Corpus-based Parsing and Sublanguage Studies

by

Satoshi Sekine

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Computer Science Department
New York University

May 1998

Acknowledgements

I would like to greatfully acknowledges Ralph Grishman, my supervisor, in the Computer Science Department at New York University for his help, advice and encouragement. His wide knowledge and keen insight provided me with substantial support enabling the completion of this thesis.

Also I wish to thank the members of the Proteus Project at New York University for their encouragement and support: Catherine Macleod, John Sterling, Adam Meyers, Andrew Borthwick and Roman Yangarber.

I wish to thank Jun'ichi Tsujii of Tokyo University who introduced me the joy and the agony of NLP research, and, also, to thank my thesis committee members, Aravind Joshi of the University Pennsylvania and Ernest Davis, Edmond Shonberg and Naomi Sager of New York University.

This thesis benefitted from discussions with many people. In particular, I would like to express my thanks to Kyo Kageura, Yuji Matsumoto, Hozumi Tanaka, Takenobu Tokunaga, Makoto Nagao, Tadao Kurohashi, Kentaro Inui, Kentaro Torisawa, Hitoshi Isahara, Kiyotaka Uchimoto, Fumiyo Fukumoto, Jun'ichi Fukumoto, Jussi Karlgren, Slava Katz, Christer Samuelsson, Ken Church and Sarah Taylor.

Also, I would like to thank the users of the Apple Pie Parser who gave me useful comments, in particular: Philip Resnik, Seiji Ookura, Alexander Krotov, Boris Granveaud, Young-Suk Lee, Inderjeet Mani and many others.

Finally, I am thankful to all the people whom I have met in my life, in particular, to my family. To my parents for raising me to be as I am, to my daughter, Yuki, for her smile, which encourages me every day and night, and to my wounderful wife, Tomoe, for her patience and encouragement.

Abstract

There are two main topics in this thesis, a corpus-based parser and a study of sublanguage.

A novel approach to corpus-based parsing is proposed. In this framework, a probabilistic grammar is constructed whose rules are partial trees from a syntactically-bracketed corpus. The distinctive feature is that the partial trees are multi-layered. In other words, only a small number of non-terminals are used to cut the initial trees; other grammatical nodes are embedded into the partial trees, and hence into the grammar rules. Good parsing performance was obtained, even with small training corpora. Several techniques were developed to improve the parser's accuracy, including in particular two methods for incorporating lexical information. One method uses probabilities of binary lexical dependencies; the other directly lexicalizes the grammar rules. Because the grammar rules are long, the number of rules is huge – more than thirty thousand from a corpus of one million words. A parsing algorithm which can efficiently handle such a large grammar is described. A Japanese parser based on the same idea was also developed.

Corpus-based sublanguage studies were conducted to relate the notion of sublanguage to lexical and syntactic properties of a text. A statistical method based on word frequencies was developed to define sublanguages within a collection of documents; this method was evaluated by identifying the sublanguage of new documents. Relative frequencies of different syntactic structures were used to assess the domain dependency of syntactic structure in a multi-domain corpus. Cross-entropy measurements showed a clear distinction between fiction and non-fiction domains. Experiments were then performed in which grammars trained on individual domains, or sets of domains, were used to parse texts in the same or other domains. The results correlate with the measurements of syntactic variation across domains; in particular, the best performance is achieved using grammars trained on the same or similar domains.

The parsing and sublanguage techniques were applied to speech recognition. Sublanguage techniques were able to increase recognition accuracy, and some promising cases were found where the parser was able to correct recognition errors.

Contents

List of Figures

List of Tables

Chapter 1

Overview

1.1 Introduction

Corpus-based language processing plays a major role in current natural language processing research. The author believes it was initially motivated by the emphasis on observation rather than introspection. In conventional natural language processing, although some observation was used, e.g. key word in context or KWIC, it was used as a help for creating knowledge manually. However, the current corpus based approaches use the data itself to make such kinds of knowledge based on statistics etc., or in a sense, the data itself is used as the knowledge. For example, grammar rules or tagging rules are induced from an annotated corpus, and word N-gram, which is simple statistics of the source data, is used in many applications. Corpus-based methods are also useful for disambiguation. In the conventional or symbolic methods, in order to produce unique analyses, we need an enormous amount of real world knowledge, which, the author believes, there is no means to acquire and formulate at the moment. By corpus-based methods, this problem can be solved by preferring the analysis which was found in the data more often. Although this can hardly be perfect, it gives empirically better analyses than, for example, by choosing based on heuristic rules.

In the course of the investigation of the corpus-based approach in this thesis, some other merits are found. One of them is that an annotated corpus is very useful for knowledge acquisition even if the size of the data is not huge. Annotating data is, in many cases, a costly process. However we found that it is sometimes less expensive than creating linguistic knowledge itself, e.g. a dictionary or a grammar. Once a corpus is annotated, then we can acquire knowledge from the data, most importantly, using different methodologies based on different theoretical frameworks, although the way of annotation imposes some limitations. For example, in acquiring grammatical knowledge, one can acquire general CFG rules or lexicalized syntactic knowledge from the same annotated corpus. The point is that this kind of freedom has not

existed in the conventional method of manual knowledge creation. Previously, if one creates a CFG grammar, it is very hard to produce a lexicalized dependency grammar based on the CFG grammar and vice versa.

Another benefit of the creation of an annotated corpus is that it provides a platform to evaluate systems. This fact is actually encouraging researchers to try different methodologies, because we can compare one system to another.

Also, a similar merit derived from corpus-based approach is that we now have an objective means to evaluate theories or hypotheses, for example the sublanguage hypothesis. It has been discussed that languages are different in different domains or sublanguages. However, because of the lack (or ignorance) of objective assessments, the discussion looked very subjective. So, we did not see clear evidence that sublanguage does really exist or how these sublanguages are different by objective measure. It is certainly beneficial to observe some objective evidence in order to design domain dependent systems.

1.2 Outline

These are two main topics in the thesis, a corpus-based parser and a study of the notion of sublanguage using multi-domain corpora. Also, the combination of the two topics, the parser based on sublanguage, will be described. There are three chapters, Chapter 2, 3 and 4 for the three topics. As the background of each topic is different, an introduction and review of related work will be presented separately in each chapter. Any reader who is interested in one of the first two chapters should not encounter any difficulty by skipping the other part, although for people who are interested in the parsing experiments based on sublanguage, the author