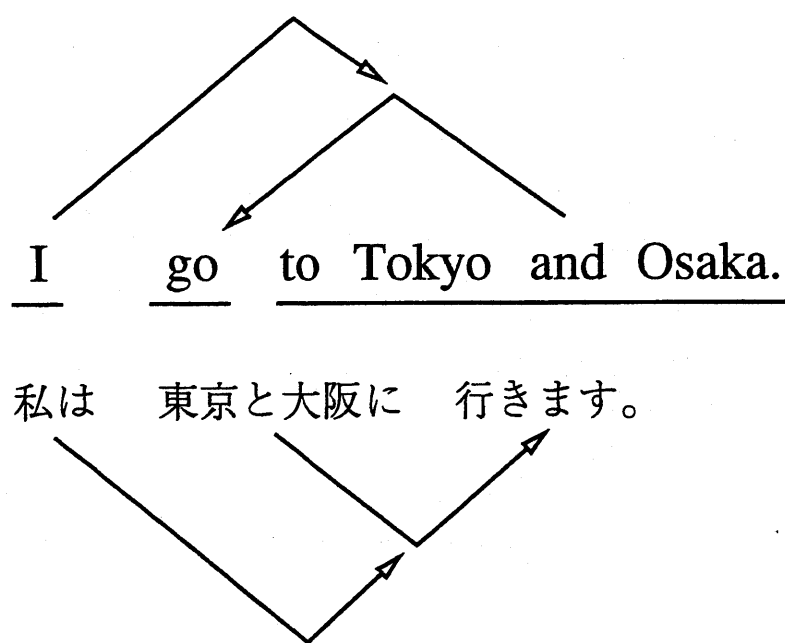


Figure 1. Alignment of English and Japanese text.

[[彼と] [[大阪に] [行く。]]
 (I go to Osaka with him.)

* [[[彼と] [大阪に]] [行く。]]
 (I go to him and Osaka??)

2) 東京と 大阪に行く

* [[東京と] [[大阪に] [行く。]]
 (I go to Osaka with Tokyo??)

[[[東京と] [大阪に]] [行く。]]
 (I go to Tokyo and Osaka.)

Both of two sentences consist of three phrases, and have two syntactically well-formed parse trees. But the analysis not marked with the sign "*" is more plausible than the other. In the first example, "彼" is human and can be an agent of an event, whereas "大阪" is the name of a place and cannot be an agent. The right branching structure (the former bracketing of the first example) corresponds to

the interpretation “I go to Osaka with him”, and all native speaker of Japanese would admit that this interpretation is correct. In the second example, both “東京” and “大阪” are the names of places, construct a coordinate structure, and become a locative of the verb “行く”. The left branching structure (the latter bracketing of the second example) corresponds to this interpretation.

A broad-coverage parser, either rule based approach or statistical approach, should correctly disambiguate syntactic structures of these examples. Both of the above examples consist of three segments (*bunsetsu*) with the following pattern.

a) $Noun_A$ +と $Noun_B$ +に 行く。

A rule based parser would examine whether $Noun_A$ can be the agent of the verb “行く”. If it is an animate, a parser may conclude that $Noun_A$ can modify “行く” with the relation type “と”¹ (which is the case of the first sentence). If $Noun_A$ is not an agent, a parser would calculate the similarity of $Noun_A$ and $Noun_B$ (because the function word “と” implies an existence of coordinate structure), and at the same time, would examine these two nouns can be a locative of the verb “行く” (which is the case of the second sentence).

In a statistical approach, the statistical model assigns probability or some sort of measure to each syntactic structure, thereby ranking competing structures in the order of plausibility. Usually, those statistics are learned from a large scale parsed corpus. For instance, in statistical dependency analysis, if you assume the independency of dependency relations, a statistical parser calculates the product of probabilities of dependency relations between $Noun_A$ and a verb “行く”, and between $Noun_B$ and “行く” (which corresponds to the right branching structure of example 1)), and the product of probabilities of dependency relations between $Noun_A$ and $Noun_B$, and between $Noun_B$ and a verb “行く” (which corresponds to the left branching structure of example 1)). After calculating the probability of each dependency structure, the analyzer selects the one that has the highest probability. It is desirable that a noun or verb’s modification tendency which is illustrated in the previous paragraph, can be learned from a corpus automatically, and that is one of the reasons for taking a statistical approach.

¹ In this dissertation, we call a sequence of function words in a segment (*bunsetsu*) as *relation type*. A relation type is crucial for disambiguation of syntactic structure of a Japanese sentence.

The third example is a more problematic one.

3) 彼と私の部屋で話した.

[[[[彼と] [私の]] [部屋で]] [話した.]]

(I talked with him in our room.)

[[彼と] [[[私の] [部屋で]] [話した.]]]

(I talked with him in my room.)

Both of the above analysis look equally plausible. A segment “彼と” and a segment “私の” can construct a coordinate structure and both of these phrases modify the segment “部屋で” (which corresponds to the former interpretation of the example 3)). On the other hand, “彼と” can modify the last segment “話した.” (which corresponds to the latter interpretation of the example 3)). By contrast with the examples 1), 2), it is not easy to disambiguate the syntactic structure without an entire dialogue or other contextual information.

For this case, a rule based parser would conclude that both syntactic structures are correct, and a statistical parser would assign equal probability to these two syntactic structures.

Consider the last example.

4) あなたしか私は分らない.

[[あなたしか] [[私は] [分らない.]]]

– I can understand only you.

– Only you can understand me.

To decide syntactic structure of the sentence 4) is easier than the case of sentence 3). But unfortunately, the parse tree would not give any information on who understands whom. For a syntactic structure for the sentence 4), both two interpretation, “I can understand only you” and “Only you can understand me”, are possible. This is rather the problem of semantic ambiguation. The goal of our dissertation is to disambiguate syntactic structure, not semantic interpretation.

1.3 Phrase Structure Analysis and Dependency Analysis

In this section, we explain the difference and correspondence between phrase structure analysis and dependency analysis. Figure 2 gives the example of the phrase structure representation and the dependency structure representation for the sentence “彼と 東京に行く。(with him, to Tokyo, go)”. Basically, phrase structure analysis gives a hierarchical constituent representation like the left tree in Figure 2. And each internal node of the tree are labeled with grammatical terms. On the other hand, dependency analysis gives word to word modification relations directory, and has no internal nodes.

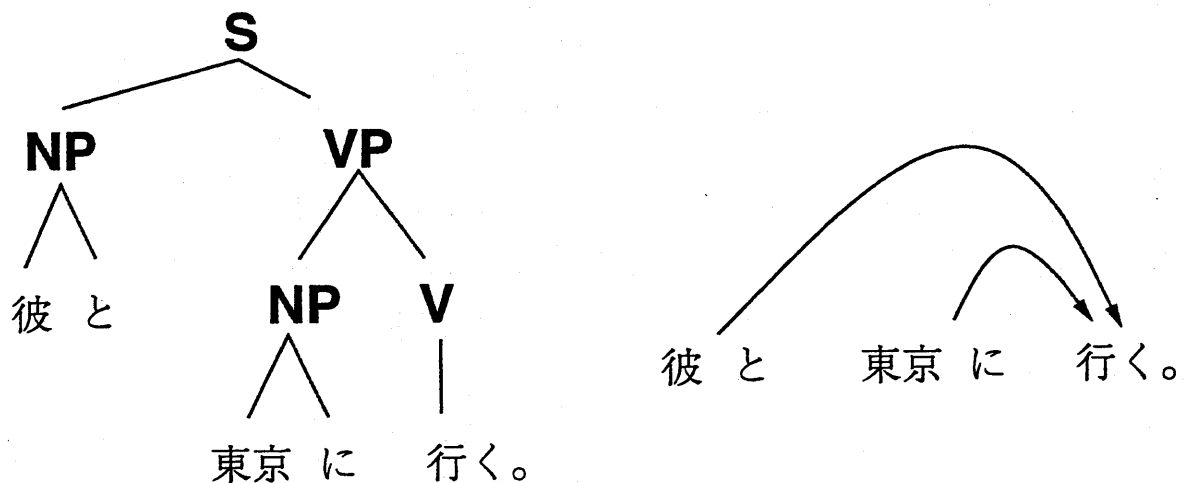


Figure 2. An example of CFG

1.3.1 Phrase Structure Analysis

Context-free grammars (CFG) are widely used as a model of natural language syntactic analysis, and various effective parsing algorithm are studied. In general, CFG for natural language parsing consist of CFG rules.

The left-hand-side of each CFG rule is a non-terminal, such as np (noun phrase), vp (verb phrase), pp (prepositional phrase), and so on. The following is an example of CFG for parsing Japanese sentences.

In this framework, disambiguation of syntactic structure becomes a problem of selecting a CFG rule and applying the left hand side of the rule to the right

s	→	np vp	noun	→	部屋 (room)
s	→	vp	verb	→	話す (talk)
np	→	noun	noun	→	大阪 (Osaka)
np	→	noun に	verb	→	行く (go)
np	→	noun と	verb	→	見る (look)
vp	→	verb	noun	→	彼 (he)
vp	→	np vp	noun	→	東京 (Tokyo)

Table 1. Example of CFG (context-free grammar rules)

hand side of other rules, and finally deriving the input sentence. For instance, for the sentence “彼と大阪に行く. ”, we can apply the following CFG rules.

1. s → np vp
2. np → noun と
3. noun → 彼
4. vp → np vp
5. np → noun に
6. noun → 大阪
7. vp → verb
8. verb → 行く.

In CFG, various parsing algorithm to obtain parse trees (in other words, to apply CFG rules in some order) are known, such as Chart parsing, CKY, Early algorithm [36] and so on.

1.3.2 Dependency Analysis

Next, we overview dependency analysis framework. Dependency analysis is widely used for Japanese syntactic analysis. In general, the following three steps are performed in a standard Japanese dependency analyzer.

1. Segmenting a sentence into a sequence of segments called *bunsetsus*.

Table 2. An example of word segmentation, POS tagging, and *bunsetsu* segmentation of a Japanese sentence

Word Segmentation	彼	と	東京	に	でかけよう
POS Tagging	pronoun	case- particle	propernoun	case- particle	verb (volitional)
Bunsetsu Segmentation (Chunking)	彼-と		東京-に		でかけよう
English Translation	<i>with he</i>		<i>to Tokyo</i>		<i>let's go out</i> (<i>Let's go to Tokyo with him.</i>)

2. Preparing a modification matrix, each value of which represents how one segment (*bunsetsu*) is likely to modify another.
3. Finding optimal dependency relations in a sentence (usually by a dynamic programming technique).

Since words in a Japanese sentence are not segmented by explicit delimiters, input sentences are first word segmented, part-of-speech tagged, and then chunked into a sequence of segments called *bunsetsu*. Each segment (*bunsetsu*) generally consists of a sequence of content words followed by function words.

Dependency analysis produces a directed tree structure that clarifies modification relations between words, and this relation is irrelevant to the word order, unlike the syntactic tree representation based on PSG. Table 2 gives examples of word segmentation, part-of-speech tagging, and *bunsetsu* segmentation (chunking) of a Japanese sentence, where the verb is tagged with their parts-of-speech as well as conjugation form.

In the step of deciding optimal dependency relations in a sentence, most practical dependency analyzers for the Japanese language usually assume the following two constraints:

1. Every segment (*bunsetsu*) except for the last one modifies exactly one segment (*bunsetsu*) to the right in the sentence.
2. No modification crosses to other modifications in a sentence.

Under these constraints, we can use the dynamic programming technique to find the most likely (in our model, the one with the highest probability) combination of dependency relations.

1.3.3 Correspondence between CFG and Dependency Analyses

If you wish to keep broad-coverage in PSG, you may need a grammar with a very large number of rules. But if you go to the opposite extreme, you would only need the following one CFG rule.

phrase \rightarrow phrase *phrase

The sign '*' indicate the head phrase among the right hand side phrases of the rule. The above rule applies to the head final language such as Japanese. This is what dependency analysis does. The last one extension needed to the above CFG rule is equality of head daughter's lexical information and that of a parent. A tree structure and a dependency structure have one to one correspondence, if you ignore the internal-nodes in tree structure.

1.4 Rule Based Approach v.s. Statistical Approach

In order to construct a broad coverage parser of Japanese sentences, rule based approach ([21, 26]), and statistical approach have been studied([14, 17]).

1.4.1 Rule Based Approach

When constructing a broad-coverage rule based parser, the difficulty is that there are lots of exceptions which may easily be overlooked by the rule writers. For instance, rule writers may assume that there is no dependency relation between a noun phrase with case marker “を” and another noun phrase with case marker “に”. But this is not always the case. The following examples illustrate this.

1. [[為替相場の終値を] 基準に] 決める.
the exchange rate/ closing price/ on the basis/ decide
(It is decided on the basis of the exchange rate of the day.)
2. ウイルスが [アジアを 中心に] 流行する.
the virus/ around Asia/ spread
(The virus spread around Asia.)

In the former case, “終値を” modifies not “決める”, but “基準に”, and in the later case, “アジアを” modifies not “流行する”, but “中心に”. In both cases, noun phrases with case marker “を” modify, not the verbs, but the noun phrases with case marker “に”.

In the rule based approach, rules must be developed by a grammarian or language expert based on their linguistic intuition. Changing rules to account for new cases may cause other errors for previously examined cases. The most important point is that no systematic way has been found to reproduce the set of rules from scratch.

For the reasons mentioned above, constructing and maintaining a rule based system is a laborious task.

1.4.2 Statistical Approach

To avoid the construction and maintenance problem of rule based approaches, statistical approach is another choice. Recently, a large amount of syntactically annotated corpus becomes available. There is a possibility to extract useful linguistic knowledge and to construct a broad-coverage syntactic parser automatically from these corpora. Various statistical or machine learning approaches are proposed.

Basically, statistical parser assigns a probability to each analysis. Thereby ranking competing parses in the order of plausibility. Early approaches of statistical parsing are based on CFG framework in which probabilities are assigned to the grammar rules conditioned on the left-hand-side non-terminal.

In general, the parameters of PCFG (probabilistic CFG), are the probabilities $P_A(A \rightarrow BC)$ and $P_A(A \rightarrow w)$, where A, B and C are non-terminals, and w is a terminal symbol. $P_A(A \rightarrow BC)$ means the probability that the CFG rule

$A \rightarrow BC$ is applied when non-terminal A appears in syntactic analysis. In PCFG, each grammar rule has a probability, and the probabilities for all the rules that share the same non-terminal sum up to 1. The task of grammarian who uses PCFG is to estimate these probabilities.

The following is a PCFG version of the CFG in Table 1. For instance, the

s	\rightarrow	np vp	0.8	noun	\rightarrow	部屋 (room)	0.1
s	\rightarrow	vp	0.2	verb	\rightarrow	話す (talk)	0.1
np	\rightarrow	noun	0.2	noun	\rightarrow	大阪 (Osaka)	0.1
np	\rightarrow	noun に (to)	0.4	verb	\rightarrow	行く (go)	0.1
np	\rightarrow	noun と (with)	0.4	verb	\rightarrow	見る (look)	0.1
vp	\rightarrow	verb	0.3	noun	\rightarrow	彼 (he)	0.1
vp	\rightarrow	np verb	0.7	noun	\rightarrow	東京 (Tokyo)	0.1

Table 3. Example of PCFG (context-free grammar rules)

probability of the parse tree constructed from the following derivation steps can be calculated by the product of the probabilities of each CFG rules appearing in the derived parse tree.

1	s	\rightarrow	np vp	0.8
2	np	\rightarrow	noun と	0.4
3	noun	\rightarrow	彼	0.1
4	vp	\rightarrow	np verb	0.7
5	np	\rightarrow	noun に	0.4
6	noun	\rightarrow	大阪	0.1
7	vp	\rightarrow	verb	0.3
8	verb	\rightarrow	行く.	0.1

One merit of PCFG to CFG is the possibility of automatic grammar induction. If you learn a grammar of CFG, you need both positive evidences (acceptable sentences) and negative evidences (unacceptable sentences) as the training data. On the other hand on PCFG, you can learn the grammar only from positive evidences. In this case, the probability of unlikely syntactic structures (in

other words, the probabilities of rarely used CFG rules) become lower by repeatedly training the PCFG model. Inside-Outside algorithm [42] can be used for the grammar re-estimation or grammar inference, in case the grammars are in Chomsky Normal Form (CNF). Further more, PCFG can construct not only the most likely parse tree, but also candidate parse trees in the order of plausibility.

But the simple framework of PCFG have the following problems:

- PCFGs cannot reflect lexical preference to the culminating probability of a parse tree.
- PCFGs simply multiply the probability of each rule derivation.

Thus, the resulting probability does not always correctly reflect the plausibility of each parse tree.

To take more contextual information into account, word collocation is applied to syntactic formalization, such as lexicalized PCFG [2, 19, 49, 32], Lexicalized Tree Adjoining Grammar [41], and Lexicalized Link Grammar [29]. The length of phrases or the distance between head words are also considered in several models [19, 30].

As an extreme of lexicalization, there are parsing methods that do not use any grammar rules. Collins [6] proposes a statistical parser based on the probabilities of dependencies between head words in parse trees. In Japanese language, we proposes a statistical model based on the collocation probabilities of segment features [14], which is described in detail in Chapter 3. We uses two types of features, one is the features defined by each segment and the other is the features which reflect distance property between modifier segment and modify segment. The collocation probabilities of these features are estimated from a corpus. Ehara and Uchimoto [10, 50] also use same kind of features to estimate a probability of each dependency relation. The difference to our model is, they use the maximum entropy method for the estimation of the probability. In English, Ratnaparkhi [39] uses the maximum entropy model for the statistical parser. Another machine learning technique used for the statistical parsing is decision tree method. Haruno [17] uses decision tree to estimate the probabilities of dependency relations based on the same kind of surface features used in Our model.

Magerman [31] proposes a statistical parser based on a decision tree model, in which the probabilities are conditioned on the derivation history of the parse trees. He compares the decision tree model with the n-gram model, and claims that the amount of parameters in the resulting model remains relatively constant, depending mostly on the number of training examples.

Charniak [5] proposes a generative statistical model and compared it with Collins', and Magerman's models and shows what aspects of these systems affect their relative performance.

In the literature of statistical syntactic analysis, about 87% of precision in terms of segment is reported, in both English and Japanese sentences.

Another merit of statistical parsing is that a probability assigned to each parse tree can be used in subsequent NLP systems. In section 4, we propose some of applications taking advantage of this fact.

As the basic framework, we adopt dependency structure analysis, which is a lexicalized formalism and needs no grammar rules.

1.5 Dependency Structure Representation

Instead of constructing a hierarchical constituent representation commonly used in phrase structure grammar (PSG), we aim to find out phrase to phrase dependency relations in a sentence. In the rest of the dissertation, we call a directed tree structure of dependency relations in a sentence as "a dependency structure", and construction of a dependency structure, as "dependency analysis".

Dependency analysis is widely used for Japanese syntactic analysis ([14, 17, 21, 26]).

The reason to use dependency structure representation is its advantages in handling some aspects of a Japanese sentence, adaptability to lexical preferences and portability to other grammar frameworks.

1.5.1 Free Word Order Language

It is often said that Japanese word order is free because the syntactic role of a given constituent in a sentence is often uniquely determined by its post positional markers. Consider the following sentence:

- (1) 難波で友だちと映画を見る。
 Namba-de tomodachi-to eiga-wo miru.
 Namba friend movie see.
 I see a movie with a friend of mine at Namba.
- (1a) 難波で友だちと映画を見る。
 (1b) 難波で映画を友だちと見る。
 (1c) 友だちと難波で映画を見る。
 (1d) 友だちと映画を難波で見る。
 (1e) 映画を友だちと難波で見る。
 (1f) 映画を難波で友だちと見る。

All of the six possible sequences of postpositional phrases constitute correct Japanese sentences. A phrase structure grammar (PSG) produces a tree structure which consists of all morphemes in a sentence and non-terminal symbols (S, NP, V, and so on). To accept these variations, PSG must prepare numerous phrase structure rules corresponding to each tree structure even if they are derived from a small set of basic rules in a systematic way.

On the other hand, dependency analysis produces a connected graph structure that clarifies which word modifies or depends on which, and the above variants of post positional phrases' order have the same dependency structure.

1.5.2 Topicalization

A certain Japanese phrase seems to modify more than one word. This happens when ellipsis or topicalization occurs. For example:

- a) 私はごはんを食べてから映画を見にいった。(topicalization)

I / meal / ate / a movie/ see / went

(After I ate a meal, I went to see a movie.)

- b) ジョンができればするでしょう。(ellipsis)

John / If possible / will do

(If possible, John will probably do it.)

In the first sentence, the phrase “私は” is a topic phrase. In the second sentence, the phrase “ジョンが” is omitted between “できれば” and “するでしょう”. One way to handle these cases in dependency analysis is to allow modifiers to modify more than one segment.

For these cases, we define that topics modify the last predicate.

1.5.3 Lexical Preference

The importance of lexical information is getting recognized, and a number of parsing frameworks incorporate lexical preference into recently proposed models (Lexicalized PCFG, Lexicalized Tree Adjoining Grammar, and Lexicalized Link Grammar).

Because dependency analysis find out a word to word relation directory, it is easy to reflect lexical preference to the whole dependency structure.

1.5.4 Adaptability

The result of dependency analysis specifies simply the relationship between the words that comprise the sentence. While this does not give an in-depth interpretation of the sentence, it is intuitively easy to understand and is amenable to many other more sophisticated grammar formalisms.

For instance, as a well-defined grammar formalism, HPSG (Head-driven Phrase Structure Grammar) [37] proposes highly lexicalized formalisms. HPSG describes necessary information for parsing in the lexicon as much as possible, and grammar rules are restricted into a six schemata. It is easy to combine with these constraint based grammar and statistical parser.

1.6 Organization of this Dissertation

The dissertation is organized in the following way. In Chapter 1, we have just addressed the problems of syntactic disambiguation in Japanese sentences and present the scope of research. For Japanese language analysis, dependency analysis is widely used as a parsing technique for some reasons. In this Chapter, we have compared several frameworks such as Phrase Structure Grammar v.s Dependency Structure Representation, and Rule Based Approach v.s. Statistical

Approach.

In Chapter 2, we overview the statistical approach for syntactic disambiguation, and statistical partial parsing method. Then we overview the work related to the structure of Japanese subordinate clauses and coordinate structures and some other features of Japanese sentences.

In Chapter 3, we propose our statistical dependency analysis model. The basic statistical model is not so complex compared with other related statistical parsers, aiming at an easy implementation and efficiency. This helps to decrease the effect of sparse data problem. We define the dependency structure as a set of dependency relations (pairs of modifier segment and modifiee segment, assume that each segment in a sentence except the last one modifies exactly one of its subsequent segments in the sentence and no two modifications may cross each other. The statistical model assign a probability to each set of dependency relations, ranking competing sets in the order of plausibility. The probability of each set is calculated as the product of the probabilities of dependency relations in a parse tree.

In Chapter 4, we propose statistical partial parsing and redundant parsing methods. Partial parsing is a method to achieve higher precision at the cost of lower recall. In other words, the partial parser outputs only plausible list of dependency relations based on the some particular measures. Redundant parsing is a method to achieve higher recall at the cost of lower precision. in other words, the redundant parser allows to output ambiguous dependency structure as far as their plausibility is high based on some predefined criterion. These methods are useful for practical Natural Language Processing applications that need more accurate syntactic analysis than the conventional syntactic parsers perform. The methods can be used for any statistical models that assume the independence of the probability of each dependency relations.

In Chapter 5, we pay attention to the scope embedding preference of subordinate clauses in a Japanese sentence, which are manually analyzed by some linguists in the literature. The scope embedding preference of subordinate clauses can be interpreted as the dependency preference between two subordinate clauses. We learn the dependency preference from syntactically annotated corpora automatically, and show that the preference contributes to the disambiguation of a

syntactic structure of the whole sentence.

In Chapter 6, we propose a method to deal with the coordinate structure analysis and the dependency analysis in a uniform way. The final probability of dependency structure reflects the form of the coordinate structures from two reasons. The first reason is that the existence of coordinate structures changes the distance features of dependency pairs. The second reason is that a coordinate structure has multiple heads, and the probability of dependency relation is calculated from the multiple probabilities defined by these multiple heads.

Finally, we summarize the dissertation and describe the future direction in Chapter 7.

2. Related Work

2.1 Statistical Dependency Structure Analysis

2.1.1 Lexical Preference and Structural Preference

One of the major ambiguity of syntactic analysis of English sentences is the PP attachment ambiguity. In general, the following construction of a sentence has an ambiguity whether (prep np_2) modifies the verb or np_1

verb np_1 (prep np_2)

Consider the following examples, with the preposition “with”:

- (a) These examples consist of two sentences with the prepositional phrase.
- (b) I go to Tokyo with him.

In the example (a), the prepositional phrase “with the prepositional phrase” attaches the noun phrase “two sentences”, while in (b), the prepositional phrase “with him” attaches the verb phrase “go to”.

Hindle and Rooth [18] use a statistical measure of lexical association to solve the structural ambiguity of prepositional phrase attachment. They propose to use co-occurrence of verbs and nouns with prepositions in a large body of text as an indicator of lexical preference.

Li [30] proposes statistical priority measures based on two psycholinguistic principles in English:

- Right Association — a constituent tends to attach to another constituent immediately to its right [24]
- Lexical Preference Rule — a case frame and its constituent tend to preserve coherent relations [13]

As a statistical model of Lexical Preference Rule, he introduces the following conditional probability:

$$P(n \mid v, s)$$

where v is a verb, s is a case frame of the verb, and n is one of the arguments of the verb in the case frame. He assumes the independency of arguments case frames

and defines “lexical preference” as the geometric mean of these probabilities. To incorporate Right Association Principle into a statistical model, Li uses a distance measure in each CFG rule. For instance, for the CFG rule $L \rightarrow H, M$, the distance is defined by the length of words dominated by the non-terminal H . This definition reflects the distance property of the dependency relation in English. The priority of each CFG rule is formulated as a function of the CFG rule and the distance of H (d), such as $S(d, (L \rightarrow H, M))$. Li combines these two priority measures by using the back-off method.

Shirai.K [43] presents a probabilistic language model which integrates lexical preference and structural preference in a consistent way. Lexical preference is defined based on a case frame model, and structural preference is based on a distance measure.

2.1.2 Grammar Extension Approach

It is often said that the basic framework of PCFG is not enough for disambiguating syntactic structure of an input sentence, because the probability of each PCFG rule does not reflect a syntactic/lexical context, and hence cannot capture non-local lexical information which is crucial for disambiguation of syntactic structures.

To take more contextual information into account, several studies extended PCFG in some way.

Magerman and Marcus [32] describe *Pearl* system, a probabilistic chart parser. While simple PCFG assumes the independence of the probability of CFG rules, their model assumes the dependence of CFG rules on the part-of-speech trigram centered at the beginning of the rule, and the dependence on the parent CFG rule.

Black et al. [2] propose the history-based grammar (HBG), that takes into account the derivation history of a parse tree. Consider the following left-to-right derivation:

$$S \rightarrow ASB \mid AB$$

$$A \rightarrow a$$

$$B \rightarrow b$$

$$S \xrightarrow{r_1} ASB \xrightarrow{r_2} aSB \xrightarrow{r_3} aABB \xrightarrow{r_4} aaBB \xrightarrow{r_5} aabB \xrightarrow{r_6} aabb \quad (1)$$

In each derivation step, they define that the partially parsed tree up to that

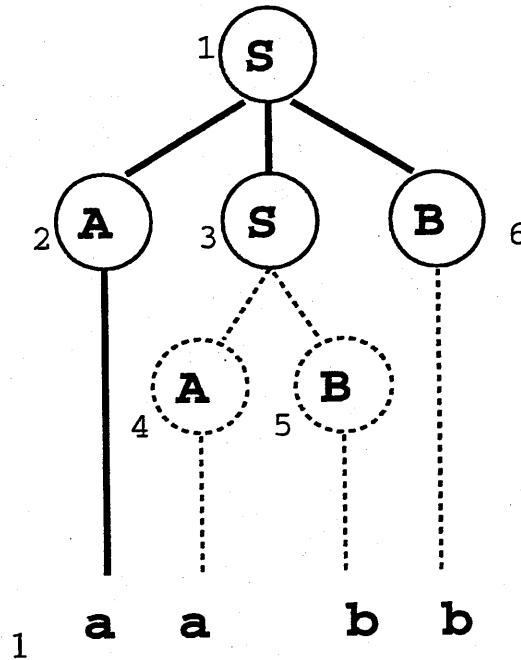


Figure 3. Derivation step of HBG.

point is a history. For instance, the history when applying r_3 is a partial parsed tree aSB . The difficulty in applying probabilistic models to natural language processing is to decide what features of the sentence are relevant to the model. To tease out the relevant features of a parse tree that will contribute to syntactic disambiguation of a sentence, they used decision tree model. However for the limitation of computational resources, they only use the path from the current node to the root in the derivation tree.

Magerman [31] considers that a parsing of a sentence is a sequence of decisions of disambiguation, such as part-of-speech tags, hierarchical constituent structures, non-terminal labels (np, vp etc.). He uses the decision tree learning technique to learn criteria for disambiguation of those linguistic constituents, and propose the statistical parser SPATTER.

A decision tree is a decision making device which assigns a probability to each of the possible choices based on the context of the decision: $P(f | h)$, where f

is an element of the *future* (the set of choices) and *h* is a *history* (the context of the decision). The probability $P(f | h)$ is determined by asking a sequence of questions $q_1 q_2 \dots q_n$ about the context, where the i -th question is uniquely determined by the answer to the $i - 1$ previous questions. Figure 4 gives an example of a decision tree.

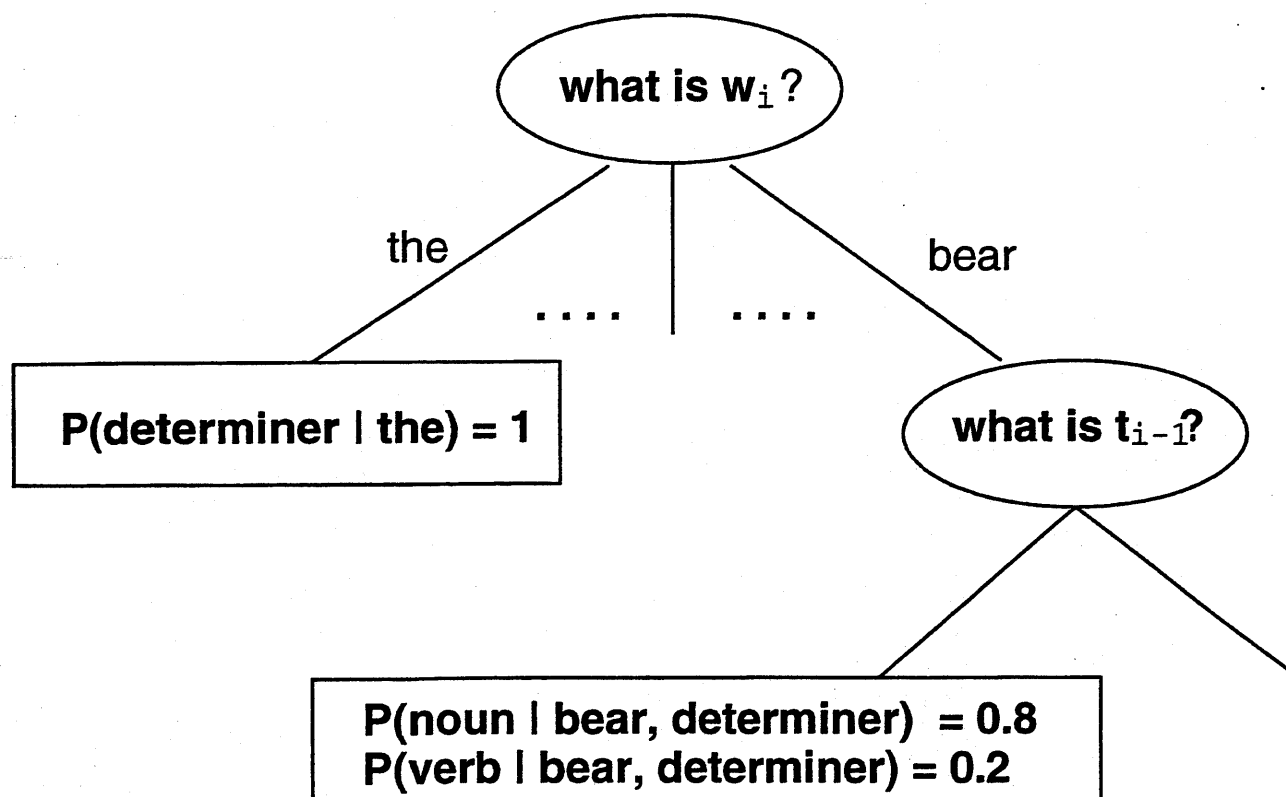


Figure 4. Partially-grown decision tree for part-of-speech-tagging.

The context of SPATTER is constructed from four elementary components: words, tags, label, and extensions. The tags are part-of-speech tags, the labels mean any non-terminals. An extension is shown in Figure 5. The name annotated to an arrow is an extension. A parsing proceeds in a bottom-up, left-to-right fashion based on the context determined by the decision tree model.

Stolcke [47] extends Earley's parser for stochastic context-free grammars, and uses the probabilities of successive prefixes in derivation of Earley's parser, to capture the lexical context.

Briscoe and Carroll [4] incorporate the probability of a lexical and structural

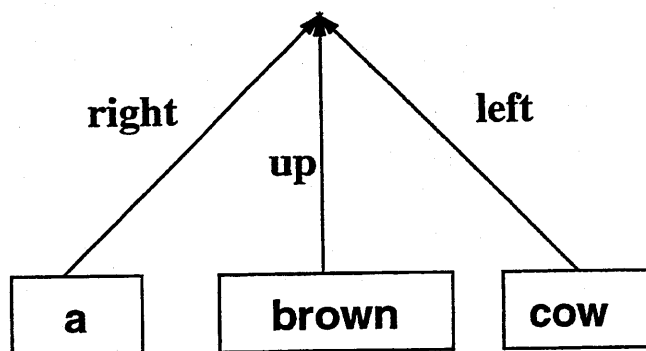


Figure 5. Representation of constituents and a labeling of extensions in SPAT-TER.

context by assigning probabilities to the shift and reduce actions of LR table.

Tagami et al. [49] incorporate the number of modifiers to a non-terminal into their parsing model. In their model, each non-terminal in a CFG rule has a variable that expresses the number of modifiers to the non-terminal. This is based on the assumption that modification likelihood to a certain modifiee is influenced by the other modifiers to the modifiee.

Hogenhout et al. [19] defines the semantic-head for each CFG rule, and every non-terminal has the variable that keeps the semantic-head of the non-terminal. For the rule $Y \rightarrow Y_1 \dots Y_m$, they consider the conditional probability $P(Y_1 : h_1, \dots, Y_m : h_m \mid Y)$, which is dependent on semantic head h_i of non-terminal Y_i . They also take into account the number of words that each non-terminal dominates.

2.1.3 Lexicalization in Various Formalisms

Lexicalization is also the trend in other formalisms, such as Lexicalized Tree Adjoining Grammars and Link Grammars. These approaches pursue to use highly lexicalized information to construct more context-sensitive models.

LTAG is a tree-rewriting system that combines trees of a large domain with *adjoining* and *substitution*. Figure 6 illustrates one derivation of the sentence *John eats peanuts hungrily*. The mark \circ indicates substitution operation that substitutes a subtree on the leftmost identically labeled leaf node of another

subtree. The square mark indicates adjoining operation that insert the right most subtree into the leftmost identically labeled internal node. Each subtree is a lexicon for LTAG.

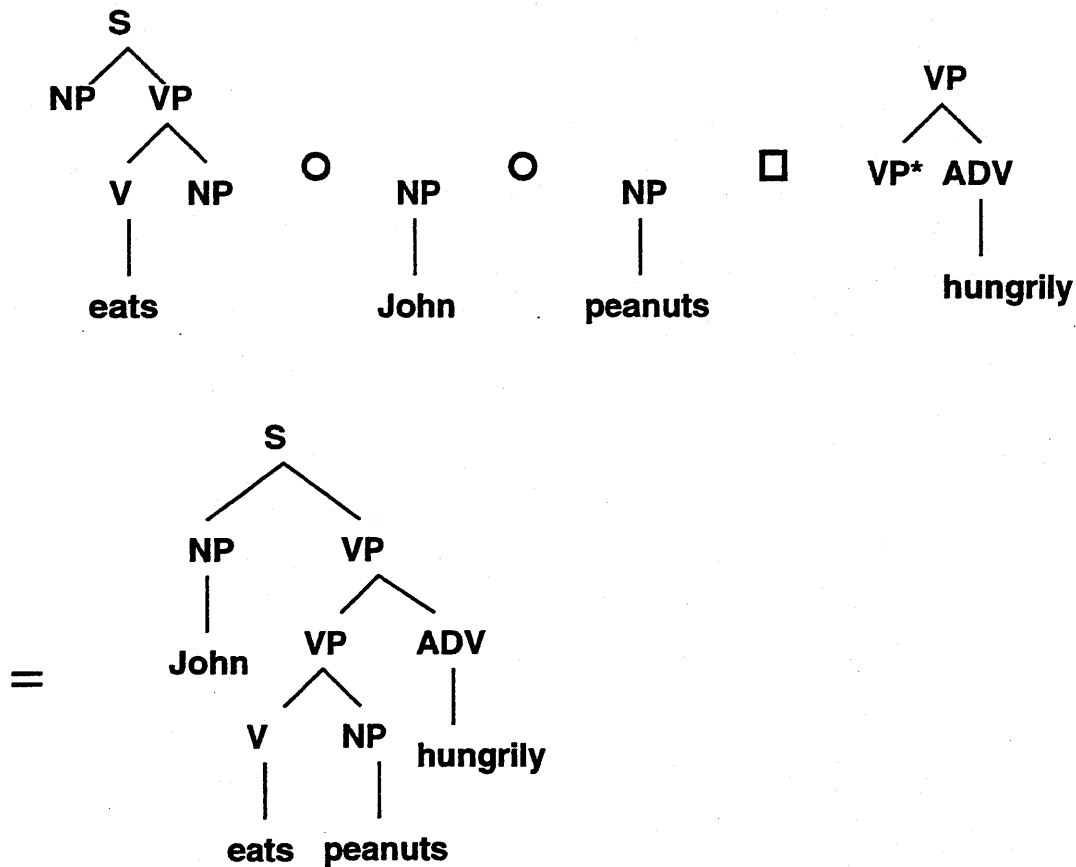


Figure 6. Example of a tree adjoining step.

Schabes [41] formally defines the notion of stochastic lexicalized tree-adjoining grammar (SLTAG). In SLTAG, the lack of context-sensitivity is overcome by assigning probabilities to larger structural units. However, it is not always evident which structures should be considered as elementary structures, and there are ambiguities in the way to construct a whole tree structure. He presents an algorithm for computing the probability of a sentence generated by a SLTAG, and introduces an inside-outside-like iterative algorithm for estimating the statistical parameters for a SLTAG from a training corpus.

Bod [3] describes DOP (Data Oriented Parsing), which is different from other statistical approaches in that it omits the step of inferring a grammar from a

corpus. Instead, an annotated corpus is directly used as a stochastic grammar. An input string is parsed by combining subtrees from the corpus, and the probability of each parse tree is defined by the sum of the probabilities of all the possible derivations for the tree. In this method, every subtree except the one constructed from one node can be considered as an elementary structure. Calculating the probability of a parse tree by exhaustively calculating all of its derivations makes the time complexity to be exponential, since the number of derivations of a parse tree of an input string grows exponentially with its length. To calculate the probability of a parse tree and to make its error arbitrarily small in polynomial time, he applies the Monte Carlo technique.

Lafferty [29] presents a probabilistic Model of Link Grammars proposed in [46]. What distinguishes this formalism from many other context-free models is the absence of explicit constituents, as well as a high degree of lexicalization. In Link Grammar, a sequence of words is accepted by the grammar if there exists at least one valid *linkage* for all the words in the sentence. A *linkage* is a connected graph structure of words by *links* (labeled arcs). A linkage is valid if the resulting graph is connected and planar with arcs drawn above the words, and at most one arc connects a given pair of words. Figure 7 shows a Link Grammar parse (linkage) for an English sentence. They describe an algorithm for determining

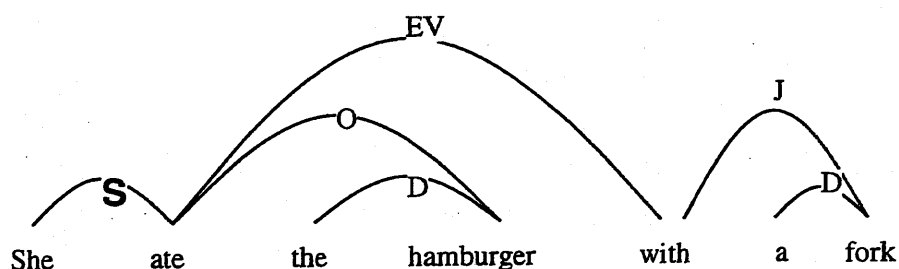


Figure 7. A link grammar parse.

the maximum-likelihood estimates of the parameters of their model.

Under the formalism of probabilistic Link Grammar, Fong and Wu [12] present an Expectation Maximization training method for estimating the probabilities, and give a procedure for learning some simple lexicalized grammar.

As for a method to assign a probability to a derivation tree, there are top down and bottom up approaches. Charniak [5] adopts a top down approach. Consider the case to assign a probability to the noun phrase "Corporate profits". In Figure 8, each non-terminal node of the parse tree is assigned with the phrase

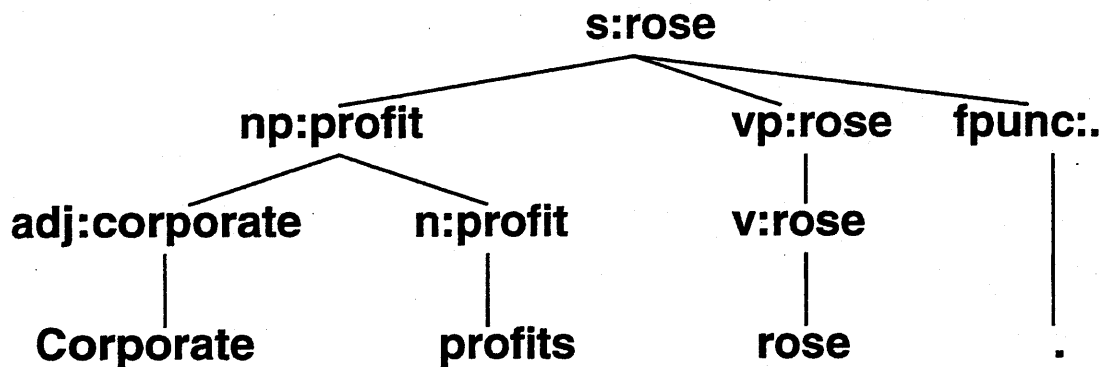


Figure 8. Parse tree of a simple sentence.

type (e.g. a noun-phrase, np) and the head of the constituent (most important lexical item). The probability of a parse tree is calculated in the following steps.

1. Calculate the probability of the phrase head
2. Calculate the probability of the form of the constituent, given the head of a phrase
3. Recursing to find the probabilities of sub-constituents

In Step 1, Charniak [5] assumes that the probability of a head only depends on its type (part-of-speech), head word of the parent node, and the type of the parent node. In other words, the probability of the head word "profits" in the phrase "Corporate profits" can be defined $P(\textit{profits} \mid \textit{rose}, \textit{np}, \textit{s})$. In Step 2, the probability of the form of the constituent given its head is defined as the probability that a constituent c is expanded using the grammar rule r given that c is of type t , is headed by h , and has parent of type l , that is $P(r \mid h, t, l)$.

Charniak compares his model with Collins' and Magerman's models and concludes that statistics on individual words outperforms the models based on word

classes, and that it is worth collecting statistics on some more detailed information.

Collins [7] proposes three new parsing models. **Model 1** is essentially a generative version of the model described in Collins 96 [6]. **Model 2** is an extension of **Model 1** that makes the complement/adjunct distinction by adding probabilities over subcategorisation frames for head words. **Model 3** is an extension of **Model 2** that gives a probabilistic treatment of wh-movement. He achieves 88.1/87.5% constituent precision/recall, an average improvement of 2.3% over Collins 96 [6].

Our statistical dependency analysis model is similar to Collins 96 model in that both model use the word collocation directly. As for a comparison we explain Collins 96 model in detail in section 3.3.3.

2.1.4 Direct Use of Words Associations

As an extreme of lexicalization, there are parsing methods that do not use any grammar rules.

Collins [6] describes a parser based on the probability of a dependency relation (defined by the collocation probability of head words) in the parse tree. The correspondence between phrase structure (tree structure) and dependency pairs is given in Figure 9. Standard bigram probability estimation techniques are extended to calculate probabilities of dependencies between pairs of words. In this way, this approach is similar to Link Grammars but the word association probability of a bigram is directory used. For the precise estimation of the collocation probability of head words, he distinguishes the cases that have different distance properties such as the number of words, verbs, and punctuations between a dependency pair. Then the probability of each dependency pair that has different distance properties is estimated from these separate data. Tests using Wall Street Journal data show that the method performs at least as well as SPATTER[31], in spite of the model being simpler than SPATTER.

The difference of our model to Collins' model principally comes from the properties of a dependency structure of a Japanese sentence. First, the type of modification relation (dependency relations) is unqiely determined by the function words or the ending form of the modifier. Second, the modifier always modifies a phrase that follows itself, because Japanese is a head-final language.

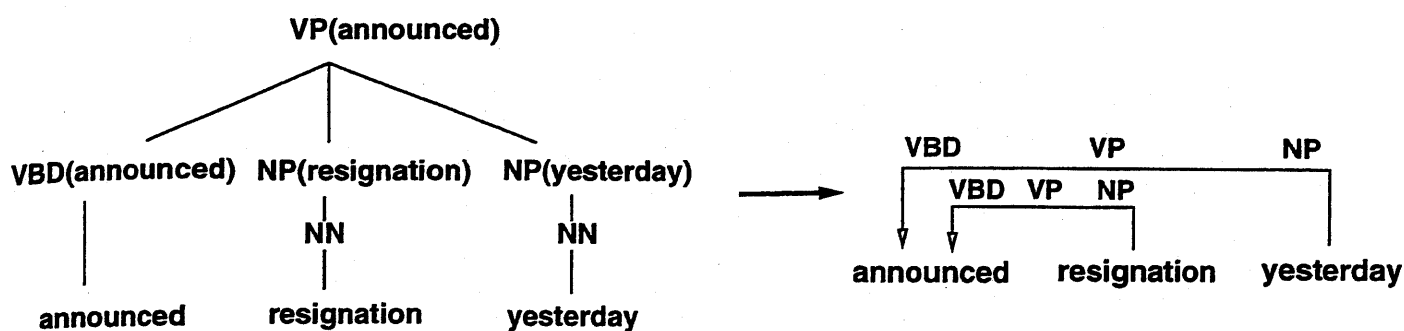


Figure 9. Each constituent with n children (in this case $n = 3$) contributes $n - 1$ dependencies.

Alves [1] uses mutual information drawn from a corpus to select the most likely dependency structure of a Japanese sentence. He calculates mutual information of each dependency relation and chooses the structure for which the product of the mutual information of dependency relations is maximized.

2.1.5 Decision Tree Model

Haruno [17] applies the decision tree learning technique [38] to the task of selecting effective features to be incorporated into the dependency analysis model of [14]. In Fujio [14], various features including the lexical form of each word are evaluated through dependency analysis performance test and an optimal set of features is manually selected, while in Haruno [17], the optimal set of features is automatically selected by using the decision tree technique, although only lexical forms of a few hundred frequent words are considered as feature candidates. Both methods are evaluated using EDR bracketed corpus [9] and achieved about 85~86% accuracies of segment level dependencies (segment roughly corresponds to the notion of *chunk*. For the details, see section 3.2.).

In the decision tree learning, a feature that most decreases the entropy on target class is selected, and the training set is divided into subsets according to the value of the feature. In Haruno's study [17], in each step of decision tree construction, only one of the feature of the modifier or the modifiee is examined, in other words, a set of features of the modifier and the modifiee that is effective

when considered simultaneously, may be overlooked.

2.1.6 Maximum Entropy Approach

In English, Ratnaparkhi [39] uses the maximum entropy model for the statistical parser. He defines the context predicate that capture the some aspect of the derivation history, and estimate the probability of each derivation step by the maximum entropy model.

Ehara and Uchimoto [10, 50] use the maximum entropy method to estimate the probabilities of dependency relations of Japanese sentences.

The maximum entropy method estimates the most uniform probability distribution of stochastic variables under the constraint where the expected value of each stochastic variable calculated from observed data and calculated from the estimated distribution are the same.

Ehara [10] defines the conformity value by the probability that two *bunsetsu*' are in a dependency relation and the probability that two *bunsetsu*'s are not in a dependency relation. These probabilities are estimated by the maximum entropy estimation based on the *bunsetsu* features. For the maximum entropy estimation, he uses maximally two combination of *bunsetsu* features. He evaluates the precision of the modification of the *bunsetsu* which has case particle “*が*” for the sentences of TV news articles.

In Uchimoto [50], the probability of dependency relations are also defined by *bunsetsu* features such as part-of-speech tags, head words and features which appear between the pair of *bunsetsu*'s, such as the number of *bunsetsu*'s and punctuations. But he applies the maximum entropy method to estimate the probability distribution. He uses the combination of those features (lower than 5). Parsing accuracy on the Kyoto University corpus shows that conventional precision is achieved from relatively small training data.

2.2 Statistical Partial Parser

A number of statistical parsing methods have been proposed. Most of the systems focus on the performance of a full parsing of sentences, and do not discuss the performance of a partial parsing (the method that achieves higher precision at

the cost of lower recall) or a redundant parsing (the method that achieves higher recall at the cost of lower precision), which is crucial for some applications, such as information retrieval, pre-processing of corpus annotation and acquisition of linguistic knowledge.

In Japanese text, Inui [20] and Fujio [14] pay attention to statistical partial parsing methods. In Inui [20], after calculating n -best parses, a statistical parser sums up the probabilities of the parse trees that contain a particular dependency relation. Then the parser normalizes the total sum of the probabilities of each dependency relation by the sum total of the probabilities of n parses. If this value is higher than a particular threshold, the parser outputs these dependency relations as a partial parse results. In section 4.1, we propose three partial parse methods including the same method used in Inui [20] and compare the performance under our statistical model. Furthermore, in section 4.2, we propose the statistical redundant parsing method, and evaluate the performance by the relation of *bunsetsu* level recall and coverage.

3 Coordinate Structures

In Japanese sentences, a coordinate structure is one of major causes of syntactic ambiguity in a long sentence. The main problem is to identify the *scope* of constituents of a coordinate structure.

Consider the following sentence.

- お年寄りたちに

車のライトの届く距離や、1234 服装の、1色の、2違い 3 による確認できる時間の 4 差などを
体験してもらう。

The correct constituents that construct the coordinate structure are 4 and 5', but there are $4 \times 5 = 20$ ambiguities for the construction of a coordinate structure.

Kurohashi [26] focuses on this point. They propose the method to compare the candidate constituents of the coordinate structure, and if they are sufficiently similar, their system marks the candidate constituents as the correct constituents of the coordinate structure. For the measure of segment similarity, they use category match, word match and semantic distance defined by a thesaurus, and then select the most likely constituent pair by using the dynamic programming method.

Once the coordinate structure in a sentence is determined, the rest of dependency structure is determined by conventional rules. They report 65%(97/150) accuracy calculated by the number of correct sentences.

Suganuma [48] describes a method for extracting coordinate structures that consist of noun phrases, and for estimating the coordinated constituents. He estimates the constituents of a coordinate structure by using clues on Japanese characters, heuristics to determine the coordinate structure, and structural similarity between two constituents. His aim is to support a document writer by indicating the presence of coordinate structures, which is a cause of communication gaps between the writer and the reader.

Nominal Coordinate Structure	[読点], [中点], と, も, や, かつ, だけで(は)なく, および, または, ならびに, あるいは, もしくは, 及び, 又は, 並びに, 或は
Partial Coordinate Structure	[読点], だけで(は)なく, および, または, ならびに, あるいは, もしくは, 及び, 又は, 並びに, 或は

Table 4. Keys to coordinate structures

There are two disadvantages in these approaches. Both of these approaches first identify the existence of coordinate structures by finding key words or patterns (see Table 4) that give an evidence of the existence of a coordinate structure, and then identify the scope of coordinate structures by examining the words sequence similarity or structural similarity. However these keys to coordinate structures or heuristics to find the scope of coordinate structures must be developed by a language expert. Constructing and maintaining these rules are a laborious task. The other disadvantage is that dependency analysis is performed based on the identified coordinate structures. In other words, the process of coordinate structure identification and dependency analysis is sequential.

In this dissertation, we do not resort to the extra heuristics for finding the similarity between the constituents of a coordinate structure (see section 6). We propose a method to treat the coordinate structure identification and the dependency analysis in a uniform way.

2.4 Subordinate Clauses

2.4.1 Scope Embedding Preference

By manually analyzing several raw corpora, Minami [34, 35] classifies various types of Japanese subordinate clauses into three categories, which are assumed to comply with a total ordering of the embedding relation of their scopes. Shirai et al. [44] reclassified and specified the three categories of Japanese subordinate clauses proposed in Minami [34, 35] in order to make them more suitable for automatic processing by computer programs. In the following, we describe more specified version of Japanese subordinate clause classification by Shirai et al. [44]. By manually analyzing 972 newspaper summary sentences, they extracted 54 clause final function expressions of Japanese subordinate clauses and classified them into the following three categories according to the embedding relation of their scopes.

Category A: Seven expressions representing simultaneous occurrences such as “*Verb₁ to-tomoni (Clause₂)*” and “*Verb₁ nagara (Clause₂)*” (both of these two expressions mean that the events expressed by *Verb₁* and *Clause₂* occur simultaneously).

Category B: 46 expressions representing cause and discontinuity such as “*Verb₁ te (Clause₂)*” (in English “*Verb₁ and (Clause₂)*”) and “*Verb₁ node*” (in English “*because (subject) Verb₁ ...*”).

Category C: One expression representing independence, “*Verb₁ ga*” (in English, “*although (subject) Verb₁ ...*”).

The category A has the narrowest scope, while the category C has the widest scope, i.e.,

$$\text{Category A} \prec \text{Category B} \prec \text{Category C}$$

where the relation ‘ \prec ’ denotes the embedding relation of scopes of subordinate clauses. Then, scope embedding preference of Japanese subordinate clauses can be specified as below:

Scope Embedding Preference of Japanese Subordinate Clauses

1. A subordinate clause can be embedded within the scope of another subordinate clause which inherently has a same or wider scope preference.
2. A subordinate clause cannot be embedded within the scope of another subordinate clause which inherently has a narrower scope.

For example, in the case of the above three classification, a subordinate clause of Category B can be embedded within the scope of another subordinate clause of Category B or Category C, but not within that of Category A.

2.4.2 Applying Scope Embedding Preference to Dependency Preference

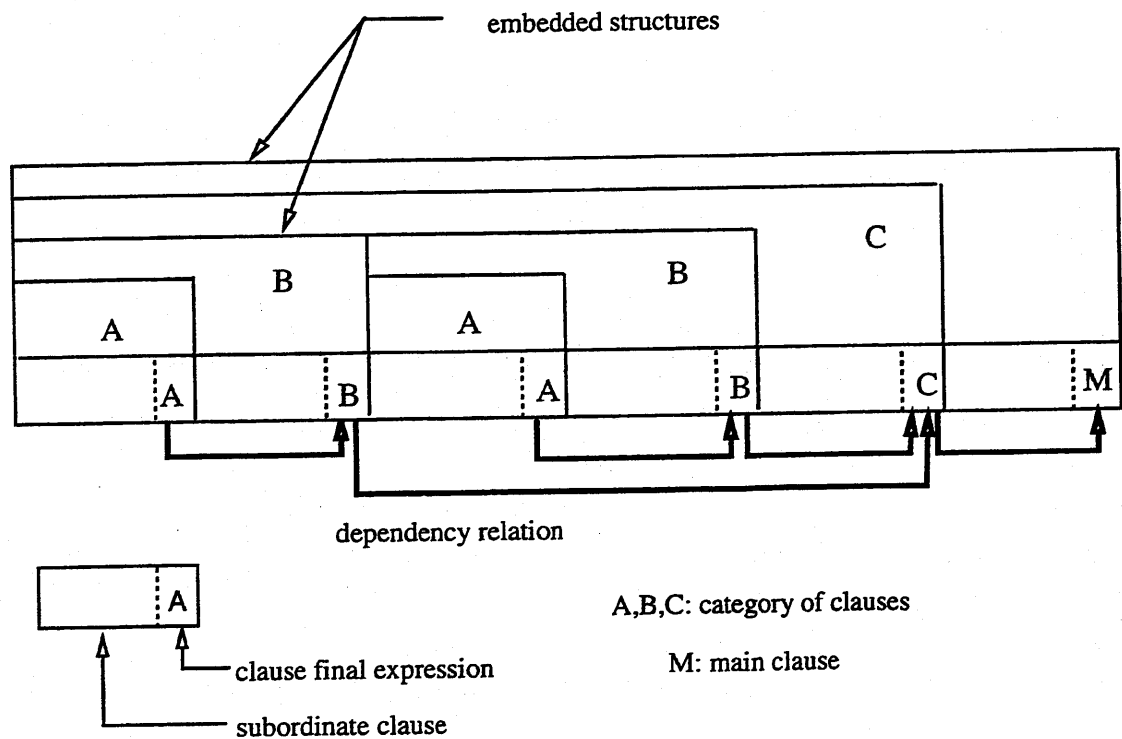


Figure 10. Embedded structures of Japanese clauses and dependency relationships between clauses.

Scope Embedding Preference of Japanese Subordinate Clauses has the close relation to Dependency Preference.

Fukumoto et al. [15] and Shirai et al. [44] restate the relation between scope embedding and dependency of two subordinate clauses as the form of the following dependency preference of subordinate clauses, and apply the scope embedding preference of Japanese subordinate clauses to rule-based Japanese dependency analysis

Dependency Preference of Japanese Subordinate Clauses

1. A subordinate clause can modify another subordinate clause which inherently has a same or wider scope.
2. A subordinate clause cannot modify another subordinate clause which inherently has a narrower scope.

Shirai et al. [44] also added the following more specific classification of subordinate clause types to the above three categories.

1. Whether the subordinate clause is punctuated or not. A punctuated clause has a wider scope than non-punctuated ones of the same category.
2. Two level strength of discontinuity of subordinate clauses. A clause with strong discontinuity has a wider scope than that of the same category with weak discontinuity.
3. Four level distinction of state/action of the head predicates of subordinate clauses. A clause with a more action predicate has a wider scope than that of the same category with a more state predicate.

However, in those manual approaches, their coverage against a large corpus containing more than 200,000 sentences such as EDR corpus is quite low², since categories of subordinate clauses are obtained by manually analyzing a small number of sentences. In Chapter 5, we propose a method of learning dependency preference of Japanese subordinate clauses from a bracketed corpus.

² For example, in our implementation, the coverage of the categories of [44], which are more specific than those of [34] and [35] and are designed for the purpose of automatic processing by computer programs, is only 30% for all the subordinate clauses included in the whole EDR corpus.

2.5 Topicalization

Most of the Japanese dependency analysis systems assume that every segment except the last one in a sentence modifies one segment in the sentence. However, there are linguistic phenomena in which one segment modifies two or more segments or to modify no other segment. One of them is *topicalization*:

- 太郎は 本は 読んだが、ノートは 読まなかった。
Taro read a book, but didn't read a note.

In this example, “太郎は” is a topic phrase, and modifies both “読んだが” and “読まなかった”. Fukumoto [15] proposes a framework of Restricted Dependency Grammar (RDG). An RDG consists of a set of simple rewriting rules including context dependent rules (type-1 rules), and can treat topic phrases naturally. In the RDG formalism, one modifier can modify more than one modifiee. To suppress the generation of useless solutions, she incorporates some types of constraints about how modifiers can modify modifiees. In the case of topicalization, we assume that topics modify the last predicate.

In Japanese, there is another type of topicalization [25].

a-1) 象は 鼻が 長い

As for an elephant, the nose is long.

a-2) 鼻は 象が 長い

As for a nose, the elephant's is long.

b-1) 東京は 物価が 高い

As for Tokyo, the price is high.

b-2) 物価は 東京が 高い

As for the price, the Tokyo's is high.

In both pairs of these examples, there are two grammatical subjects, only one of which can semantically modify the predicate. This type of ambiguity depends on the semantic relationship between the modifier and the modifiee. For instance in a-1) and a-2), “鼻が” modifies “長い” and “象は” does not modify “長い”. In our formalism, we do not deal with semantic interpretation, thus we simply assume that both phrases modify the predicate.

2.6 Ellipsis

Ellipsis is another phenomenon that causes multiple modification. In Japanese, the subject is often omitted, when it is easily retained from the context. The omitted subject is called a zero anaphora. In the following examples, ϕ denotes zero anaphora.

a) ジョンができれば ϕ するでしょう。

If possible, John will do it.

b) もしこの岩を持ち上げられるなら, ϕ 動かしてくれませんか。

If you can hold up this rock, please move it.

In the first example, ϕ is an ellipsis that refers to “ジョン”, and this phrase modifies both “できれば” and “するでしょう”. The second example, ϕ is an ellipsis that refers to “岩”, and this phrase modifies “持ち上げるなら” and “動かしてくれませんか”. These phenomena are also treated in the RDG formalism [16]. By using context dependent rewriting rules (type-1 rules), the referent of an omitted element can have more than one dependency relation.

In these cases, we consider that “ジョンが” should modify the former predicate “できれば” and in the same way “岩を” should modify “持ち上げる”.

3. Basic Dependency Analysis Model

This Chapter describes a statistical parsing model based on the features of segments (*bunsetsu*). A segment (*bunsetsu*) is a unit of dependency analysis. We aim to find out the most likely combination of dependency relations between segments (*bunsetsu*).

This Chapter presents the statistical dependency analysis of Japanese sentences and is organized as follows:

Section 3.1 defines the constraints on dependency structure. Section 3.2 defines segments (*bunsetsu*) and its features. Section 3.3 gives the statistical model of dependency structure analysis. Section 3.4.4 describes the actual parsing algorithm.

The model described here will be combined or extended in the later Chapter to handle subordinate clauses and coordinate structures (see Chapter 5, 6).

3.1 Constraints

We place the following constraints on the dependency structure of sentences.

1. Every segment except the right most one modifies exactly one of its succeeding phrases in the sentence.
2. No two modification may cross each other (crossing constraint).

Those are acceptable constraints on Japanese dependency structure.

3.2 Definition of a Segment

3.2.1 Composition of a Segment

A segment is a unit of dependency structure analysis in this study. A segment consists of one or a sequence of content words and its succeeding function words. The composition of segments can be defined by regular expressions.

The following is sample definition of a segment:

(名詞//+ 助詞//*)

This regular expression matches a sequence of nouns followed by zero or more particles.

There is a room for users to customize the definition of segment in our system, to take care of exceptional cases which do not fall into a general pattern, and to manage conceptual differences between system designs.

3.2.2 Definition of Segment Features

For each segment, a set of features are assigned. Those features are used for the estimation of the probability of a dependency relation.

The features of a segment are defined by the result of morphological analysis. They include part-of-speech (POS) tags, conjugation types, punctuation, and other grammatical or surface information. Some features are determined not directly from the modifier segments. and modifiee . For instance, the number of segments between the modifier and the modifiee can also be a feature. In the sequel, we use the following symbols to represent segment features:

- h_i : head word
- t_i : head pos
- c_i : head class
- r_i : relation type
- p_i : punctuation
- d_{ij} : the number of segments between the dependency pair
- p_{ij} : the number of punctuations between the dependency pair

The basic rules for assigning segment features are as follows.

- The rightmost content word in a segment is the head feature.
- Morphological information (such as a word, tag, and conjugation form) of function words constitutes the relation type.

[非散布地域より、]₁[発生率が]₂[高いとの]₃[調査結果が]₄[ある。]₅
 [no diffusion area]₁[incidence]₂[higher]₃[a finding]₄[there is]₅.

	Modifier			Modifiee					
	h_i	r_i	p_i	d_{ij}	p_{ij}	h_j	r_j	p_j	
1 → 4	非散布地域	より	1	1	0	高い	基本形-と-の	0	
2 → 3	発生率	が	0	0	0	高い	基本形-と-の	0	
3 → 4	高い	基本形-と-の	0	1	0	ある	基本形	2	
4 → 5	調査結果	が	0	0	0	ある	基本形	2	

Table 5. Examples of segment features.

The head feature takes the form of either the word form, part-of-speech, or the semantic class. We use the Japanese thesaurus 'Bunrui Goi Hyou'(BGH) [37] to define the semantic class.

If a segment has no function words, the relation type is defined by the pair of the tag and the conjugation form of the last content word in a segment.

The features d_{ij} and p_{ij} are defined by the sequence of Base Phrases, which introduce distance measures to the model.

Table 5 gives an example of the features of correct dependency pairs for the sentence which consists of four segments. The first column denotes the dependency relation in the sentence, and the rest of the columns shows the features explained above. The fourth and ninth columns indicate the existence of punctuation. The value 0 indicate there is no punctuation in the segments. The value 1 indicates that the last morpheme of the segment is a comma. The value 2 indicates that the last morpheme of the segment is a full stop.

When defining the features d_{ij} , p_{ij} , it is not necessary to use exact count of phrases or punctuations. For example, while the distinction of dependency relation of distances 1 and 2 may be important, the difference of distances 5 and 6 is not big difference. The same value for the distance can be assigned when their distance is larger than 2. Table 6 shows the variations we used for the features d_{ij} and p_{ij} .

Other conceivable distance measures are the number of case particles, or the

variations for distance features	d_{ij}	p_{ij}
dst1	exact value	exact value
dst2	1,2, and f	exact value
dst3	1,2,and f	1,2,and f
dst4	1,2,B,C	exact value

Table 6. Variations of distance features. The sign “f” means larger than 2. The sign “B” means more than 2 and less than 5. The sign “C” means larger than 5.

be effective, when a modifier segment has a case particle as a relation type. But as a result of repeating preliminary experiments, both of those distance measures exhibited poorer performance than the case using d_{ij} , p_{ij} (We will discuss the reason in Chapter 7). In section 3.6, we will only show the results of using d_{ij} and p_{ij} .

3.3 Statistical Model

We propose a probabilistic model based on the features of segments defined in the previous section.

First, we introduce notational conventions. The symbol S denotes a sentence. $T = \langle w_1, t_1 \rangle, \dots, \langle w_n, t_n \rangle$ is a sequence of word and tag pairs, where w_i is the i -th word and t_i is the i -th part-of-speech tag. $F = \langle b_1, f_1 \rangle, \dots, \langle b_m, f_m \rangle$ is a sequence of pairs of segment and features, where b_i represents a segment and f_i represents the features associated with b_i . We use the notation $Dep(i) = j$ to indicate that the i -th segment in the sequence modifies the j -th segment.

The subscripts m , and n stand for the number of segments and words in a sentence, respectively. L is the sequence of dependencies: $L = \langle Dep(1), Dep(2), \dots, Dep(m-1) \rangle$.

In general, a statistical parsing model estimates the conditional probability, $P(P_t | S)$, for each candidate parse tree P_t for a sentence S . In the dependency analysis, the final goal is to identify L rather than P_t . Our aim is to find the L, F, T that maximize the probability $P(L, F, T | S)$.

The most likely dependency analysis under the model is then:

$$\operatorname{argmax}_{L,F,T} P(L, F, T|S) = \operatorname{argmax}_{L,F,T} P(L|F, T, S) P(F|T, S) P(T|S) \quad (2)$$

We assume that the composition of a segment only depends on word/tag pairs, hence $P(F | T, S) = P(F | T)$, and assume that a dependency structure L can be determined only by segment features, thus $P(L | F, T, S) = P(L | F)$. The equation (2) is now written into:

$$\operatorname{argmax}_{L,F,T} P(L|F) P(F|T) P(T|S)$$

For simplicity, we assume that the morphological analysis and the segment composition are both deterministic. For the morphological analysis, we use the most likely output of the Japanese morphological analyzer ChaSen [33]. For the segment composition, we use a finite state transducer constructed from regular expressions of word/tag pairs (see section 3.2). In other words, we assume $P(T | S) = 1$ and $P(F | T) = 1$.

What we need to do now is to find L_{best} that satisfies the following equation.

$$L_{best} = \operatorname{argmax}_L P(L | F) \quad (3)$$

Suppose that each dependency relations are mutually independent, the right hand side of the equation (3) can be expanded as follows:

$$P(L | F) = \prod_{i=1}^{m-1} P(i \xrightarrow{\text{rel}} j | \mathbf{f}_1, \dots, \mathbf{f}_m) \quad (4)$$

The term $P(i \xrightarrow{\text{rel}} j | \mathbf{f}_1, \dots, \mathbf{f}_m)$ indicates the probability that the segment b_i modifies the segment b_j , given the set of features, $\mathbf{f}_1 \dots \mathbf{f}_m$, after segments are constructed.

Our model defines the probability $P(i \xrightarrow{\text{rel}} j | \mathbf{f}_1, \dots, \mathbf{f}_m)$ as the product of *Head Collocation Probability* and *Distance Probability*. Those probabilities are defined by the features described in section 3.2.

$$P_h^{ij} = P(i \xrightarrow{\text{rel}} j | h_i, r_i, p_i, h_j, r_j, p_j) \quad (5)$$

$$P_d^{ij} = P(i \xrightarrow{\text{rel}} j | r_i, p_i, d_{ij}, p_{ij}) \quad (6)$$

P_h^{ij} indicates *Head Collocation Probability* for the dependency relation pair b_i and b_j . P_d^{ij} indicates *Distance Probability* for the dependency relation pair b_i and b_j .

Note that the definition of *Distance Probability* uses the relational and punctuation features of the modifying side, r_i and p_i , not the modified side. This is because the relation type, and the existence of a punctuation have close relation to the distance of dependency pair of segments. At the same time, we assume that the features r_i and p_i explain the tendency of the distance property of dependency pairs well, thus *Distance Probability* is assumed to be independent from other features.

The product of *Head Collocation Probability* and *Distance Probability* defines the probability of the dependency probability.

$$P(i \xrightarrow{\text{rel}} j \mid \mathbf{f}_1, \dots, \mathbf{f}_m) = P_h^{ij} \times P_d^{ij}$$

The division of the probability $P(i \xrightarrow{\text{rel}} j \mid \mathbf{f}_1, \dots, \mathbf{f}_m)$ into P_h^{ij} and P_d^{ij} contribute to reduce the sparseness of training data.

Those probabilities are estimated from the training corpus by the following equations.

$$P_h^{ij} \approx \frac{C(i \xrightarrow{\text{rel}} j, h_i, r_i, p_i, h_j, r_j, p_j)}{C(h_i, r_i, p_i, h_j, r_j, p_j)} \quad (7)$$

$$P_d^{ij} \approx \frac{C(i \xrightarrow{\text{rel}} j, r_i, p_i, d_{ij}, p_{ij})}{C(r_i, p_i, d_{ij}, p_{ij})} \quad (8)$$

$C(\dots)$ in the above equation represents the frequency of the set of features appeared in the training corpus. $C(i \xrightarrow{\text{rel}} j, \dots)$ represents the frequency of the set of features of dependency relation pair appeared in the training corpus.

3.3.1 Training of Head Collocation Probability

Ideally, the word form of the head word should be used as the head feature h_i . But, then, head collocation probability suffers from the problem of the data sparseness.

Variations of the Head Feature

To cope with the problem of the data sparseness, we define the following five variations of *Head Collocation Probability* that differs in the head features used

for the probability estimation.

$$P_h^{ij} = P(i \xrightarrow{\text{rel}} j \mid h_i, r_i, p_i, h_j, r_j, p_j) \quad (9)$$

$$P_h^{ij} = P(i \xrightarrow{\text{rel}} j \mid h_i, r_i, p_i, t_j, r_j, p_j) \quad (10)$$

$$P_h^{ij} = P(i \xrightarrow{\text{rel}} j \mid t_i, r_i, p_i, h_j, r_j, p_j) \quad (11)$$

$$P_h^{ij} = P(i \xrightarrow{\text{rel}} j \mid c_i, r_i, p_i, c_j, r_j, p_j) \quad (12)$$

$$P_h^{ij} = P(i \xrightarrow{\text{rel}} j \mid t_i, r_i, p_i, t_j, r_j, p_j) \quad (13)$$

The symbol t_i is a part-of-speech tag of the head word h_i , and the symbol c_i is the semantic class that h_i belongs to. We use the Japanese thesaurus ‘Bunrui Goi Hyou’(BGH) [36] for the semantic classes. There are many smoothing techniques [11, 23]. We adopted a method similar to back-off [8] estimation for its simplicity of implementation.

BGH is a thesaurus of six-layered abstraction hierarchy in a tree form, where more than 80,000 words are allocated at the leaves. We use the classes at the second layer from the bottom of the hierarchical tree for the semantic classes. For the limitation of computational resources, we decided to select the semantic class at the same levels in the thesaurus, though various combinations of the selection of semantic classes are possible. In other words, the model learns 5 variations (from second to six-th layer) for the equation (12).

When training the model, we estimate all of the above statistics. The maximum likelihood estimation of those probabilities (from the equation (10) to (13)) can be calculated by the following equations.

$$P_h^{ij} \approx \frac{C(i \xrightarrow{\text{rel}} j, h_i, r_i, p_i, t_j, r_j, p_j)}{C(h_i, r_i, p_i, t_j, r_j, p_j)} \quad (14)$$

$$P_h^{ij} \approx \frac{C(i \xrightarrow{\text{rel}} j, t_i, r_i, p_i, h_j, r_j, p_j)}{C(t_i, r_i, p_i, h_j, r_j, p_j)} \quad (15)$$

$$P_h^{ij} \approx \frac{C(i \xrightarrow{\text{rel}} j, c_i, r_i, p_i, c_j, r_j, p_j)}{C(c_i, r_i, p_i, c_j, r_j, p_j)} \quad (16)$$

$$P_h^{ij} \approx \frac{C(i \xrightarrow{\text{rel}} j, t_i, r_i, p_i, t_j, r_j, p_j)}{C(t_i, r_i, p_i, t_j, r_j, p_j)} \quad (17)$$

Table 7. Features used for estimating *head-collocation probability*.

Modifier			Modifiee		
Head	Relation Type	Punctuation	Head	Relation Type	Punctuation
非散布地域	より	1	高い	基本形と-の	0
非散布地域	より	1	高い	基本形と	0
非散布地域	より	1	高い	基本形の	0
非散布地域	より	1	高い	-	0
発生率	が	0	高い	基本形と-の	0
発生率	が	0	高い	基本形と	0
発生率	が	0	高い	基本形	0
発生率	が	0	高い	-	0
高い	基本形と-の	0	ある	基本形	2
高い	基本形と	0	ある	基本形	2
高い	基本形	0	ある	基本形	2
高い	基本形と-の	0	ある	-	2
高い	基本形と	0	ある	-	2
高い	基本形	0	ある	-	2
調査結果	が	0	ある	基本形	2
調査結果	が	0	ある	-	2

Variation of the Relation Type

If a segment has succeeding function words at the end, the relation type of the segment cannot be uniquely determined according to which combination of the functional words are to be adopted.

For instance, the relation type of 3rd segment in Table 5 has at least three variations, “と-の”, “の”, and “と”.

Table 7 shows the patterns of features for *Head Collocation Probability* estimation for the dependency relations in Table 5. Note that modifier's relation type always has the right most function words of the modifier in its constituent, and that modifiee's relation type always has the left most function word (the conjugation form, if the head word has conjugation forms) in its constituent. We consider that the right-most function word captures the modifier's property, and the left

Relation Type	Punctuation	Punctuation between Dependency Pair	Segments between De- pendency Pair
より	1	0	1
が	0	0	0
形容詞/連用	0	0	1
が	0	0	0

Table 8. Features used for estimating Distance probability.

most function word (or the conjugation form, if the head word has conjugation forms) captures the modifier's property.

The model also counts the case where the modifiee does not take the relation type. Those statistics are used in the way similar to back-off [8] method (see section 3.4.2).

3.3.2 Training of distance probability

Distance probability also uses the modifier's relation type as a feature of the probability estimation. Then, if a modifier has succeeding function words at the end, the model can learn a different set of features which differs in the portion of relation type.

Table 8 shows the patterns of features for *Distance Probability* estimation for the dependency relations in Table 5.

Those statistics are used in the way similar to back-off [8] method (see section 3.4.3).

3.3.3 The Collins' Model

There are several approaches to approximate the probability $P(i \xrightarrow{\text{rel}} j \mid \mathbf{f}_1, \dots, \mathbf{f}_m)$ in equation (4). In the field of statistical parsing of English text, Collins [6] proposes the model based on the collocation probability of head words. As a comparison, we briefly explain the statistical model of Collins [6].

To approximate the probability $P(i \xrightarrow{\text{rel}} j \mid \mathbf{f}_1, \dots, \mathbf{f}_m)$ in equation (4), Collins [6]

considered the probability $F_c(R | \langle w_i, t_i \rangle, \langle w_j, t_j \rangle)$. This is the probability that $\langle w_i, t_i \rangle$ modifies $\langle w_j, t_j \rangle$ with relationship R , given that both $\langle w_i, t_i \rangle$ and $\langle w_j, t_j \rangle$ appear in the same sentence. Then, the maximum-likelihood estimate of $F_c(R | \langle w_i, t_i \rangle, \langle w_j, t_j \rangle)$ is as follows:

$$\hat{F}_c(R | \langle w_i, t_i \rangle, \langle w_j, t_j \rangle) = \frac{C(R, \langle w_i, t_i \rangle, \langle w_j, t_j \rangle)}{C(\langle w_i, t_i \rangle, \langle w_j, t_j \rangle)}$$

$C(\langle w_i, t_i \rangle, \langle w_j, t_j \rangle)$ is the number of times both $\langle w_i, t_i \rangle$ and $\langle w_j, t_j \rangle$ are seen in the same sentences in the training data. $C(R, \langle w_i, t_i \rangle, \langle w_j, t_j \rangle)$ is the number of times both $\langle w_i, t_i \rangle$ and $\langle w_j, t_j \rangle$ are seen in the same sentences in the training data and simultaneously $\langle w_i, t_i \rangle$ modifies $\langle w_j, t_j \rangle$ with relationship R .

Then he made the following approximation:

$$P_c(i \xrightarrow{\text{rel}} j | S, B) \approx \frac{F_c(R | \langle w_i, t_i \rangle, \langle w_j, t_j \rangle)}{\sum_{k=1, \dots, m, k \neq i, p \in P} F_c(p | \langle w_i, t_i \rangle, \langle w_k, t_k \rangle)}$$

B stands for the set of BaseNP's, which corresponds to segments in our model. P stands for the set of dependency relationships. Figure 11 illustrates BaseNP's and dependency relations. Dependency structures are derived from tree structures in Penn Tree Bank. Each dependency relation is labeled by a triple of a modifier's non-terminal, a parent's non-terminal, and a head-child's non-terminal.

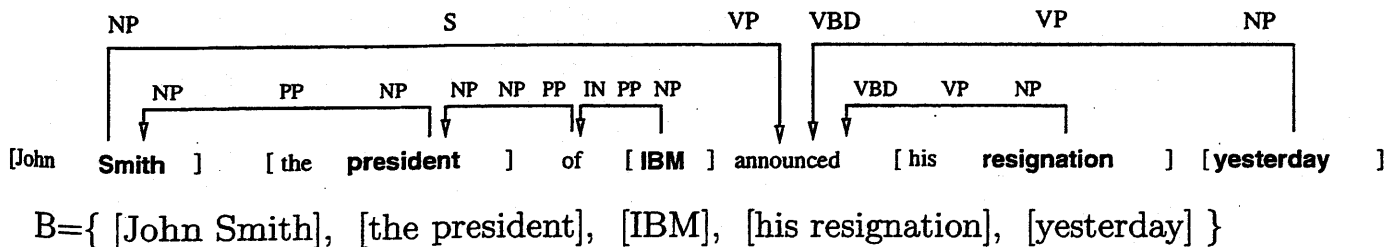


Figure 11. An example of BaseNP and dependency analysis

The probability $P_c(i \xrightarrow{\text{rel}} j | S, B)$ in the Collins' model is the counter part of $P(i \xrightarrow{\text{rel}} j | \mathbf{f}_1, \dots, \mathbf{f}_m)$ of our model in equation (4).

Collins' model does not separate *Head collocation probability* and *Distance Probability* in contrast with our model.

$$P(i \xrightarrow{\text{rel}} j | \mathbf{f}_1, \dots, \mathbf{f}_m) \stackrel{\text{def}}{=} P(i \xrightarrow{\text{rel}} j | h_i, r_i, p_i, h_j, r_j, p_j, d_{ij}, p_{ij})$$

Then, maximum likelihood estimation of $P(i \xrightarrow{rel} j \mid \mathbf{f}_1, \dots, \mathbf{f}_m)$ is done as follows:

$$P(i \xrightarrow{rel} j \mid \mathbf{f}_1, \dots, \mathbf{f}_m) \approx \frac{C(i \xrightarrow{rel} j, h_i, r_i, p_i, h_j, r_j, p_j, d_{ij}, p_{ij})}{C(h_i, r_i, p_i, h_j, r_j, p_j, d_{ij}, p_{ij})}$$

3.4 Parsing Algorithm

In this section, we explain the parsing algorithm under the model described above. First, we describe the actual parsing steps, and then describe back-off method to estimate dependency probabilities.

3.4.1 Parsing Procedure

Dependency analysis is executed in the following order:

1. Tokenize and tag the input Japanese sentence.
2. Identify the segments and define their features.
3. Calculate the probability of each segment pair.
4. Compose the most likely (or n-best) dependency structure based on the statistical model described in section 3.3. We use CKY algorithm in this step.

At the first step, we use the morphological analyzer, ChaSen [33]. There is an ambiguity when applying learned statistics at the third step. It comes from the length of a relation name, and a head feature. See section 3.4.2 and 3.4.3, for the detailed calculation method. In the fourth step, we can use CKY algorithm (see section 3.4.4) to effectively select the most likely (n-best) combination of dependency relations.

3.4.2 Back-Off Method for Head Collocation Probability

Because our model uses the statistics of lexical terms, the statistical model suffers from the sparse data problem.

By using the equation (9) ... (13) take up in section 3.3.1, we define the following three models.

- POS model (13)
- LEX model (9),(10),(11),(13)
- BGH model (9),(10),(11),(12),(13)

For each of the LEX, POS, and BGH models, the system searches the statistical data usage in the following order in the way of back-off method.

- a head word → a pos tag
- a head word → a semantic class → a pos tag

For the length of each relation type, the system searches the statistical data from longer relation types. The parser searches statistical data in the following order.

Consider the case to estimate *Head Collocation Probability* of the relation 3 → 4 in the example in Table 5, under the LEX model. For the first time, our parser searches the sets of features shown in the following order:

	h_i	r_i	p_i	h_j	r_j	p_j	probability
1	高い	と一の	0	調査結果	が	2	not found
2	形容詞	と一の	0	調査結果	が	2	0.7
	高い	と一の	0	名詞	が	2	0.6
3	形容詞	と一の	0	名詞	が	2	0.6
4	高い	と	0	調査結果	が	2	not found
5	形容詞	と	0	調査結果	が	2	0.7
	高い	と	0	名詞	が	2	0.3
6	形容詞	と	0	名詞	が	2	0.6
7	高い	と一の	0	調査結果	-	2	0.4
...							

1. Search three sets of features, " $h_i, r_i, p_i, h_j, r_j, p_j$ ", and " $h_i, r_i, p_i, t_j, r_j, p_j$ ", and " $t_i, r_i, p_i, h_j, r_j, p_j$ ", that has at least one lexical terms.

2. *IF* (a probability is defined for those features,)

{use the average of the found values as the *Head Collocation Probability*.}

ELSE

search " $t_i, r_i, p_i, t_j, r_j, p_j$ "

IF (a probability is defined for the searched features,)

{use that value as the *Head Collocation Probability*.}

ELSE

IF (the modifiee's relation type r_j not null,)

{delete the right most morpheme of r_j and repeat the process from 1~2.}

ELSE

IF (the modifier's relation type r_j consists of more than one morpheme,)

{recover r_j to the original length and delete the left most morpheme of r_i , and repeat the process 1~2.}

ELSE

{use pre-determined probability³, as the *Head Collocation Probability*.}

The first column in Table 3.4.2 shows the order that the system searches the statistical data for the features in the second column, and the third column indicates the dependency probabilities for these features. Suppose the probabilities are defined as shown above, the probability as *Head Collocation Probability* is calculated by the average of found probabilities as $(0.7 + 0.6)/2 = 0.65$.

3.4.3 Back-Off Method for Distance Probability

The parser searches the data for *Distance Probability* that has longer relation type with higher priority.

Consider the case to estimate *Distance Probability* of the relation $3 \rightarrow 4$ of the example in Table 5. For the first time, our parser searches the set of features “の-ようだ (連用) 0 0 0”. If a probability is defined for these features, the parser uses that probability as *Distance Probability*. If a probability is not defined, the parser searches other features “ようだ (連用) 0 0 0” (the left most morpheme of the relation type is deleted). If no data is found for any variation of the relation type, the parser assigns pre-determined probability⁴.

3.4.4 CKY Algorithm

Under the constraints in section 3.1, CKY algorithm (e.g. [22, 54]) is applicable to find the most likely (or n-best) dependency structure. CKY algorithm requires CFG rules to be in Chomsky Normal Form and not to contain ϵ transitions. By regarding a dependency relation as a binary tree structure, it is straightforward to apply the CKY algorithm to dependency tree analysis.

Figure 12 illustrates the CKY algorithm for selecting the most likely combination of dependency relations of a sentence consisting of four segments. The symbol $t_{i,j}$ specifies the segment sequence of length j starting from the i -th segment. The construction of $t_{i,j}$ from $t_{i,k}$ and $t_{i+k,j-k}$ corresponds to adding new dependency relation $k \rightarrow j$.

In Table 12, construction of $t_{1,4}$ from $t_{1,1}$ and $t_{2,3}$ adds the dependency relation $1 \rightarrow 4$. The probability of that dependency structure is calculated from the product of the probability of dependency structures of $t_{1,1}$, $t_{2,3}$ and the probability of

⁴ we set this value to 0.00001

4	$t_{1,4}$			
3	$t_{1,3}$	$t_{2,3}$		
2	$t_{1,2}$	$t_{2,2}$	$t_{3,2}$	
1	$t_{1,1}$	$t_{2,1}$	$t_{3,1}$	$t_{4,1}$
	1	2	3	4

Figure 12. CKY algorithm

dependency relation between the 1st segment and the 4-th Base Phrase. Each $t_{i,j}$ preserves the most-likely (or n-best) dependency structure and its probability.

3.5 Training and Test Corpora

For the training and evaluation of our model, we use EDR[9] bracketed corpus and Kyoto University parsed corpus [27]. EDR corpus is mainly used for the evaluation of basic statistical model and the subordinate clause model. Kyoto University corpus is mainly used for the evaluation of the coordinate structure model (Chapter 6).

3.5.1 EDR Corpus

EDR corpus contains about 208,000 sentences collected from newspapers and magazine articles. We use morphological and structural (bracketing) information contained in the corpus.

Figure 13 shows an example of a sentence from EDR corpus with the morphological, and structural annotations. The following syntactic categories are used in EDR corpus:

名詞 (*noun*)、動詞 (*verb*)、形容詞 (*adjective*)、形容動詞 (*adjective*)、
 連体詞 (*adnominal*)、副詞 (*adverb*)、接続詞 (*conjunction*)、数字 (*num*)、
 感動詞 (*interjection*)、助詞 (*particle*)、助動詞 (*auxiliary*)、
 語尾 (*inflection*)、接頭辞 (*prefix*)、接尾辞 (*suffix*)、記号 (*symbol*)

Note that the category “動詞” is assigned to a verb-stem and conjugation part

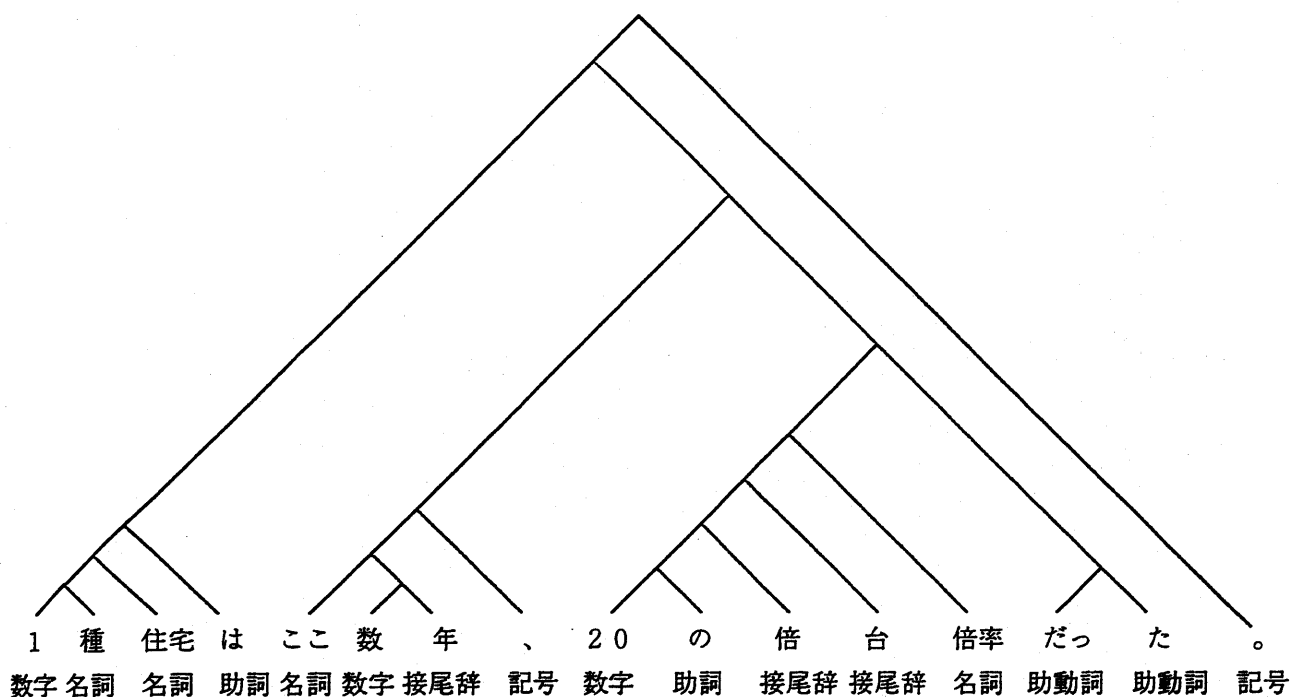


Figure 13. Example of EDR bracketed sentence.

adjectives and auxiliary verbs.

Generally speaking, elements of bracketed EDR sentences are finer than segments produced by our parser. We decide to correct dependency relation of segments in the following way:

1. If our segmentation of a segment and the segmentation of EDR corpus conflicts, the segment is assumed to modify the next segment.
2. Otherwise, the modifiee of segment is defined by the bracketing assigned to an element in EDR corpus that matches the last part of modifier string.

Figure 14 illustrates this process. The upper part of the Figure 14 indicates the structure of a bracketing in EDR, and the lower part of the Figure 14 indicates the dependency relation derived from the structure.

In Figure 14, you can see that the second element in EDR corpus matches the last element of the first segment. Suppose that the second element in EDR corpus modifies the sixth element in EDR corpus. Since the sixth element in EDR

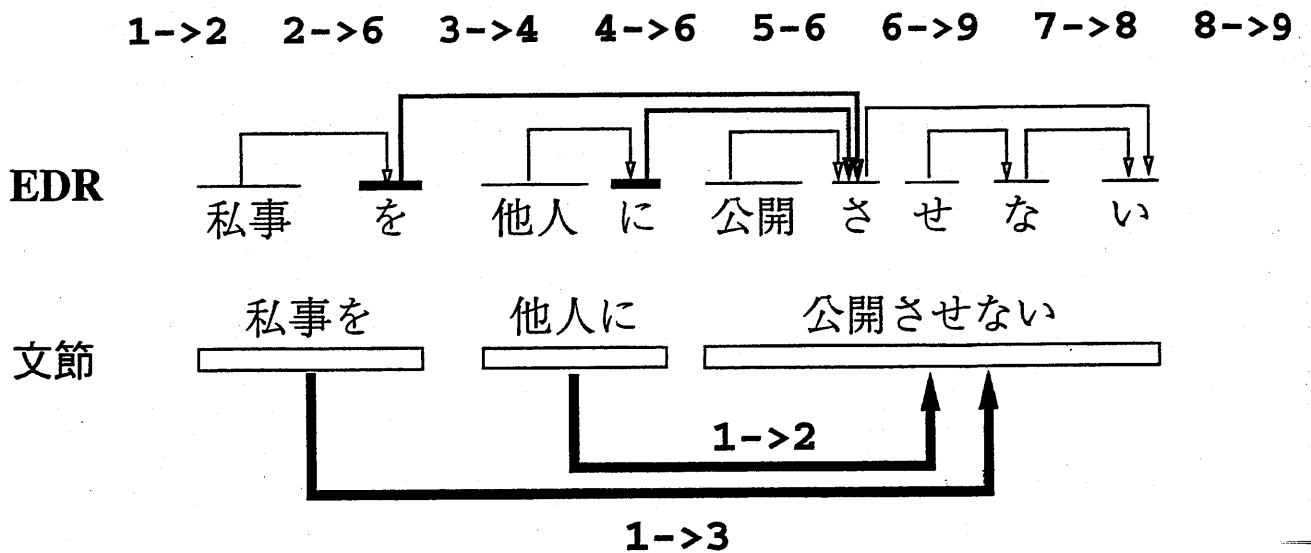


Figure 14. Extraction of a dependency relation from a bracketed EDR sentence.

corpus belongs to the third segment, we conclude that the first segment modifies the third segment.

Some EDR sentences have bracketing errors. Consider the example of Figure 13. The actual order of the morphemes “20 の倍台倍率だった” should be “20倍台の倍率だった”, which contains two segments “20倍台の” and “倍率だった”. But in the bracketed structure, the order of word “の” is permuted, thus the segment “20倍台の” modifies itself, which is a wrong dependency relation. We do not use these sentences for both training and evaluation.

We delete sentences that fail to segment correctly by our segment transducer (because of the error of the morphological analysis), or that has wrong dependency relation because of the errors in EDR corpus or crossing of two dependency relations. As the result, the number of the whole training and evaluation sentences becomes 206381.

3.6 Evaluation

We divide the bracketed sentences obtained so far into twenty files. One of these files is held out for evaluation and others are used for training.

3.6.1 Evaluation Measure

As the evaluation measures, we use the following segment level and sentence level precisions.

$$\text{Segment Level Precision} = \frac{\text{The Number of Segments whose Modifye Segment is Estimated Correctly}}{\text{The Number of Segments whose Modifye Segment is Estimated by the System}}$$

$$\text{Sentence Level Precision} = \frac{\text{The Number of Sentences in which Modifyees of all the Segments are Estimated Correctly}}{\text{The Number of Sentences in which Modifyees of all the Segments are Estimated by the System}}$$

3.6.2 Results of Segment Level Precision

Figure 9 shows the results under our model. The row in which *Distance Features* is “-” indicates the results without *Distance Probability* (in other words, *Distance Probability* is always “1”). Figure 10 shows the results under the Collins’ model explained in section 3.3.3. Both of the model are trained by about 19,000 sentences, and evaluated by held out 10,000 sentences.

BGH has a six-layered abstraction hierarchy, in which more than 80,000 words are allocated at the leaves. For the limitation of computer resources, we cannot use all the combination of word classes (the combination of a modifier and modifye). We test the combinations of same depth of layers from second to the six-*th*.

In either experiment, the *Head Collocation Model* using six-*th* layer of ‘Bunrui Goi Hyou’(BGH)[37] exhibited highest precision in the BGH model. Therefore “BGH” in the rest of this Chapter means the BGH model which uses the six-*th* layer of ‘Bunrui Goi Hyou’.

As can be seen from Tables 9, the model under LEX + dst2 shows the highest precision. But the difference between the result of the LEX model and the POS model is not so large.

Unexpectedly, in either model of LEX, POS, BGH, the precision under the model using dst1 is lower than the model that does not use *Distance Probability*. This may be because when using exact count of segments as the distance measure,

Table 9. Precision of correct dependency relations under our statistical model.

	Head Features			
Distance Features	POS	LEX	BGH	Base Line
-	0.7615	0.8001	0.7710	
dst1	0.6670	0.6836	0.6633	
dst2	0.8649	<u>0.8689</u>	0.8524	0.6237
dst3	0.8637	0.8675	0.8524	
dst4	0.8626	0.8674	0.8534	

Table 10. Precision of correct dependency relations. Under the model equivalent to Collins' model.

	Head Features		
Distance Features	POS	LEX	BGH
dst1	0.7954	0.8146	0.7666
dst2	0.8131	0.8189	0.8094
dst3	0.8021	0.8169	0.7775
dst4	0.7722	0.7987	0.7483

the problem of data sparseness becomes more apparent, and the plausibility of the estimated probability may decrease. In fact, the frequency of usage of *Distance Probability* estimated from the cases fewer than 10 was 4009 for the dst1, in contrast to the 398 for the dst2⁵.

For all the variations of distance measure we tested, the BGH model performed poorer than the POS model. This result was constant for changing the amount of training data. One reason will be that the hierarchy of Japanese thesaurus 'Bunrui Goi Hyou' does not reflect the head words' preferences for the dependency relation.

When evaluating Collins' Model, we tested smoothing methods such as linear interpolation used in Collins[6], and back-off method used in our model. Table 10

⁵ Evaluated with 10,000 sentences.

shows the results of the method that exhibited higher performance. Under the Collins' model, LEX + dst2 exhibited the highest performance too.

As can be seen from Tables 9 and 10, our model exhibited higher performance than Collins' model. In the training of dependency probabilities in Japanese, relation types⁶ are included in the conditional part of the probability estimation of a dependency relation, which makes the problem of data sparseness severer than the case of training in English. As the result, our model that separated *Head Collocation Probability* and *Distance Probability* shows higher precision in dependency analysis of Japanese sentences.

The following is the examples that have the modifier whose modifiee estimated correctly in the LEX model, but incorrectly in the POS model. The sign "○" shows correct modifiee of the modifier surrounded by a box, and the sign "×" shows the estimated modifiee by our model (POS model).

- ダライ・ラマは、ラサを 中心に○ 東チベットに 影響力を 持ってきた。×
(The Dalai Lama had influence on East Tibet, especially on Lhasa)
- 炉の 優れた× 冶金特性が○ 明かにされた。
(The outstanding character of the furnace was revealed.)

In the first example, the high collocation of the dependency relation "[名詞]を (noun+wo) → 中心に" does not override the low collocation of case particle "を (wo)" and "に (ni)".

In the second example, the dependency relation "[名詞]の 優れた" is a natural expression as you can see in the expressions such as "性能の 優れた (high performance)". But when you look closer to the lexical level, "炉の 優れた (??)" is unnatural, and the probability of the dependency relation "炉の → 特性 (the property of a furnace)" becomes higher than that of "炉の → 優れた"

On the contrary, the following is the example that has the modifier whose modifiee estimated correctly in POS model, but incorrectly in LEX model.

- 決ったことは 周知の 通りだが○ 新渡戸には 議論が あった。×

⁶ For the equivalence in Collins' model, see section 3.3.3

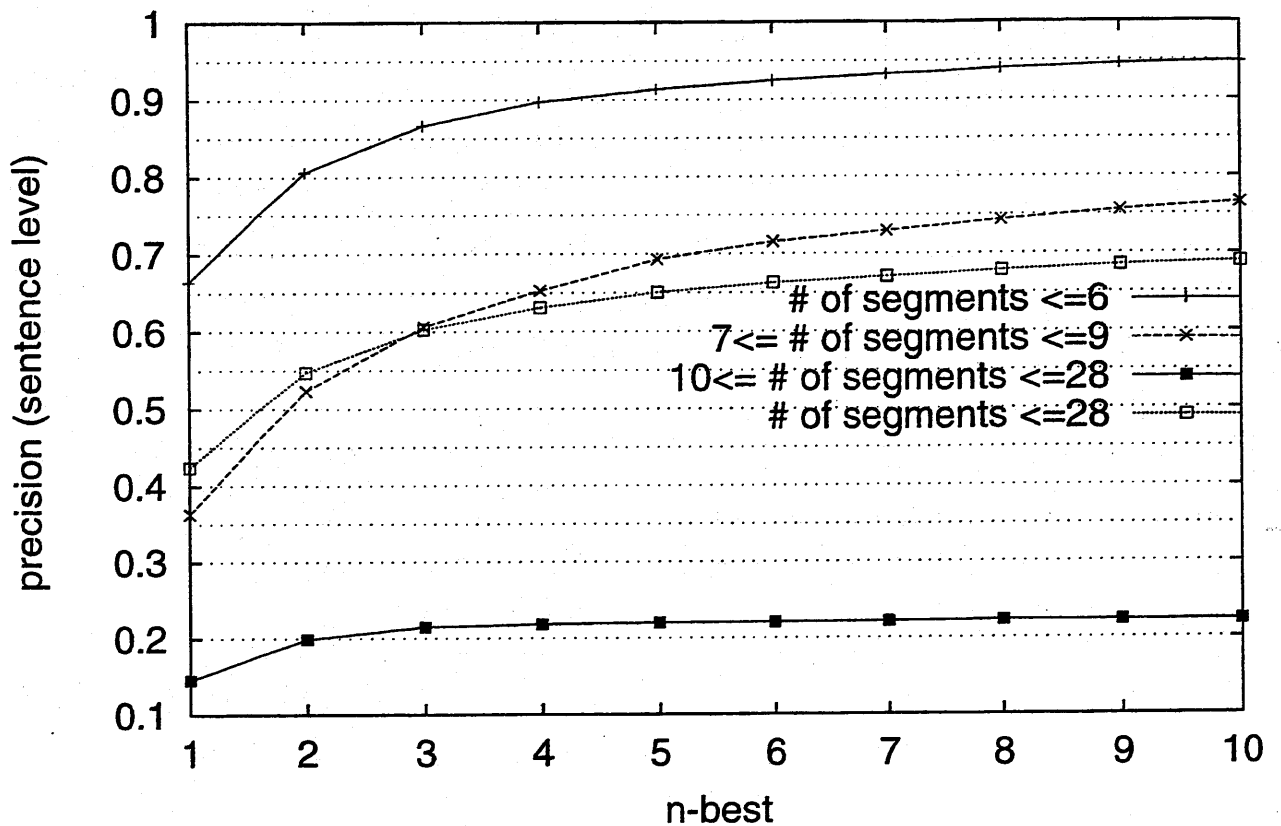


Figure 15. Precision of correct parses within n-best parses.

In this case, the frequency of the appearance of the word “ある” is very large, and a noun phrase or an equivalent to a noun phrase would modify “ある” with the high probability.

As can be seen in the above examples, there is a case that using lexical level information leads to wrong estimation. But in average, we can say that lexical level information allows the model more accurate estimation of the dependency probability, leading to higher precision under the LEX model in segment level.

3.6.3 Precision of Correct Dependency Structures

We evaluate the sentence level accuracy in this section. A sentence is regarded as correct if the totally correct dependency structure is found in the n -best parses of the parser, where n is a predetermined value.

For the evaluation of sentence level precision, we used the LEX model for *Head Collocation Probability* and *dst2* for *Distance Probability*. We trained the model with about 19,000 sentences, and evaluated the model with held out 10,000 sentences.

In Figure 15, the x-axis shows the value of n when calculating n -best parses, and the y-axis shows the value of sentence level precision. Figure 15 shows the results with distinct line graphs, according to the number of segments in a sentence. The maximum number of segments of the sentences in the evaluation data was 28.

The possible number of dependency structures changes in accordance with the number of segments in a sentence. For instance, a sentence with 6 segments has 42 possible dependency structures, a sentence with 6 to 9 segments has 163 to 1430 possible dependency structures⁷. The linear growing number of phrases in a sentence leads to the exponentially growing number of analysis, leading to exponentially growing degree of difficulties to estimate correct analysis of the whole dependency relations in a sentence.

Figure 15 shows that the sentence level precision for sentences with 7 to 9 segments accomplishes over 75%, by considering 10-best parses. In the case of shorter sentences, whose numbers of constituent phrases are lower than 6, the sentence level precision accomplishes nearly 95%, by considering 10-best parses.

The sentence level precision for the sentences whose numbers of constituent phrases are between 10 and 28 is poor. We can say that is the predictable result, considering its huge number of possible parses ($4,862 \sim 0.e \times 10^9$).

3.6.4 Evaluation for Dependency Relations

We examine the precision for each relation type as shown in Table 11. The first column shows the relation types, which consists of a word, a tag, an conjugation form. The second column in Table 11 indicates precisions. The third column indicates the frequency of correct relation types. The right most column is the total number of dependency relations.

For the training and evaluation, we used the LEX model for *Head Collocation Probability* and *dst2* for *Distance Probability*. we trained the model with about

⁷ $k(n) = \sum_{i=2}^n k(i-1) * k(n-i+1); k(1) = k(2) = 1$

Table 11. Precision examined by dependency types.

relation type (word/tag/conjugation form)	precision	correct	total
/形容詞/連体 (/adjective/attributive)	0.9558	1039	1087
を/格助詞/(wo/case-particle/)	0.9415	7178	7624
の/名詞接続助詞/(no/noun-conjuncting-particle/)	0.9251	11210	12118
に/格助詞/(ni/case-particle/)	0.9197	5901	6416
する/動詞/基本 (suru/verb/sentence-final)	0.9047	503	556
形容詞/連用 (/adjective/conjunctive)	0.8923	978	1096
と/述語接続助詞/(to/verb-conjuncting-particle/)	0.8878	696	784
が/格助詞/(ga/case-particle/)	0.8856	5115	5776
/動詞/基本 (/verb/sentence-final)	0.8828	1379	1562
/動詞/タ形 (/verb/ta-kei)	0.8594	721	839
が/述語接続助詞/(ga/verb-conjuncting-particle/)	0.8529	580	680
と/格助詞/(to/case-particle/)	0.8450	1586	1877
も/副助詞/(mo/particle/)	0.8412	1706	2028
で/格助詞/(de/case-particle/)	0.8306	1015	1222
だ/判定詞/テ形 (da/declarative/te-kei)	0.8263	980	1186
は/副助詞/(ha/particle)	0.8037	6276	7809
/動詞/テ形 (/verb/te-kei)	0.8033	960	1195
/時相名詞/(/temporal-noun/)	0.7973	1192	1495
/普通名詞/(/common-noun/)	0.7599	1190	1566
/動詞/連用 (/verb/conjunctive)	0.7516	838	1115

19,000 sentences, and evaluated the model with held out 10,000 sentences.

The frequency of the segments whose relation types have “は” (topic-marking particle), 時相名詞 (temporal-noun), 普通名詞 (common-noun), 動詞 (verb) in conjunctive-form (verb modification form), are high, but the precisions of these relation types are low. We can safely say that those relation types have great influence on the performance of dependency analysis.

The temporal-noun (時相名詞) sometimes modifies verbs appearing far in distance, while time and date expressions such as 春 (spring), 朝 (morning), 3月 (March), sometimes constructs compound nouns with neighboring nouns. This is one of the reason that the precision of temporal-noun(時相名詞) is low.

A noun + punctuation pattern is also a problematic case, because it can be a constituent of a coordinate structure. While it behaves like an adverb (temporal noun and adverbial noun) or forms subordinate clauses (“場合,” (in case), “結果,” (as a result)).

The particle “ha(は)” and “verb/conjunctive” can construct subordinate clauses in Japanese, and in some cases, it is difficult even for humans to consistently determine its modifiee.

Almost all of the relation types whose precision are lower than 70% also have lower frequency in a training corpus⁸. These are 名詞性名詞助数辞 (nominal noun unit suffix), 名詞接続助詞 (noun conjunction particle), 副詞的名詞 (adverbial noun), 形容詞性述語接尾辞 (adjectival predicate suffix), 述語接続助詞 (predicate conjunctive particle), 動詞性接尾辞 (verbal suffix). These are all function words. In our current implementation, all function words are further ramified by lexical terms when defining relation types. It may be effective to define relation types, so as not to distinguish relation types for the function words whose frequency in the training corpus are low.

The coordinate structure and subordinate clause are major causes of ambiguity in long Japanese sentences. It is also difficult for humans to decide correct dependency relations for the sentences with many subordinate clauses or coordinate structures, because of the ambiguities of semantic level. For the cases of topicalization, some linguists say that these modifiers modify more than one

⁸ In these relation types, only the relation type “など/名詞性名詞接尾辞/” has more than 100 frequency. Many of them are lower than 10.

modifiees on its right.

One approach is to leave the difficult cases unspecified. This does not conflict the purpose of using the system for other application such as information retrieval, knowledge acquisition, and machine translation, because it is favorable to output reliable partial parses rather than output unreliable full parses. But to judge which dependency relation is reliable and which is unreliable is another problem. Section 4 discusses this problem.

4. Application of Dependency Probability

Sufficiently high performance of dependency analysis will be great help for applications such as acquiring linguistic knowledge, machine translation, information extraction and so on. The performance of dependency analysis we achieved in Chapter 3 is about 87% by the segment level precision, which is not sufficient for these applications.

In the rest of this section, we will propose extended usages of the probabilities that the basic statistical model (see section 3.3) assign to each dependency structures (relations).

Section 4.1 describes statistical partial parsing method based on the probabilities of dependency relations. In this section, we propose three measures that will be effective for judging the plausibility of each dependency relation, and shows these measures function properly for the statistical partial parsing.

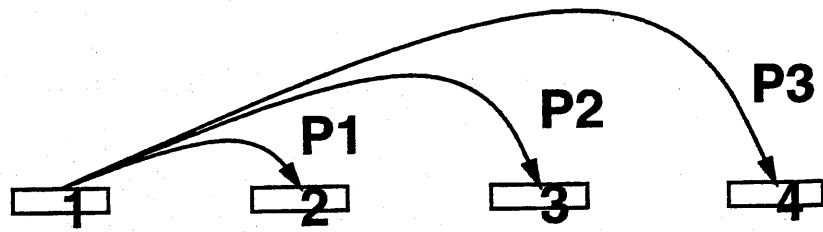
Section 4.2 describes statistical redundant parsing method based on the probabilities of dependency relations. In this section, we use three measures proposed in section 4.1, and shows some of those measures function properly for the statistical redundant parsing as well.

These ways are adaptable for the parser based on other statistical models that assumes independence between dependency relations.

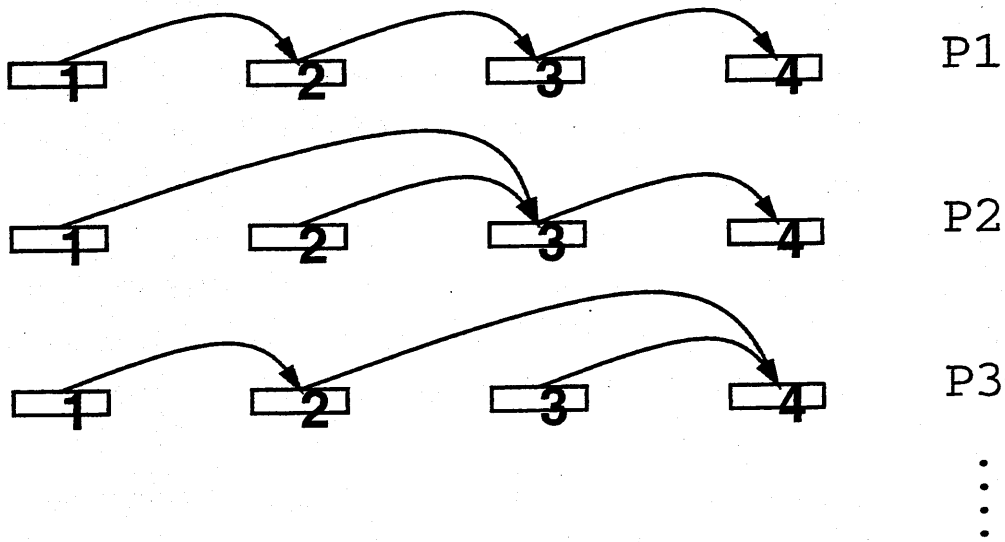
4.1 Partial Parsing

The partial parsing is the method to output only reliable dependency relations and achieve higher precision under an available statistical parser. In other words, partial parsing is the method that achieves higher precision at the cost of lower recall. To select plausible dependency relations, we need some measures of reliability.

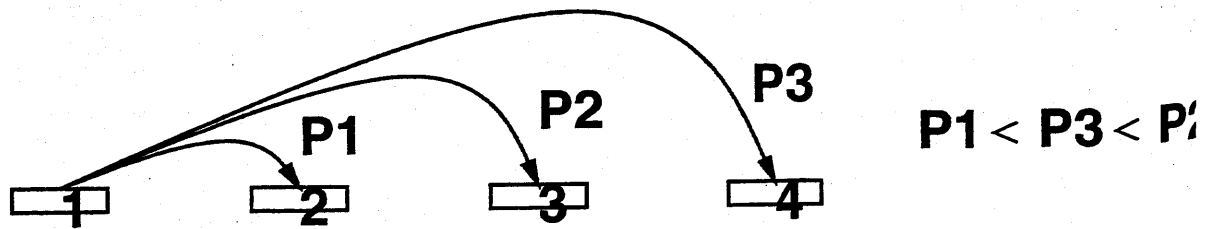
We propose the following three measures to select plausible dependency relations. These measures are also used in the **Redundant Parsing** in section 4.2.



Local/norm $1 \rightarrow 2 = \frac{P1}{P1 + P2 + P3}$



Global $1 \rightarrow 2 = \frac{P1 + P3}{P1 + P2 + P3 \dots}$



Ratio/next $1 \rightarrow 2 = \frac{P1}{P3}$

Figure 16. Measures to select plausible dependency relations

- Global** After calculating n -best parses, sum up the probabilities of all dependency relation that appear in the n -best parses. Then normalize the total sum of the probabilities of each dependency relation by the total sum of the probabilities of n parses.
- Ratio/next** Sort candidate modifies of each segment according to the probability of each dependency relation, then calculate the ratio of the probability of the first candidate to that of the second candidate.
- Local/norm** Divide the probability of each dependency relation by the total sum of the probabilities of candidate dependency relation.

In **Global**⁹, we set the value of n to 50. In **Ratio/next**, we consider that the ratio between dependency probabilities can be distinctive factor for judging plausibility of each dependency relation. In **Local/norm**, we consider that the proportion of a probability to the total probabilities of candidate dependency relations would be a good indicator for judging plausibility than the probability of a dependency relation as it is. Note that **Ratio/next** and **Local/norm** calculate only one-best parse, whereas **Global** needs to calculate 50-best parses.

4.1.1 Algorithm

By using the measures defined above, partial parsing is done by the following steps.

1. Calculate the measures defined above (In the rest of this dissertation, we call these measures as “confidence value”).
2. For each segment, choose the dependency relation that has the highest confidence value.
3. For each dependency relation chosen in the previous step, if the confidence value is larger than the pre-determined threshold, output the dependency relation.

⁹ For the detail, see [20]

4.1.2 Evaluation Measure

The number of estimated dependency relations by using partial parsing method changes in accordance with the threshold in step 3. We evaluate the performance of partial parsing method by the following segment level precision and coverage.

$$\text{Segment Level Precision} = \frac{\text{The Number of Segments whose Modifye Segment is Estimated Correctly}}{\text{The Number of Segments whose Modifye Segment is Estimated Uniquely}}$$

$$\text{Segment Level Coverage} = \frac{\text{The Number of Segments whose Modifye Segment is Estimated Uniquely}}{\text{The Total Number of Dependency Relations}}$$

4.1.3 Results

We used the LEX model for *Head Collocation Probability* and *dst2* for *Distance Probability*. Figure 17 shows the relationship between precision and coverage when changing the threshold used in the step 3 in section 4.1.1. The *x*-axis shows coverage and *y*-axis shows segment level precision.

Figure 17 shows that all three measures function properly for statistical partial parsing. In other words, by changing the threshold value, we can achieve desirable degrees of precision.

Comparing three measures defined above, **Local/norm** and **Ratio/next** behaves almost equally. **Global** exhibits a little lower precision in the range of 0.4 to 0.9 of the coverage, but a little higher precision in other range coverage. And, when using **Global**, coverage does not become lower than 0.28, which means 28% of segments have unique modifyees in 50-best parses, thus confidence value takes the maximum value "1". As a result, these relations are always chosen as a plausible dependency relations by the partial parser.

In terms of analysis speed, **Local/norm** and **Ratio/next** only need to calculate the best analysis, whereas **Global** needs to calculate 50-best parses, and results in slower partial parsing speed. We also test **Global** with 100-best parses, but the precision does not improve so much (about 0.5 % at most), although the analysis speed becomes much slower.

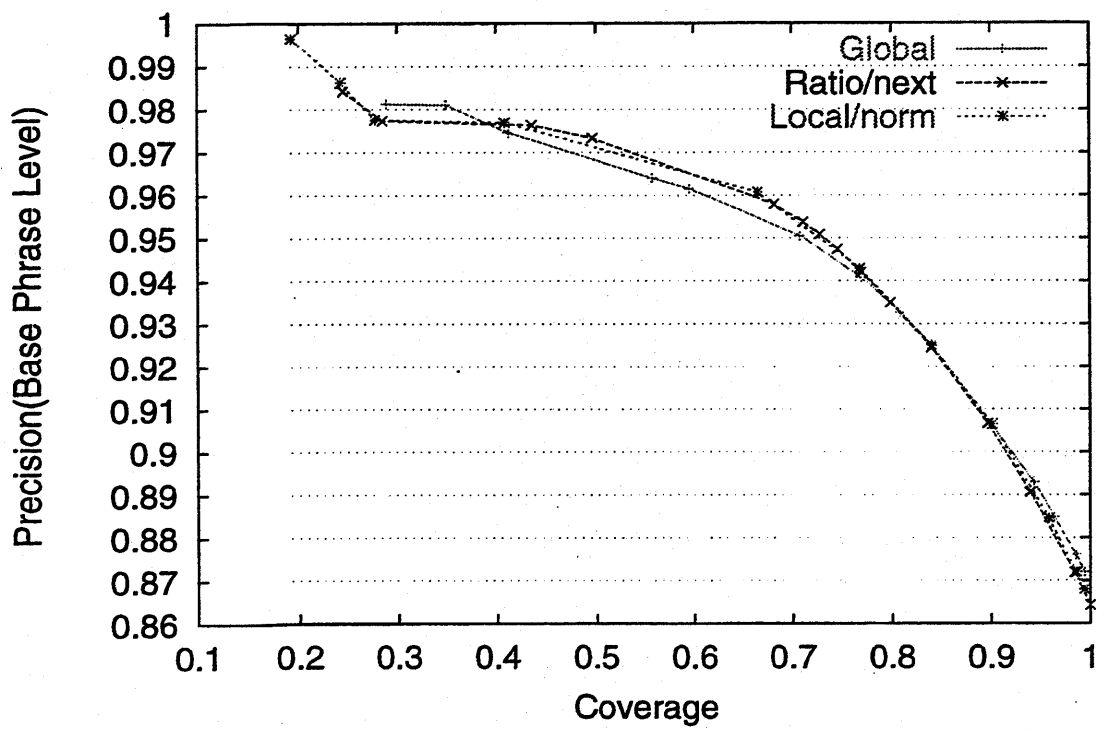


Figure 17. Relationship between precision and coverage.

In conclusion, at least under our statistical model, partial parser should use **Local/norm** or **Ratio/next** than **Global**.

4.2 Redundant Parsing

The redundant parsing is the method to output all equally plausible dependency relations of the same modifier, and achieve higher recall under an available statistical parser. In other words, redundant parsing is the method that achieves higher recall at the cost of lower precision. To select equally plausible dependency relations, we need some measures of reliability.

The same measures defined in section 4.1 can be used for Redundant Parsing.

4.2.1 Algorithm

1. Calculate confidence values (defined in section 4.1) for all the dependency relations in a sentence.
2. Output all the dependency relations in the best parse of a sentence.
3. Output all the dependency relations whose confidence values are higher than the pre-determined threshold.

4.2.2 Evaluation Measure

The number of over-generated dependency relations by using redundant parsing changes in accordance with the threshold in step 3. We evaluate the performance of redundant parsing by the following segment level recall and redundancy rate.

$$\text{Segment Level Recall} = \frac{\text{The Number of Segments whose Modifiee Segment is Estimated Correctly}}{\text{The Total Number of Dependency Relations}}$$

$$\text{Segment Level Redundancy Rate} = \frac{\text{The Number of Dependency Relations that System Outputs}}{\text{The Total Number of Dependency Relations}}$$

4.2.3 Results

We use LEX model for *Head Collocation Probability* and *dst2* for *Distance Probability*. We use the plausibility measure defined in Chapter 4.1.

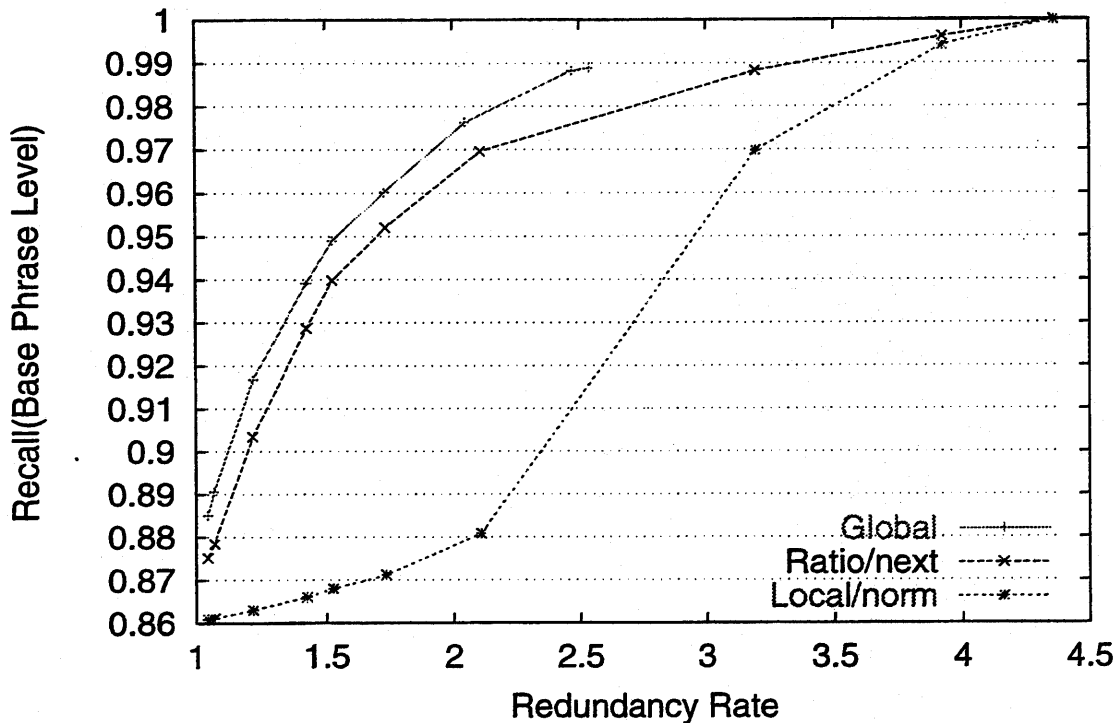


Figure 18. Relationship between recall and redundancy rate.

Figure 18 shows the relationship between recall and redundancy rate when changing the threshold used in the step 3 in section 4.2.1. The x -axis shows redundancy rate and the y -axis shows segment level recall.

It is seen that **Global** and **Ratio/next** function properly for statistical redundant parsing. In other words, by changing the threshold value, you can achieve desirable degrees of recall. On the contrary to the Figure 17, Figure 18 shows that **Local/norm** exhibits extremely poorer performance. **Global** achieves about 1 % higher recall than **Ratio/next**. But when using **Global**, recall does not become higher than 99%. This is because how lower the threshold value is set, it cannot achieve 100% recall, if n -best parses do not include whole correct dependency relations. Because the linear growing number of segments leads to the

exponentially growing number of possible analysis (see footnote in section 3.6.3), using 100-best instead of 50-best in **Global** does not change so much the upper limit of redundancy rate. In fact, comparing the redundancy rate between using 50-best parses and using 100-best parses in **Global**, the difference is only 0.6%.

In conclusion, at least in our model, **Ration/next** performs best for the purpose of partial or redundant parsing. Besides, if speed is not so important and the accuracy does not need be more than 99%, we should use **Global**.

5. Statistical Model for Subordinate Clauses

In dependency analysis of a Japanese sentence, among various source of ambiguities in a sentence, dependency ambiguity of subordinate clauses is one of the most problematic ones. In general, dependency ambiguities of subordinate clauses cause scope ambiguities of subordinate clauses, which result in enormous number of syntactic ambiguities of other types of phrases such as noun phrases.

In this Chapter, we propose a statistical model of subordinate clauses which takes into account the scope embedding preference of subordinate clauses manually analyzed by Minami [34], Minami [35] and Shirai [44] (see Chapter 2.4).

The most important point of the statistical model of subordinate clauses is that the model considers the “beyond” relation between two segments. Most of all statistical parsing model including our basic statistical model defined in Chapter 3 consider the “modify” or “not modify” relations, but the model described here considers “modify” or “beyond” relations.

The model learns the relation of scopes of two subordinate clauses (which one of two scopes is wider or narrower) from a corpus, as a tendency of “beyond” or “modify”, based on the combination of modifier’s features and modifiee’s features.

The other point is, we select effective set of features automatically by a statistical method. we employ the decision list learning method of Yarowsky [52], where optimal combination of features are selected and sorted in the form of decision rules, according to the strength of co-relation between features and the dependency preference of the two subordinate clauses.

5.1 Definition of Subordinate Clauses

This Chapter gives the definition of what we call a “Japanese *subordinate clause*” throughout this paper. First, as we described in section 3.4, an input sentence is segmented into a sequence of segments. Then, a *clause* in a sentence is represented as a sequence of segments.

Since the Japanese language is a head-final language, the clause head is the final segment in the sequence. Each segment generally consists of a set of content words and function words. For the detailed grammatical definition of a Japanese *subordinate clause*, see Appendix.

5.2 Learning Dependency Preference

In this Chapter, dependency preference between two segments means a tendency whether anterior segment *modifies* or *beyond* posterior segment. This relation has a strong relation to scope embedding preference (see section 2.4) of subordinate clauses.

Roughly speaking, the relation between scope embedding preference of subordinate clauses and dependency preference is as follows¹⁰

Dependency Preference of Japanese Subordinate Clauses

1. A subordinate clause can modify another subordinate clause which inherently has a scope of the same or a broader breadth.
2. A subordinate clause cannot modify another subordinate clause which inherently has a narrower scope.

5.2.1 Decision List Learning

A decision list [40, 52] is a sorted list of the decision rules each of which decides the value of a *decision* D given some *evidence* E . Each decision rule in a decision list is sorted in descending order with respect to some preference value, and rules with higher preference values are applied first when applying the decision list to some new test data.

First, let the random variable D representing a decision varies over several possible values, and the random variable E representing some evidence varies over '1' and '0' (where '1' denotes the presence of the corresponding piece of evidence, '0' its absence). Then, in the decision list learning method of [52], given some training data in which the correct value of the decision D is annotated to each instance, the conditional probabilities $P(D=x | E=1)$ of observing the decision $D=x$ under the condition of the presence of the evidence E ($E=1$) are calculated and the decision list is constructed by the following procedure.

1. For each piece of evidence, calculate the *likelihood ratio* of the largest conditional probability of the decision $D=x_1$ (given the presence of that piece

¹⁰ For a detailed discussion of the relation between scope embedding preference and modify/beyond relation of subordinate clauses, see Utsuro et al. [51].

of evidence) to the second largest conditional probability of the decision $D = x_2$:

$$\log_2 \frac{P(D = x_1 | E = 1)}{P(D = x_2 | E = 1)}$$

Then, a decision list is constructed with pieces of evidence sorted in descending order with respect to their likelihood ratios, where the decision of the rule at each line is $D = x_1$ with the largest conditional probability.¹¹

2. The final line of a decision list is always defined as ‘a default’, where the likelihood ratio is calculated as the ratio of the largest marginal probability of the decision $D = x_1$ to the second largest marginal probability of the decision $D = x_2$:

$$\log_2 \frac{P(D = x_1)}{P(D = x_2)}$$

The ‘default’ decision of this final line is $D = x_1$ with the largest marginal probability.

5.2.2 Model for Dependency Preference of Subordinate Clauses

In the model of dependency preference of subordinate clauses, *decision* D and *evidence* E is defined as follows

The decision D : represents the dependency relation of two subordinate clauses in a sentence and varies over the two values “modify”, where the anterior head segment modifies the posterior head segment, and “beyond”, where the anterior head segment modifies the head segment of another subordinate clause which follows the posterior head segment.

The evidence E : represents a pair (F_1, F_2) , where as F_1 and F_2 , we consider every possible subset of the feature sets \mathcal{F}_1 and \mathcal{F}_2 which Seg_1 and Seg_2 (the head segments of the given two subordinate clauses) have, respectively.

Table 12 gives features of subordinate clauses used for decision list learning.

¹¹ [52] discusses several techniques for avoiding the problems which arise when an observed count is 0. Among those techniques, we employ the simplest one, i.e., adding a small constant α ($0.1 \leq \alpha \leq 0.25$) to the numerator and denominator. With this modification, more frequent evidence is preferred when there exist several evidences for each of which the conditional probability $P(D = x | E = 1)$ equals to 1.

All the subset of features of modifier and modifiee segments can be sorted in descending order with respect to the preference value defined in previous section, and combination of features with higher preference values are applied first when applying the decision list to some new test data. In other words, effective sets of features among all possible sets of features are selected based on the likelihood ratios between the probability of “beyond” or “modify” relation of two subordinate clauses. Next section describes the way to use learned decision lists to syntactic disambiguation of subordinate clauses in a Japanese sentence.

5.3 Calculation of Preference Value for Subordinate Clause Dependencies

Next, we describes how to analyze dependencies of subordinate clauses in one sentence according to the probabilities of the dependencies between two subordinate clauses which are estimated from the decision list for dependency preference of subordinate clauses.

Suppose that an input sentence S contains $n - 1$ subordinate clauses and let $Seg_1, \dots, Seg_{n-1}, Seg_n$ be the head segments of those $n - 1$ subordinate clauses plus the sentence-final segment. Let us denote the sequence of those segments as S_{sb} :

$$S_{sb} = Seg_1, \dots, Seg_{n-1}, Seg_n$$

Next, let $mod(Seg_i)$ be the segment which Seg_i modifies. For simplicity, we assume that the modifiee segment $mod(Seg_i)$ of Seg_i is one of the subsequent segments $Seg_{i+1}, \dots, Seg_{n-1}, Seg_n$.

Then, we consider all the dependencies $mod(Seg_1), \dots, mod(Seg_{n-1})$ among the sequence $Seg_1, \dots, Seg_{n-1}, Seg_n$ of the head segments of subordinate clauses in the sentence S , and denote them as $Dep(S_{sb})$:

$$Dep(S_{sb}) = mod(Seg_1), \dots, mod(Seg_{n-1})$$

Here, we assume that no modification crosses to other modifications in a sentence.

In order to calculate the preference value of the dependencies $Dep(S_{sb})$, first, we estimate the probability $P(D = x \mid (Seg_i, Seg_j))$ of the decision $D = x$ given

Feature Type	# of Features	Each Binary Features
Punctuation	2	with-comma, without-comma
Grammatical	17	adverb, adverbial-noun, formal-noun, temporal-noun, quoting-particle, predicate-conjunctive-particle, topic-marking-particle, copula, sentence-final-particle (some of these features have distinction of segment-final/middle)
Conjugation Form of Segment-final Conjugative Word	12	stem, base, mizen, ren'you, rentai, conditional, imperative, <i>ta</i> , <i>tari</i> , <i>te</i> , conjecture, volitional
Lexical (lexicalized forms of 'grammatical' features, with more than 9 occurrences in EDR corpus)	235	adverb (e.g., <i>ippou-de</i> , <i>irai</i>) adverbial-noun (e.g., <i>tame</i> , <i>baai</i> , <i>you</i> , <i>hou-ga</i>) formal-noun (e.g., <i>no-ha</i> , <i>koto</i> , <i>koto-ga</i>) temporal-noun (e.g., <i>ima</i> , <i>shunkan</i> , <i>mae-ni</i>), quoting-particle (<i>to</i>), predicate-conjunctive-particle (e.g., <i>ga</i> , <i>kara</i> , <i>nagara</i>), topic-marking-particle (e.g., <i>ha</i> , <i>mo</i> , <i>dake</i> , <i>nara</i>), copula (e.g., <i>dearu</i>), sentence-final-particle (e.g., <i>ka</i> , <i>yo</i>)

Table 12. Features of Japanese subordinate clauses

a pair (Seg_i, Seg_j) as the maximum of the probabilities $P(D = x | (F_i, F_j))$ for every possible pair of (F_i, F_j) and denote it as $\hat{P}(D = x | (Seg_i, Seg_j))$:

$$\hat{P}(D = x | (Seg_i, Seg_j)) = \max_{(F_i, F_j)} P(D = x | (F_i, F_j))$$

Then, we introduce the notion *preference value* of the dependency of Seg_i 's modifying Seg_k and denote it as $Q(D = \text{"modify"} | (Seg_i, Seg_k))$. The preference value $Q(D = \text{"modify"} | (Seg_i, Seg_k))$ is calculated by the procedure below:

1. In the case where $k < n$ holds:

$Q(D = \text{"modify"} | (Seg_i, Seg_k))$ is calculated as the geometric mean of the probability of Seg_i 's modifying Seg_k and those of Seg_i 's modification being beyond Seg_j ($j = i + 1, \dots, k - 1$).¹²

$$Q(D = \text{"modify"} | (Seg_i, Seg_k)) = \left(\hat{P}(D = \text{"modify"} | (Seg_i, Seg_k)) \times \prod_{j=i+1}^{k-1} \hat{P}(D = \text{"beyond"} | (Seg_i, Seg_j)) \right)^{\frac{1}{k}}$$

2. In the case where $k = n$ holds:

Since the segment Seg_n is sentence-final, we can assume $\hat{P}(D = \text{"beyond"} | (Seg_i, Seg_n)) = 0$ and $\hat{P}(D = \text{"modify"} | (Seg_i, Seg_n)) = 1$ for $i = 1, \dots, n - 1$. Then, $Q(D = \text{"modify"} | (Seg_i, Seg_n))$ is calculated as the geometric mean of the probabilities of Seg_i 's modification being beyond Seg_j ($j = i + 1, \dots, n - 1$).

$$Q(D = \text{"modify"} | (Seg_i, Seg_k)) = \left(\prod_{j=i+1}^{n-1} \hat{P}(D = \text{"beyond"} | (Seg_i, Seg_j)) \right)^{\frac{1}{n-i}}$$

Finally, given the sequence S_{sb} of the head segments of subordinate clauses and their dependencies $Dep(S_{sb})$, its preference value $Q(S_{sb}, Dep(S_{sb}))$ is calculated as the product of the preference value of each dependency:

$$Q(S_{sb}, Dep(S_{sb})) = \prod_{i=1}^{n-2} Q(D = \text{"modify"} | (Seg_i, mod(Seg_i)))$$

¹² We calculate the preference value by the geometric mean rather than the product in order to make a fair comparison among the cases of different number of intermediate segments Seg_j 's.

Table 13. Statistics of test sentences extracted from EDR corpus

	Subsets		Full Set
	Sentences Including More Than One Subordinate Clauses	Sentences Including One or Zero Subordinate Clause	
# of Sentences	3,128 (30.3%)	7,192 (69.7%)	10,320
# of Segments	32,038 (39.9%)	48,281 (60.1%)	80,319
Ave. # of Segments / Sentence			
# of Subordinate Clauses			
Dependency Analysis Precision of Basic DA Model (Chapter 3)			
Segment Level	85.3%	86.7%	86.1%
Sentence Level (Best One)	25.4%	47.5%	40.8%
Sentence Level (Best Five)	35.8%	60.2%	52.8%

Then, the dependency which gives the highest preference value is selected as the estimation $\hat{Dep}(S_{sb})$ of the dependencies among the sequence S_{sb} of the head segments of subordinate clauses.

$$\hat{Dep}(S_{sb}) = \underset{Dep(S_{sb})}{\operatorname{argmax}} Q(S_{sb}, Dep(S_{sb}))$$

5.4 Experiments and Evaluation

5.4.1 Test Data

As shown in Table 13, 5% test data set of the whole EDR corpus consists of about 10,000 sentences, and about 30% of them have more than one subordinate clauses per each sentence, i.e., having ambiguities of dependencies of subordinate clauses. The number of those ambiguous subordinate clauses is 4,207 in total.

As the test data sets, we use the following three subsets of 5% test data set

of whole EDR corpus.

1. "Whole Data Set"

This is the collection of 3,128 sentences in Table 13, each of which includes more than one subordinate clauses.

2. "Sentences with at least One Initial Dependencies"

This is a subset of the data set "Whole Data Set", where each member of it is a sentence which has at least one initial dependency with the fixed probability 1.

3. "Sentences with Complete Initial Dependencies"

This is a subset of the data set "Sentences with at least One Initial Dependencies", where each member of it is a sentence for which all the dependencies of its subordinate clauses have fixed probabilities 1.

5.4.2 Evaluation Method

We evaluate the estimated dependencies of subordinate clauses by integrating them into basic statistical dependency analyzer of a whole sentence described in Chapter 3. First, we estimate the dependencies of subordinate clauses in a sentence by the procedure of Chapter 5.3, then, regard them as correct dependencies when parsing under the basic dependency analysis model¹³ in Chapter 3. More specifically, we assign the probability 1 to the dependencies of subordinate clauses estimated by the procedure of section 5.3, and fix this probability during the statistical dependency analysis of the whole sentence.

5.4.3 Results

We change the threshold of the probability $P(D | E)$ in the decision list and plot the trade-off between threshold and precision 3.6.1. When we use the learned decision list, we ignore decision rules whose probabilities (conditional probabilities for the decision D of each decision rule) are lower than the threshold. Figures 19

¹³ The basic statistical dependency analyzer in Chapter 3 is trained using the same training data set as used in the decision list learning of section 5.2.1.

and 20 show the results of the segment level and sentence level precisions for the three test data sets as well as the upper and lower bounds of the precisions.

The upper bounds of the segment level and sentence level precisions are estimated by using basic dependency analysis model in Chapter 3 with correct dependencies of subordinate clauses extracted from the bracketing of EDR corpus.

The lower bounds of the segment level and sentence level precisions are obtained by parsing under the basic statistical model in Chapter 3 with no initial dependencies of subordinate clauses.

The result of evaluating segment level precision in Figure 19 shows that the precision for "Whole Data Set" is maximized when the threshold is 0.8, and its maximum is greater than the lower bound by about 1.7%, even though the segment level coverage of estimating dependencies of subordinate clauses is about 80%. This result means that the partially estimated dependencies of subordinate clauses are still useful for improving the segment level precision of the dependency analysis of the whole sentence.

Next, the result of evaluating sentence level precision in Figure 20 shows that the partially estimated dependencies of subordinate clauses are also useful for improving the sentence level precision of the dependency analysis of the whole sentence. One surprising result is that the sentence level precision calculated using top five results almost reaches the upper bound when the threshold is below 0.7.

5.4.4 Comparison between Related Works

This section discusses the merit of our method described in this Chapter compared with the conventional statistical parser ([14, 17, 10, 51]). The main points of conventional statistical parser that differs from our method are the following three points.

- (1) Consider the cases where two segments are a dependency relation or not dependency relation, and estimate the probability of dependency relation by the following definition (or pursuant definition):

$$\frac{\text{Two Segments are a Dependency Relation}}{\text{The Frequency of Two Segments Appears in a same Sentence}}$$

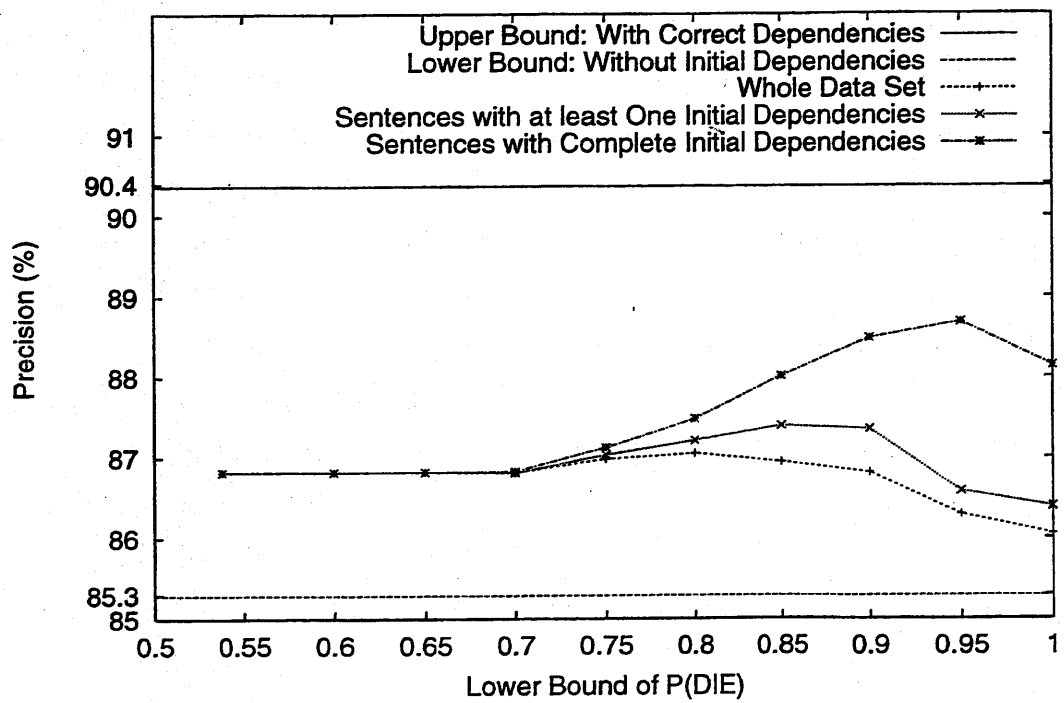


Figure 19. Changes of segment level precisions of dependencies of a whole sentence

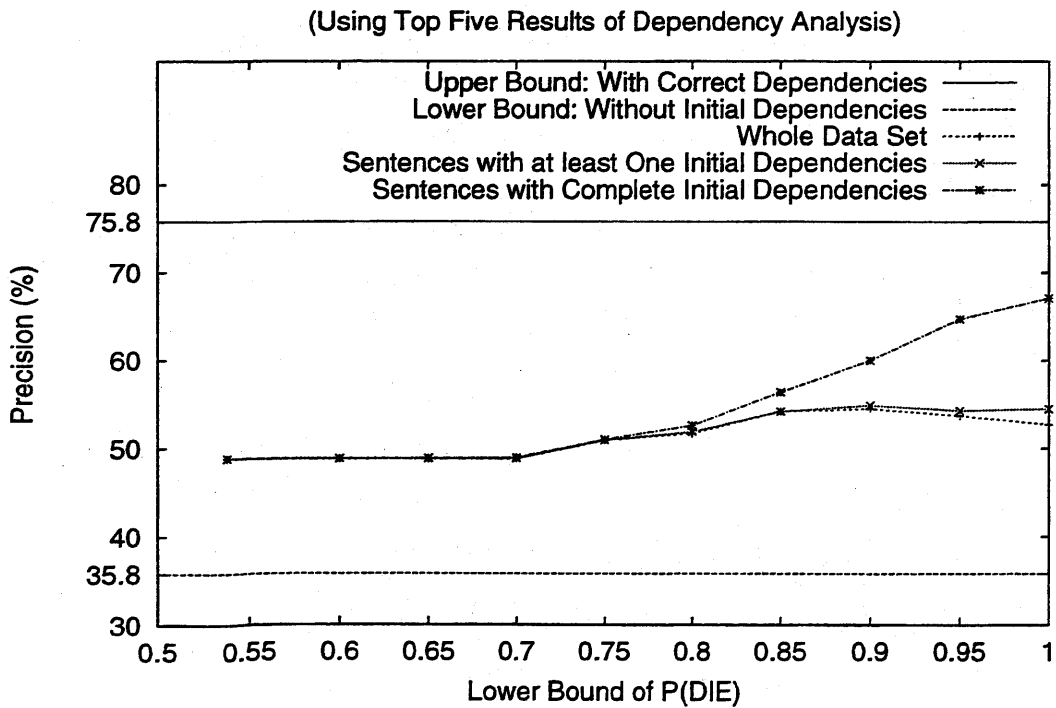
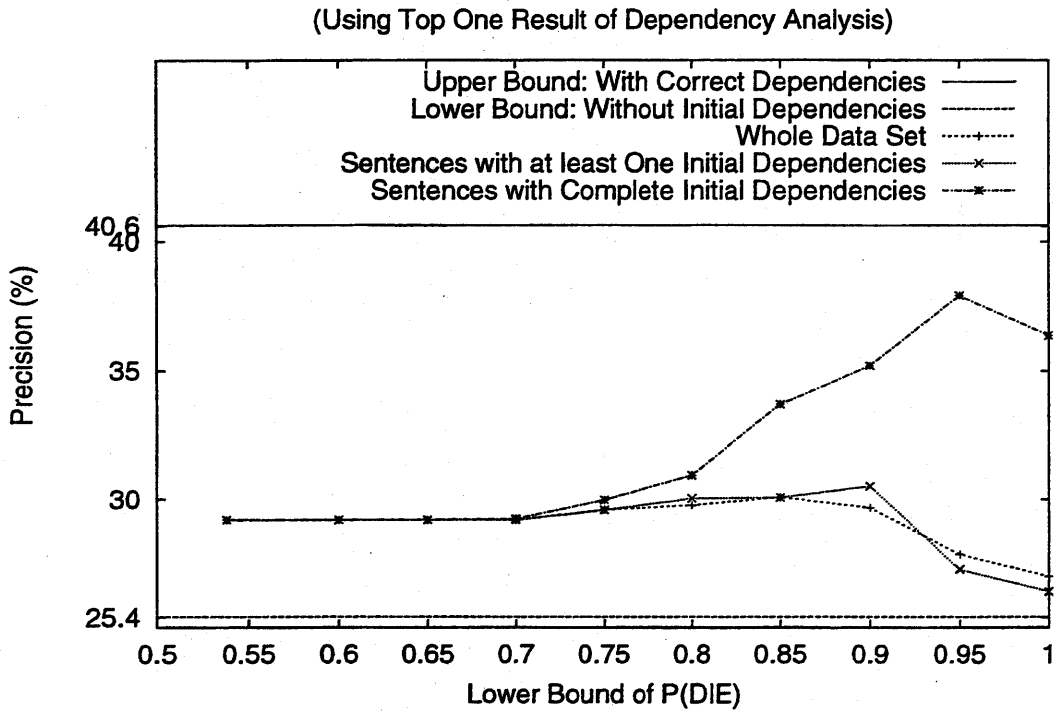


Figure 20. Changes of sentence level precisions of dependencies of a whole sentence

- (2) Define the probability of dependency analysis of the sentence as a product of the probabilities of all dependency relations in a dependency structure, and find the dependency structure that has highest probability.
- (3) Among the above conventional statistical model except Haruno [17], features used for the estimation of the probability of a dependency relation is fixed and there is no explicit way to select features automatically. And in Haruno [17], they use decision tree learning [38] method, not decision list learning.

To compare our method with the statistical model that satisfies above (1) and (2) conditions, we consider the model that is added the following changes to the decision list model defined in section 5.2.2 and the parsing framework of subordinate clauses in a section 5.3.

- Add “end-of-sentence” features to the decision list model (conventional statistical models use this feature).
- Consider two value “modify” where the anterior head segment modifies the posterior head segment, and “not modify” where the anterior head segment modifies the head segment of another subordinate clause which follows the posterior head segment, as the *decision* D .
- As a *preference value* of the dependency $Q(D = \text{“modify”} \mid (Seg_i, Seg_k))$ of Seg_i 's modifying Seg_k , use the following equation: $Q(D = \text{“modify”} \mid (Seg_i, Seg_k)) = P(D = \text{“modify”} \mid (Seg_i, Seg_k))$

For this model, decision list model was learned by the training data extracted from 95% of EDR bracketed corpus. In the resulting decision list, $D = \text{“not modify”}$ becomes default rules, and the probability of this default rule is $P(D = \text{“modify”}) = 0.6180$. By using this decision list, we evaluated the precision of dependency analysis.

The resulting coverage of both segment level and sentence level were very low compared with our model, for the same threshold $P(D \mid E)$. From this result, we can say that our model outperforms the conventional statistical models which considers “modify” or “not modify” relation and only uses the probability of “modify”.

Next, we compare our method with Haruno [17] which uses decision tree model to automatically select effective features.

In decision tree learning, features that decrease an entropy on target class most are selected, and a training set is divided to subsets according to the value of the feature. In Haruno's [17] study, each step of decision tree construction, only one of the feature of modifier and modifiee is examined, in other words, a set of features of a modifier and modifiee that become effective for the first time when combined may be overlooked. To examine the effect of the combination of modifier's features and modifiee's features to the parsing accuracy, we applied decision tree method (Quinlan [38]) to learn dependency preference of subordinate clauses.

We configured the set of features based on the features used in Haruno [17]. The target class is "modify" and "beyond", as is the case of decision list learning in section 5.2.2.

The resulting precision of the parsing of a whole sentence under decision tree model was lower than that of decision list model for almost all the value of coverage (defined by the threshold to the lower limit of probabilities at the leaf nodes of a decision tree). The decision list model outperformed about 2~3% in terms of segment level precision. This result suggests that in decision list learning, the combination of modifier's features and modifiee's features contributes to the accuracy of the parsing of a whole sentence.

5.5 Conclusion

In this section, scope embedding preference of subordinate clauses is exploited as a useful information source for disambiguating dependencies between subordinate clauses.

We successfully increased the accuracies of both segment level and sentence level dependencies thanks to the estimated dependencies of subordinate clauses.

In this Chapter, the model first (partially) disambiguated dependency relations of subordinate clauses, then performed the dependency analysis of a whole sentence. But this is not necessarily the best way to integrate dependency preference of subordinate clauses with our statistical model (Chapter 3). We need to study the best way to incorporate dependency preference of subordinate clauses

into our statistical model.

Note that the segmentation of subordinate clauses in this Chapter is not always the same as the segmentation used in Chapter 3. When considering to construct a uniform statistical parser, we need the definition of the segment that fits to the analysis of subordinate clauses.

Instead of using geometric mean of the probability of Seg_i 's modifying Seg_k and those of Seg_i 's modification being beyond Seg_j ($j = i + 1, \dots, k - 1$) as the *preference value* $Q(D = \text{"modify"} \mid (Seg_i, Seg_k))$, the value of product exhibited higher precision when the lower limit of the probability $P(D \mid E)$ is high. One of the reasons is that the product put a bias to the shorter distance of dependency relations (because the number of probabilities multiplied is fewer). This result suggests that considering distance information on dependency relation in decision list learning would result in higher precision of dependency analysis of subordinate clauses in a sentence.

6. Statistical Model for Coordinate Structure

In Japanese sentences, coordinate structure is one of the major causes of syntactic ambiguity in long sentences, and leads to the difficulties in dependency analysis of Japanese sentences. The main problem is to identify the *scope* of constituents of coordinate structures.

A conventional rule based parser of Japanese sentences handles coordinate structures by hand-coded heuristics (Kurohashi[26],Suganuma [48]).

These approaches first identify the existence of coordinate structures by finding key words or patterns that give an evidence of the existence of a coordinate structure, and then identify the scope of the coordinate structure by examining the word sequence similarity or structural similarity. But these keys for coordinate structures or heuristics to find the scope of coordinate structures must be developed by a language expert. Constructing and maintaining these rules is a laborious task. Another disadvantage is that the processes of coordinate structure identification and dependency analysis are sequential. In other words, dependency analysis is performed based on the identified coordinate structures, then the failure of coordinate structure identification, directly influences the accuracy of dependency analysis.

In this Chapter, we do not resort to extra heuristics for finding the similarity between the constituents of a coordinate structure. We propose a method to handle the coordinate structure identification and the dependency analysis in a uniform way.

The final probability of dependency structure reflects the existence of the coordinate structures from two reasons. The first reason is that the existence of coordinate structures affects the distance features of dependent pairs. The second reason is that a coordinate structure has multiple head words, and the probability of dependency relation is calculated from the multiple probabilities defined by these multiple head words. In this dissertation, we use the harmonic mean of multiple dependency probabilities for such cases of multiple head words. This comes from the natural intuition that every element of coordinate structure have an equally high probability to modify the same modifiee or to be modified from the same modifier. The harmonic mean has a tendency to get more influence by the lower probability than the simple mean of probabilities.

6.1 Definition of Coordinate Structures

According to Shutou [46], there are three types of coordinate structures.

- (a) Predicate coordinate structure (constituents of a coordinate structure has sentence form)
- (b) Nominal coordinate structure (constituents of a coordinate structure are noun phrases)
- (c) Partial coordinate structure (constituents of a coordinate structure are a sequence of segments except the right most segment in a sentence form)

The followings are examples of those coordinate structures.

- (a) 15分には 明大のスクラムサイドの突進をつぶして、₁ 井沢が逆襲をかけ、₂ ラトウがロングキック。₃
In fifteen minutes, they broke the scam of Meiji University,₁ Izawa counter-atta
and Latu kicked long.₃
- (b) 当面、行政改革、₁ とりわけ規制緩和、₂ 特殊法人の見直し、₃ 地方分権など₄
大きな課題がある。
For a time, their are important problems such as political reform,₁ deregulation,₂
reformation of quasi-non-governmental organization,₃ and decentralization.₄
- (c) 浪花屋の鯛焼きは浴衣、₁ よそのはどてらを、₂ 着ている
Taiyaki of Naniwaya wears a yukata,₁ and others a padded kimono.₂

Among these coordinate structures, we deal with the coordinate structure of type (b) and (c). In this dissertation, the dependency relation of type (a) is treated as a dependency relation between clause level dependency.

6.2 How to Learn from Coordinate Structures

There are three important points in our framework of coordinate structure analysis.

- The model does not use any heuristics to calculate the similarity between the constituents of a coordinate structure.

- The model performs dependency analysis and coordinate structure analysis at the same time, and selects the most likely structure.
- The model learns the probabilities defined in Chapter 3 (*Head Collocation Probability* and *Distance Probability*). The method we explain in this Chapter is available for other statistical dependency analyzer that assumes the independency of dependency relations.

The existence of a coordinate structure influences the set of positive cases (features set of dependency relations) in training data, in other words, we have to reconstruct the training corpus so as to add new correct dependency pairs and change distance features used for the probability estimation of *Distance Probability* defined in Chapter 3.

Figures 21,22 and 23 give examples in which modifier or modifiee segments have coordinate structures.

For instance, in Figure 21, both segment “4” and segment “7” modify the segment 8. If the structure in Figure 21 has a flat structure, the distance of dependency relation between the segment “4” and the segment “8” is “4”, but if it constructs a coordinate structure, the distance becomes 1. At the same time, the model learns a new dependency relation (coordinate relation) between the segment “4” and segment “7”.

The coordinate structure also influences the features of dependency relations between the segments before the coordinate structure and the elements in the coordinate structure (Figure 22). The distance feature between the segment “0” and “7” is 3 in Figure 22, but 7 if the structure in Figure 22 has a flat structure.

In case both the modifier and modifiee have coordinate structures (in Figure 23)), every possible combination of the elements in the modifier coordinate structure and those in the modifiee coordinate structure becomes training data.

In the training phase, we have an option whether the model distinguishes the statistics of coordinate relations and normal dependency relations or not. In section 6.5, we evaluate the parsing accuracy for both of these cases.

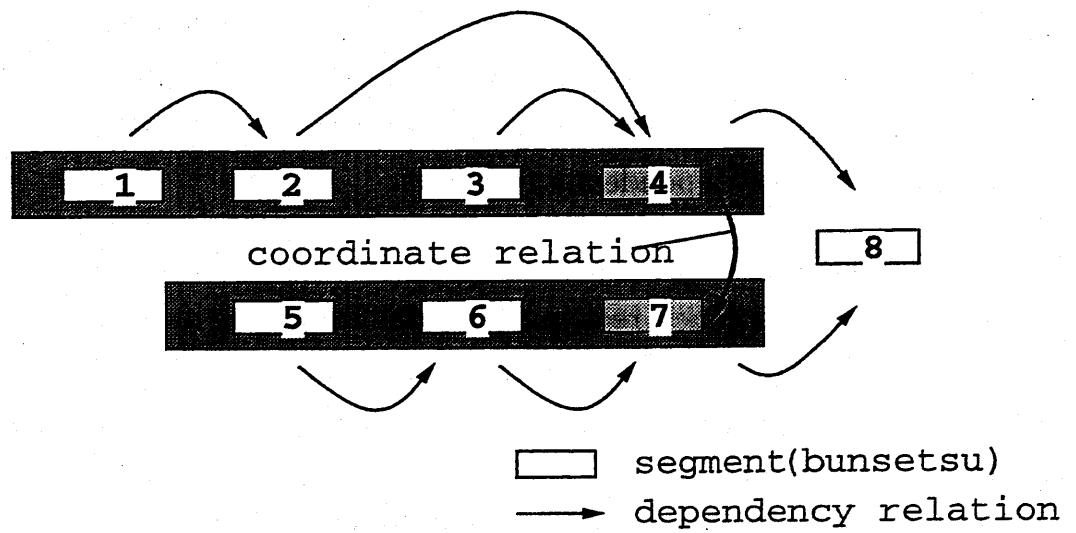


Figure 21. Modifier has a coordinate structure.

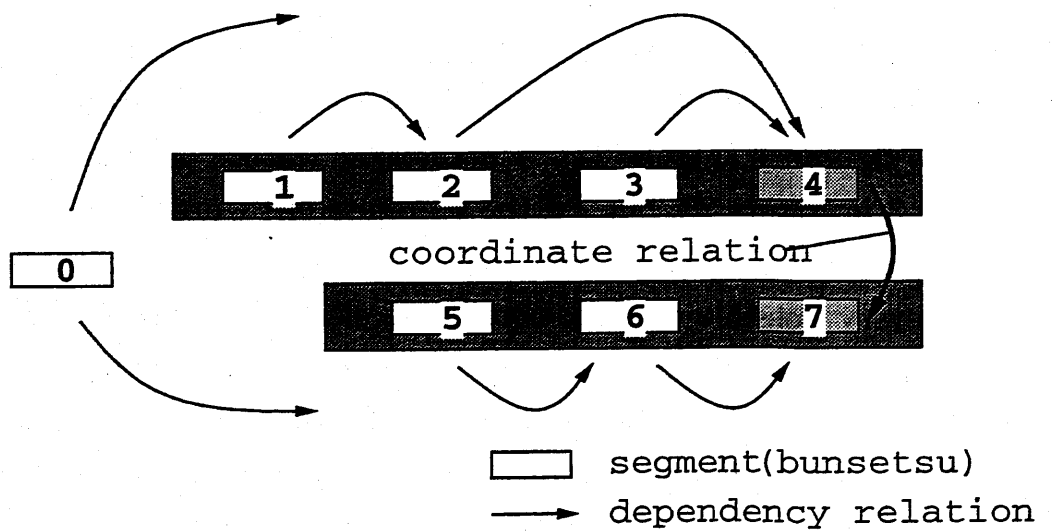


Figure 22. Modifiee has a coordinate structure.

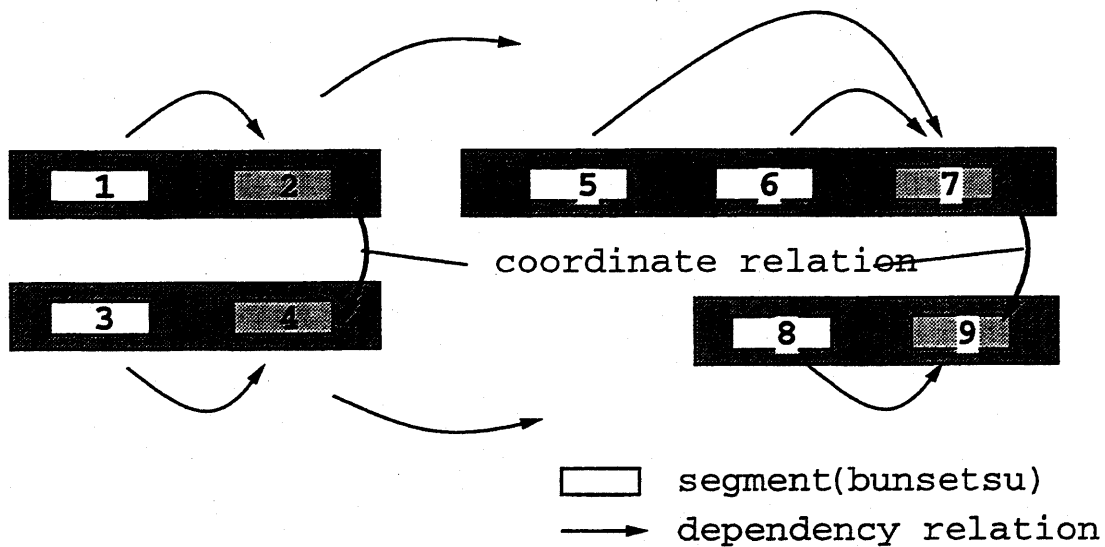


Figure 23. Both modifier and modifiee have coordinate structures.

6.3 Parsing Algorithm

We extend the statistical CKY algorithm described in section 3.4.4 to handle coordinate structures.

To construct a parse tree for a particular range of a sequence of segments, CKY algorithm combines two adjacent parse trees that cover the same range of the sequence of segments, where the left most segment of the anterior parse tree coincides with the left most segment of the target parse tree, and the right most segment of the posterior parse tree coincides with the right most segment. Figure 24 gives an example of this process. In Figure 24, the parser constructs a parse tree that covers the segments from "1" to "7", by considering two subtrees, the anterior one covers the segment from "1" to "4", and the posterior one covers the segment from "5" to "7". To combine these subtrees means to add a dependency relation from the head segment of anterior sub-tree to the head segment of posterior sub-tree. In this example, there can be 5 more possible pairs of sub-trees.

In addition to these normal structures, we consider the possibilities to construct coordinate structures. Figure 25 gives the case to construct a coordinate

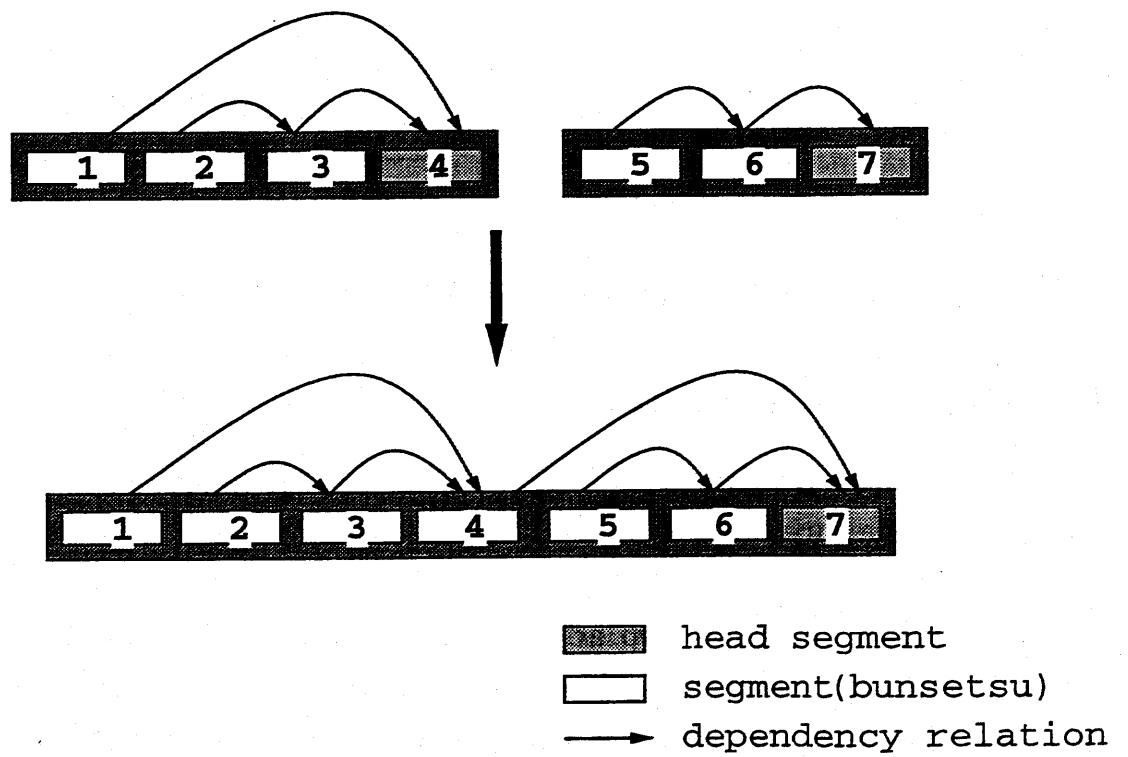


Figure 24. Normal construction of a new candidate partial parse in CKY algorithm.

structure from the same pair of sub-trees in Figure 24.

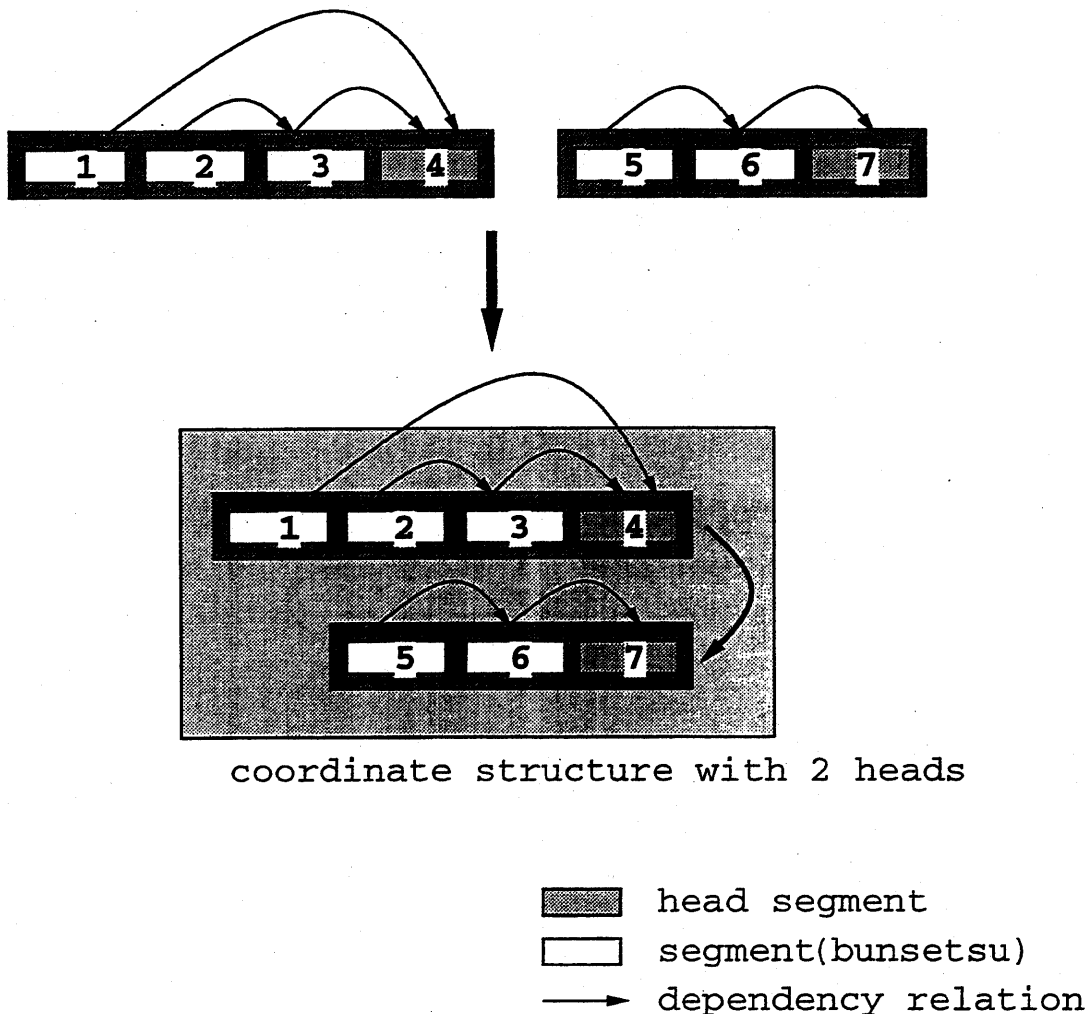
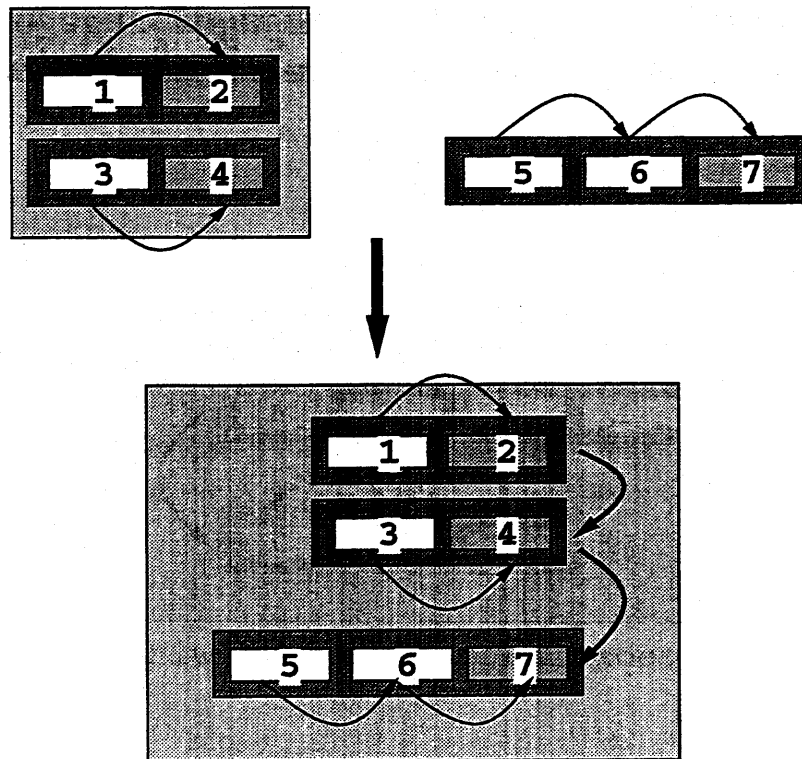


Figure 25. Coordinate structure construction in CKY algorithm.

Note that both parse trees in Figure 24 and Figure 25 have the same number of dependency relations, which means the product of the probabilities of constituent dependency relations can be used as a ranking measure to compare with the competing syntactic structures.

By simply adding the possibility of a coordinate structure at each step of CKY algorithm, we can achieve a uniform treatment of dependency analysis and coordinate structure identification. Note that this procedure enables us to treat



coordinate structure with 3 heads





-  head segment
-  segment(bunsetsu)
-  dependency relation
-  coordinate relation

Figure 26. Multiple coordinate structure construction in CKY algorithm.

the coordinate structures which consist of more than two elements without any other mechanism. The only thing to do for the coordinate structure is to keep multiple heads for later use. Figure 26 illustrates this situation.

6.3.1 Calculation of the Dependency Probability

The difference of a coordinate structure and a normal structure appears when calculating the probability of dependency relations whose modifier or modifiee is among the head segments of a coordinate structure (by changing *Head Collocation Probability* (see Chapter 3)) or when there are coordinate structures between modifier and modifiee (by changing *Distance Probability* (see Chapter 3)).

Figures 27, 28 and 29 give four possible cases, in which at least the modifier or the modifiee is an element of head segments in a coordinate structure. Figure 30 gives the case where the modifiee substructure contains a coordinate structure in it.

Figure 27 is the case that the root node of modifier structure has a coordinate structure. Figure 28 is the case that the root node of the modifiee structure has a coordinate structure. Figure 29 is the case that both of the root nodes of the modifier structure and modifiee structure have coordinate structures.

In these Figures, the left structure (subtree) correspond to the modifier substructure, and the right structure correspond to modifiee substructure. The arrows indicate the dependency relations whose probabilities must be calculated when constructing larger structures. The shaded rectangles indicate head segments in each substructure.

In the rest of this section, first, we explain the cases where the dependency probability is calculated from multiple dependency relations (Figure 27, 28 and 29). Then, we explain the case where the coordinate structures appear inside of the modifiee substructure, and has influence on the calculation of *Distance Probability* (Figure 30).

Calculation of the Dependency Probability from Multiple Heads

Figures 27, 28 and 29 illustrate the basic patterns where coordinate structures have influence on the calculation of dependency probabilities. we calculate the

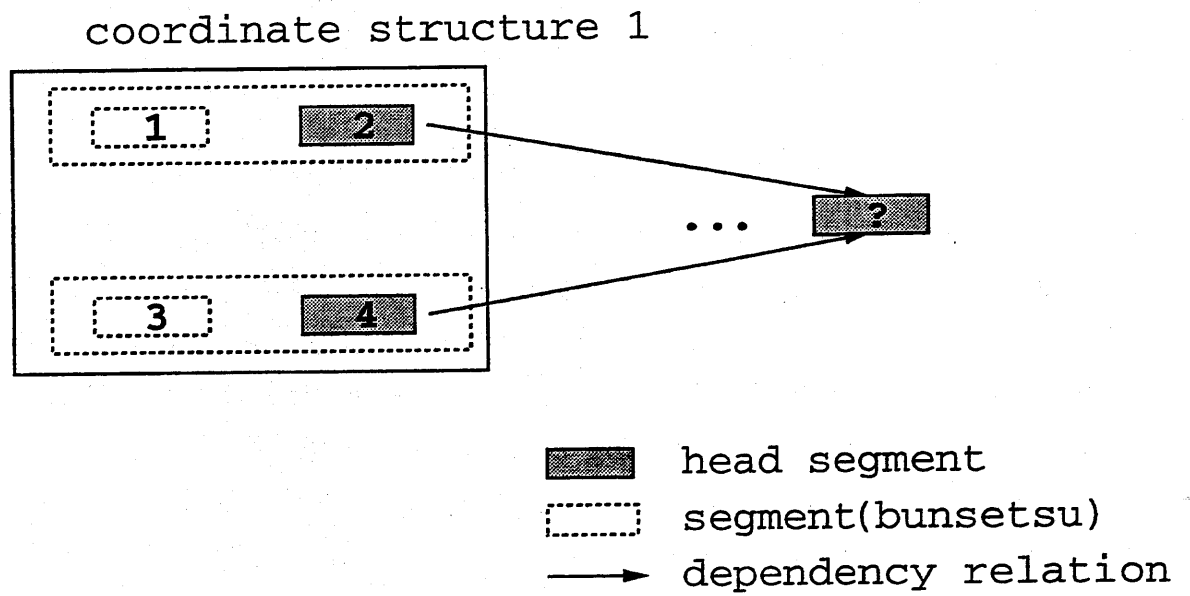


Figure 27. Modifier is a coordinate structure.

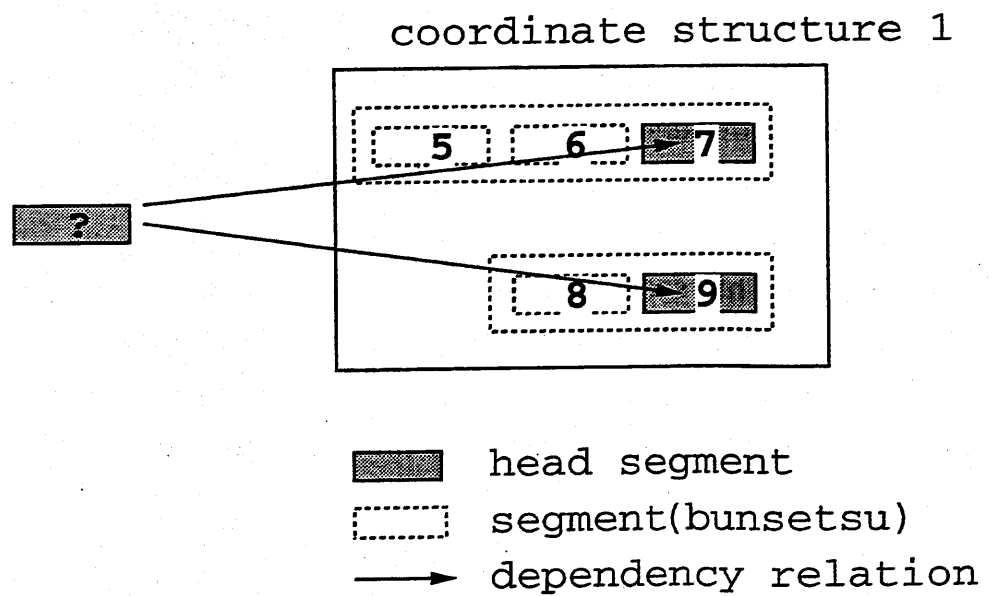


Figure 28. Modifiee is a coordinate structure.

coordinate structure 1 coordinate structure 2

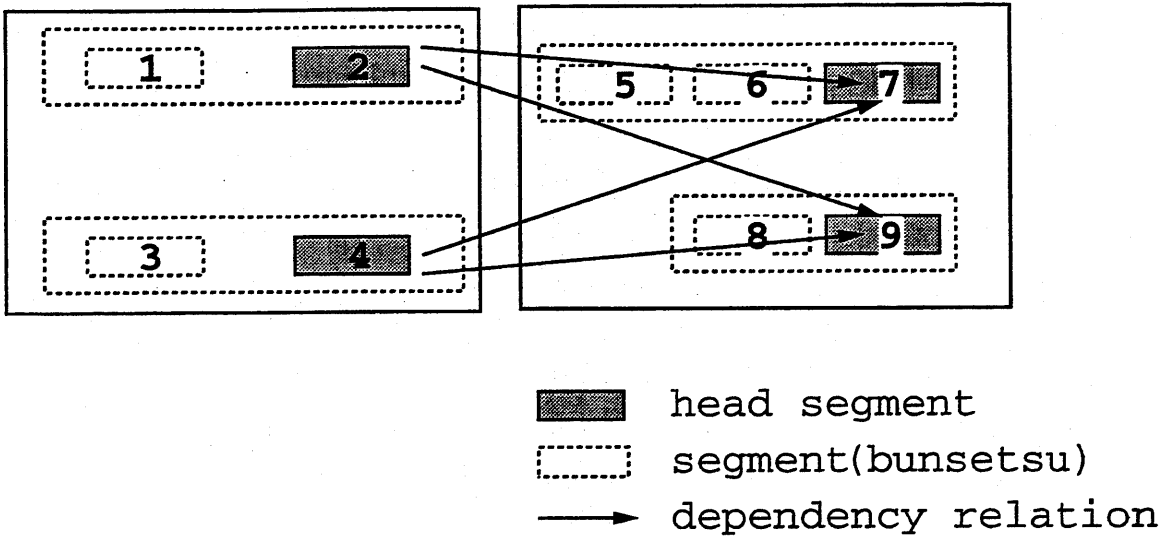


Figure 29. Both modifier and modifiee are coordinate structures.

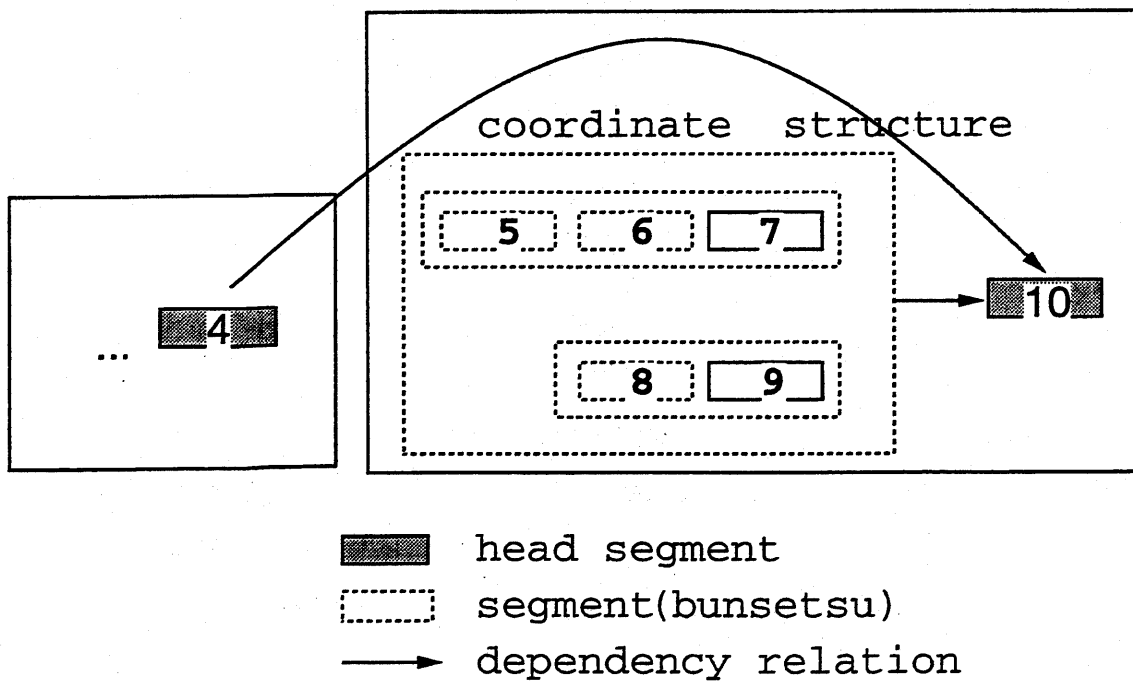


Figure 30. There are coordinate structures between a modifier and a modifiee.

dependency probability of every combination of head segments in the coordinate structure of the modifier and the modifiee.

If the modifier is the structure that has n head segments and the modifiee is the structure that has m head segments, the number of total dependency relations becomes $n \times m$.

When we calculate the probability of a dependency relation (at least either the modifier or the modifiee has a coordinate structure), we must answer the following questions:

1. How can we define distance features, when calculating *Distance Probability* defined in section 3.3, for the every possible combination of the modifier head segments and the modifiee head segments ?
2. How can we calculate the final probability from all the possible combinations of the modifier heads and the modifiee heads

Because Japanese is a head final language, the distance features depend on the modifiee structure. Consider the example in Figure 29. In the case of dependency relations that modify the segment "7", we use the distance features of the upper substructure (consisting of segments "5", "6" and "7") in the modifiee tree, while in the case of dependency relations that modify the segment "9", we use the distance features of the lower one (consisting of segments "8" and "9").

For the second points, we use the harmonic mean of the probabilities of possible dependency relations. The coordinate structure has multiple heads, and the probability of dependency relation is calculated from the multiple probabilities defined by these multiple heads. In this dissertation, we use the harmonic means of multiple dependency probabilities. This comes from the natural intuition that every element of coordinate structure should have close probability values to modify the same modifiee or to be modified from the same modifier. The harmonic mean has a tendency to get more influence by the lower probability than the simple mean of probabilities.

For the n probabilities of dependency relations, the harmonic mean can be calculated by the following equation:

$$P_{hm} = \frac{n}{\sum_{i=1}^n 1/p(i)}$$

P_{hm} : the harmonic mean of n probabilities.

$p(i)$: the probability of n -th dependency relation

Influence on *Distance Probability*

Figure 30 illustrate the case where there are coordinate structures in the inside (not the top node) of the modifiee structure. On the contrary to the above cases, this pattern has single modifier and single modifiee, but the distance features changes by the existence of the coordinate structures.

In our framework, each structure (including coordinate structure) has the distance features (information about the number of segments) in it. For the distance features of a coordinate structure, we simply use the right most substructure of the coordinate structure.

6.4 Training and Test Corpora

For the training and evaluation for the statistical model of coordinate structures, we use Kyoto University corpus [27]. Kyoto University corpus not only has annotation of dependency relation, but also has annotation of coordinate structures, while EDR corpus has no annotation of coordinate structures.

6.4.1 Kyoto University Corpus

Kyoto University parsed corpus consists of the articles of Mainichi newspaper from 'January 1st to 10th, 1995 (about 10,000 sentences.), the editorial articles from 'January 11th to June 30th, 1995 6/30 (about 10,000 sentences). These texts are morphologically analyzed by JUMAN[28], and syntactically analyzed by KNP, and then errors are corrected manually. Kyoto University corpus has segment information, then we train the statistical model based on the segmentation of Kyoto University corpus. We only need to assign segment features when learning from Kyoto University corpus.

Kyoto University corpus distinguishes four kinds of dependency relations:

Coordinate Relation

例) チーズを₁
 食べ, *—P
 ビールを₁
 飲んだ. *

Dependency Relation in a Partial Coordinate Structure

例) 本を—I
 兄の₁
 太郎に, *—P
 ノート—I
 弟の₁
 三郎に₁
 かしている. *

Apposition

例) 泥棒は——
 指輪など, *—A |
 多数の₁ |
 高級品を₁ |
 盗んだ. *

Normal modification relation

- 「僕が書いた (he wrote.)」 (a case element modifies a predicate),
- 「僕の本が (my book)」 (a noun modifies a noun),
- 「書いた本が (written book)」 (adnominal clauses modifies a noun),
- 「書けば, 売れる (If you write, the book would sell well)」 (a subordinate clause modifies the predicate of a main clause)

The type "Coordinate Relation" includes, predicate coordinate and nominal coordinate structures. These structures are marked by the sign "P" in Kyoto University corpus. In this dissertation, we deal with nominal coordinate structures, and consider that predicate coordinate relations as a normal modification

relations. The type “Partial Coordinate Structure” is the structure in which constituents of the structure are the sequences of segments that share the common modifiee. The segments inside such a constituent are marked by the sign “I” in Kyoto University corpus. In the above example, the pair “本を” and “太郎に”, and the pair “ノートを” and “三郎に” are the partial coordinate structures. The correct modifiee of “本を” is not “太郎に” but “かしている.” which is outside of the partial coordinate structure. And the correct modifiee of “ノートを” is not “三郎に” but “かしている.”. In general, a partial coordinate structure appears with another partial coordinate structure to construct a coordinate structure, and the broader coordinate structure modifies the common predicates (in the above example, “かしている”) which follows the coordinate structure. In this dissertation, we do not deal with partial coordinate structures. The third type of structure is “Apposition”. “Apposition” is marked by “A” in Kyoto University corpus. The “Apposition” and “Nominal Coordinate” are sometimes vague, but basically “Apposition” has some markers such as “など” and “から～まで” and so on. We deal with “Apposition” as normal modification relation.

Because Kyoto University corpus does not distinguish predicate coordinate and nominal coordinate structures, we use following heuristics to distinguish these structures.

- If the head word of the elements of a coordinate structure is verb, we define them as the predicate coordinate structure.

6.5 Experiment

6.5.1 Test Data

Both of the training and evaluation are done on the sentences extracted from Kyoto University corpus. Table 14 gives the statistics of these sentences. We divide these sentences into 10 groups and one of them was kept for the evaluation.

6.5.2 Evaluation Measure

For evaluation measures, we use the following two kinds of segment level precision.

Table 14. Statistics of test sentences extracted from Kyoto University corpus

	Subsets		Full
	Sentences Including Nominal Coordinate Structures	Sentences Not Including Nominal Coordinate Structures	
	# of Sentences	2,398 (26.1%)	
# of Segments	29,900 (33.1%)	60,384 (66.9%)	90,284
Ave. # of Segments / Sentence	12.5	8.4	
Dependency Analysis Precision of Basic DA Model (Chapter 3) Segment Level	85.3%	86.7%	86.0%

- evaluation measure (1)

$$\text{Segment Level Precision} = \frac{\text{The Number of Segments whose Modifyer Segment is Estimated Correctly}}{\text{The Number of Segments whose Modifyer Segment is Estimated by the System}}$$

- evaluation measure (2)

$$\text{Segment Level Precision} = \frac{\text{The Number of Segments whose Modifyer Segment and the Type of Modification is Estimated Correctly}}{\text{The Number of Segments whose Modifyer Segment is Estimated by the System}}$$

In the evaluation measure (2), the type of modification means whether the modification is coordinate relation or normal dependency relation.

In the evaluation measure (1), we simply count the correct dependency relations regardless of the type of the dependency relation. In the evaluation measure (2), we define the correct dependency relation as the relation that both the modifyer and the type of dependency relation are correct.

Because it is difficult to discriminate coordinate relation or dependency relation for the cases of predicate coordinate, we ignore the type of modification relation for the predicate coordinate structure in evaluation measure (2).

6.5.3 Results

As training and evaluation data, we used the Mainichi news articles from 'January 1st to 9th, 1995', which contain 9204 sentences (Table 14). We adopted 10-fold cross validation by changing the held out data from 'January 1st to 9th, 1995'. 26% (2398 sentences) of the sentences have noun phrase coordinate structures. Evaluation was done for the sentences that has noun phrase coordinate structures (with NC), that has no noun phrase (without NC), and the whole set of sentences. Table 15 and 16 shows the results. In Table 15 and 16, Base Line is the basic model explained in Chapter 3, but trained by the data explained above. Tables 15 and 16 gives the results by the evaluation measure (1) and (2) defined in section 6.5.2.

As we mentioned in section 6.2, we have an option whether the model distinguishes the statistics of coordinate relations and dependency relations or not. Model1 does not distinguish the statistics learned from the inside of coordinate structures and the statistics learned from the outside of coordinate structures, while Model2 distinguishes those two statistics.

The result shows that both of Model1 and Model2 outperforms the results of the Base Line model. The second column in Table 15 shows that there seems to be no side effect by incorporating coordinate structure analysis. Table 15 shows that Model1 slightly outperforms Model2 for both sentences with NC and without NC. However, Table 16 shows that the performance of Model1 for the sentences with NC is quit low compared with the performance of Model2. This result means that Model1 outputs more correct dependency relations than Model2 does, but outputs more incorrect "types" of dependency relations than Model2 does.

6.6 Conclusion

In this Chapter, we propose a method to treat the coordinate structure identification and the dependency analysis in a uniform way. The final probability of

Table 15. Segment level precision by the evaluation measure (1).

Evaluation (1)	With NC	Without NC	Whole
Base Line	0.8122	0.8593	0.8428
Model1	0.8219	0.87630	0.8572
Model2	0.8195	0.87365	0.8533

Table 16. Segment level precision by the evaluation measure (2).

Evaluation (2)	With NC	Without NC	Whole
Base Line	0.70434	0.8593	0.8052
Model1	0.6959	0.8756	0.8182
Model2	0.7824	0.8699	0.8386

dependency structures reflects the existence of the coordinate structures from two reasons. The first reason is that the existence of coordinate structures changes the distance features of dependency pairs. The second reason is that a coordinate structure has multiple heads, and the probability of dependency relation is calculated from the multiple probabilities defined by these multiple heads. In this dissertation, we use the harmonic means of multiple dependency probabilities. This comes from the natural intuition that all dependency relations of the head segments of a coordinate structure will have high probabilities. The harmonic mean has a tendency to get more influence by the lower probability than the simple mean of probabilities.

By considering coordinate structures, when training the statistical model, segment level parsing accuracy improved about 1% for both sentences with NC and without NC.

7. Conclusion

7.1 Summary

The claim of this dissertation is that statistics of surface features, such as part-of-speech tags and head words, extracted from a large corpus of parsed sentences, along with particular algorithm, can produce accurate parses.

If we want to achieve a higher rate of accuracy, it is necessary to use more information. However, it causes a problem of increasing complexities of models and the effect of sparse-data problem is also inevitable.

The basic statistical model is not so complex as other statistical parsers in the literature of a computational statistical parser. We stick to a statistical model of simple setting aiming at an easy implementation and efficiency of parsing. Instead, we address the problem of subordinate clauses and coordinate structures, which are among the major causes of difficulty in the syntactic analysis of Japanese sentences. We also propose statistical partial parse and redundant parse methods.

In Chapter 5, the scope embedding preference of subordinate clauses is exploited as a useful information source for disambiguating dependencies between subordinate clauses.

We successfully increased the accuracies of both segment level and sentence level dependencies thanks to the estimated dependencies of subordinate clauses.

The model first (partially) disambiguated dependency relations of subordinate clauses, then performed the dependency analysis of the whole sentence. But this is not necessarily the best way to integrate dependency preference of subordinate clauses with basic statistical model (defined in Chapter 3). We need to study the best way to incorporate dependency preference of subordinate clauses into our statistical model.

In Chapter 6, we propose a method to treat the coordinate structure identification and the dependency analysis in a uniform way. The final probability of dependency structures reflects the existence of the coordinate structures from two reasons. The first reason is that the existence of coordinate structures changes the distance features of dependency pairs. The second reason is that a coordinate structure has multiple heads, and the probability of dependency relation is cal-

culated from the multiple probabilities defined by these multiple heads. In this dissertation, we use the harmonic means of multiple dependency probabilities. This comes from the natural intuition that all dependency relations of the head segments of a coordinate structure will have high probabilities. The harmonic mean has a tendency to get more influence by the lower probability than the simple mean of probabilities.

By considering coordinate structures, when training the statistical model, segment level parsing accuracy improved about 1% for both sentences with NC and without NC.

We may achieve higher performance of the coordinate structure analysis by the following facts. First in this experiment, we only used noun phrase coordinate structure information (in Kyoto University corpus, marked by "P"). But sometimes, dependency relations marked by "A" in Kyoto University corpus also seem to construct noun phrase coordinate structures. To improve the precision of coordinate structure analysis, using these data may be effective. Second, our framework captures the adaptability of a coordinate structure indirectly by the similarity of head segments in terms of modification relation to other segments, and does not reflect the other constituents that construct the elements of coordinate structures. By introducing structural similarity measure, we may improve the accuracy much more.

7.2 Future Work

We need to study the best way to integrate the dependency preference of subordinate clauses with the basic statistical dependency analysis model. Also we use the distance features when learning scope embedding preference of subordinate clauses.

To study how far the same technique can be applied to English sentences, we will implement a parser for an English sentence.

We also apply automatic feature selection by decision lists to the probabilistic estimation of basic statistical dependency analysis model in Chapter 3. But to learn decision lists for whole dependency relation needs vast amount of computational resources, and if there are many features, the combination of features increases exponentially. Because distance features are continuous value, there are

a many possibility to divide the range of value into discrete features (for instance, see Table 6). This possibility also leads to the explosion of the combination of features, and naive application of the decision list approach becomes impossible. Then we will first apply the decision list approach to estimate the *Head Collocation Probability*. continuance

References

- [1] E. Alves. The selection of the most probable dependency structure in Japanese using mutual information. *Proceedings of the Annual Meeting of ACL*, pp. 372–374, 1996.
- [2] E. Black. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the 31st Annual Meeting of ACL*, pp. 31–37, 1993.
- [3] R. Bod. Using an annotated corpus as a stochastic grammar. In *Proceedings of the 6th ACL European Chapter*, pp. 37–44, 1993.
- [4] T. Briscoe and J. Carroll. Generalized probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, Vol. 19, No. 1, pp. 25–29, Mar. 1993.
- [5] E. Charniak. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the 14th AAAI*, pp. 598–603, 1997.
- [6] M. Collins. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of ACL*, pp. 184–191, June 1996.
- [7] M. Collins. Three generative, lexicalised models for statistical parsing. pp. 16–23, Jul. 1997.
- [8] M. Collins and J. Brooks. Prepositional phrase attachment through a backed-off model. In *Proceedings of the 3rd Workshop on Very Large Corpora*, pp. 27–38, 1995.

- [9] *EDR Electronic Dictionary Specifications Guide*. Japan Electronic Dictionary Research Institute, Ltd., 1995.
- [10] T. Ehara. Comparison of several statistical parsing methods for Japanese bunsetsu dependency analysis. In *Proceedings of The Fourth Annual Meeting of The Association for NLP*, pp. 382–385. The Association for NLP, 1998.
- [11] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th ACL*, pp. 310–318, Jun. 1996.
- [12] E. W. Fong and D. Wu. Learning restricted probabilistic link grammars. *IJCAI-95 Workshop on New Approaches to Learning Natural Language Processing*, pp. 49–56, Aug 1995.
- [13] M. Ford, J. Bresnan, and R. Kaplan. The sausage machine: A new two-stage parsing method. *Cognition*, Vol. 6, pp. 291–325, 1979.
- [14] M. Fujio and Y. Matsumoto. Japanese dependency structure analysis based on lexicalized statistics. In *Proceedings of the 3rd Conference on EMNLP*, pp. 88–96, 1998.
- [15] F. Fukumoto, H. Sano, Y. Saitoh, and J. Fukumoto. A framework for restricted dependency grammar. In *Proceedings of the 3rd International Workshop on Natural Language Understanding and Logic Programming*, pp. 68–81, 1991.
- [16] F. Fukumoto and J. Tsujii. Kakariuke no kyoudo nimotozuku izon bunpou — seigen izon bunpou —. *Transactions of IPSJ*, Vol. 33, No. 10, pp. 1211–1223, 1992. (in Japanese).
- [17] M. Haruno, S. Shirai, and Y. Oyama. Using decision trees to construct a practical parser. In *Proceedings of the 17th COLING and the 36th Annual Meeting of ACL*, pp. 505–511, 1998.
- [18] D. Hindle and M. Rooth. Structural ambiguity and lexical relations. *Computational Linguistics*, Vol. 19, No. 1, pp. 103–120, 1993.

- [19] W. R. Hogenhout and Y. Matsumoto. Training stochastic grammars on semantical categories. In S. Wermter, E. Riloff, and G. Scheler, editors, *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, pp. 160–172. Springer-Verlag, 1996.
- [20] T. Inui, H. Kimura, and K. Inui. Toukeiteki bubunkaisekiki no furumai nituite. In *Workshop on Proceedings of The Fifth Annual Meeting of The Association for NLP*, pp. 33–40. The Association for NLP, 1999. (in Japanese).
- [21] M. Kameda. A portable & quick Japanese parser: QJP. In *Proceedings of the 16th COLING*, pp. 616–621, 1996.
- [22] T. Kasami. An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA, Report, 1965.
- [23] S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-35, No. 3, pp. 400–401, Mar. 1987.
- [24] J. Kimball. Seven principles of surface structure parsing in natural language. *Cognition*, Vol. 2, pp. 15–47, 1973.
- [25] T. Kodama. *Izon Bunpou no Kenkyu*. Kenkyusha Syuppan, 1987. (in Japanese).
- [26] S. Kurohashi and M. Nagao. A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, Vol. 20, No. 4, pp. 507–534, 1994.
- [27] S. Kurohashi and M. Nagao. Kyoto daigaku tekisuto kopasu purojekuto. In *Proceedings of The Third Annual Meeting of The Association for NLP*, pp. 115–118. The Association for NLP, 1997. (in Japanese).
- [28] S. Kurohashi and M. Nagao. *Nihongo Keitaiso Kaiseki Sisutem JUMAN* version 3.61. Graduate School of Infomatics, Kyoto University, 1999. (in Japanese).

- [29] J. Lafferty, D. Sleator, and D. Temperley. Grammatical trigrams: A probabilistic model of link grammar. In *Proceedings of the AAAI Conference on Probabilistic Approaches to Natural Language*, pp. 89–97, October 1992.
- [30] H. Li. A probabilistic disambiguation method based on psycholinguistic principles. *Journal of Japan Society for Software Science and Technology*, Vol. 13, No. 6, pp. 489–501, 1996.
- [31] D. M. Magerman. Statistical decision-tree models for parsing. In *Proceedings of the 33rd Annual Meeting of ACL*, pp. 276–283, 1995.
- [32] D. M. Magerman and M. P. Marcus. Perl: A probabilistic chart parser. In *Proceedings of the 5th ACL European Chapter*, pp. 15–20, Apr. 1991.
- [33] Y. Matsumoto, A. Kitauchi, T. Yamashita, and Y. Hirano. Japanese morphological analyzer ChaSen2.0 users manual. Information Science Technical Report NAIST-IS-TR99008, Nara Institute of Science and Technology, 1999.
- [34] F. Minami. *Gendai Nihongo no Kouzou*. Taishuukan Shoten, 1974. (in Japanese).
- [35] F. Minami. *Gendai Nihongo Bunpou no Rinkaku*. Taishuukan Shoten, 1993. (in Japanese).
- [36] M. Nagao. *Sizengengo Shori*. The Iwanami software science series. Iwanami Shoten, 1996. (in Japanese).
- [37] National Language Research Institute. *Word List by Semantic Principles*. Syuei Syuppan, 1964,1993. (in Japanese).
- [38] C. Pollard and I. A. Sag. *Head-Driven Phrase Structure Grammar*, Vol. 2. The University of Chicago Press, 1994.
- [39] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
- [40] A. Ratnaparkhi. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of the Second Conference on EMNLP*, pp. 1–10, Aug 1997.

- [41] R. L. Rivest. Learning decision lists. *Machine Learning*, Vol. 2, pp. 229–246, 1987.
- [42] Y. Schabes. Stochastic lexicalized tree-adjoining grammars. In *Proceedings of the 14th COLING*, pp. 425–432, 1992.
- [43] Y. Schabes, M. Roth, and R. Osborne. Parsing the Wall Street Journal with the inside-outside algorithm. In *Proceedings of the 6th ACL European Chapter*, pp. 341–347, 1993.
- [44] K. Shirai, K. Inui, T. Tokunaga, and H. Tanaka. An empirical evaluation on statistical parsing of japanese sentences using lexical association statistics. In *Proceedings of the Third Conference on EMNLP, Granada, Spain*, pp. 80–87, 1998.
- [45] S. Shirai, S. Ikehara, A. Yokoo, and J. Kimura. A new dependency analysis method based on semantically embedded sentence structures and its performance on Japanese subordinate clauses. *Transactions of IPSJ*, Vol. 36, No. 10, pp. 2353–2361, 1995. (in Japanese).
- [46] K. Shutou, K. Yosimura, and K. Tuda. Nihongo gijutubun niokeru heiretu kouzou. *Transactions of IPSJ*, Vol. 27, No. 2, pp. 183–190, 1986. (in Japanese).
- [47] D. Sleator and D. Temperley. Parsing english with a link grammar. Technical report CMU-CD-19-196, Department of Computer Science, 1991.
- [48] A. Stolcke. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, Vol. 21, No. 2, pp. 165–201, Jun. 1995.
- [49] A. Suganuma, H. Yamamura, and K. Ushijima. Analysis of coordinate structures in japanese documents, and its application to a writing tool. *Transactions of IPSJ*, Vol. 38, No. 7, pp. 1296–1307, Jul. 1997.
- [50] A. Tagami, T. Tanabe, Y. Tomiura, and T. Hitaka. Probabilistic context free grammar expressing the number of modifier. *Technical Report of IEICE*, Oct 1996.

- [51] K. Uchimoto, S. Sekine, and H. Isahara. Japanese dependency structure analysis based on maximum entropy models. *IPSG SIG NOTES*, Vol. 98, No. (98-NL-128), pp. 31-38, 1998.
- [52] T. Utsuro, S. Nishiokayama, M. Fujio, and Y. Matsumoto. Extraction of preference of dependency between japanese subordinate clauses from corpus and its evaluation. In *Journal of NLP*, Oct. 1999.
- [53] D. Yarowsky. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proceedings of the 32nd Annual Meeting of ACL*, pp. 88-95, 1994.
- [54] D. Younger. Recognition and parsing of context-free languages in time n^3 . *Information and Control*, 10(2), pp. 189-208, 1967.

Appendix

A Japanese *subordinate clause* is a clause whose head segment satisfies the following properties.¹⁴

1. The content words part of the segment is one of the following types:
 - (a) A predicate (i.e., a verb or an adjective).
 - (b) nouns and a copula like "*Noun₁ dearu*" (in English, "*be Noun₁*").
2. The function words part of the segment is one of the following types:
 - (a) Null.
 - (b) Adverb type such as "*Verb₁ ippou-de (Clause₂)*" (in English, "(subject) *Verb₁ ...*, *on the other hand*, *(Clause₂)*").
 - (c) Adverbial noun type such as "*Verb₁ tame*" (in English, "*in order to Verb₁*").
 - (d) Formal noun type such as "*Verb₁ koto*" (in English, gerund "*Verb₁-ing*").
 - (e) Temporal noun type such as "*Verb₁ mae*" (in English, "*before (subject) Verb₁ ...*").
 - (f) A predicate conjunctive particle such as "*Verb₁ ga*" (in English, "*although (subject) Verb₁ ...*").
 - (g) A quoting particle such as "*Verb₁ to (iu)*" (in English, "*(say) that (subject) Verb₁ ...*").
 - (h) (a)~(g) followed by topic marking particles and/or sentence-final particles.

¹⁴ This definition includes adnominal clauses or noun phrase modifying clauses "*Clause₁ NP₁*" (in English, relative clauses "*NP₁ that Clause₁*"). Since an adnominal clause does not modify any posterior subordinate clauses, but modifies a posterior noun phrase, we consider adnominal clauses only as modifyees when analyzing dependencies between subordinate clauses in a sentence.

List of Publications

Journal Papers

- Dependency analysis based on lexical collocation probability and its Evaluation, Masakazu Fujio, and Yuji Matsumoto, Transactions of IPSJ, Dec.1999
- Extraction of Preference of Dependency Between Japanese Subordinate Clauses from Corpus and its Evaluation, Takehito Utsuro, Shigeyuki Nishioyama, Masakazu Fujio, and Yuji Matsumoto, Oct. 1999, Journal of Natural Language Processing, vol 6, No 4
- An Integrated Parsing Method using Stochastic Information and Grammatical Constraints, Osamu Imaichi, Yuji Matsumoto, Masakazu Fujio, Jul.1998, Journal of Natural Language Processing, vol 5, No 3

International Conference

Japanese Dependency Structure Analysis based on Lexicalized Statistics, M. Fujio and Y. Matsumoto, Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing, pp88-96, 1998

Other Publications

- 「統計情報を用いた日本語係り受け解析システムの実装と評価」, 藤尾 正和 竹内 孔一 山本 靖 平野 善隆 河部 恒 松本 裕治, 「大規模資源と自然言語処理」シンポジウム論文集, Nov. 1996, pp174-181
- 「EDR 括弧付きコーパスを利用した, 統計的日本語係り受け解析」, EDR 電子化辞書シンポジウム, Apr. 1997
- 「Japanese Dependency Structure Analysis based on Statistics」, Fujio Masakazu and Yuji Matsumoto, IPSG SIG NOTES, VOL 97, No. 4 (97-NL-117), pp83-90, 1997
- 「Statistical Partial Parsing and Redundant Parsing based on the lexical term collocation probability」, Masakazu Fujio, and Yuji Matsumoto, "Pro-

ceedings of Work Shop Program The Fifth Annual Meeting of The Association for Natural Language Processing", 1999, The Association for Natural Language Processing, pp71-78,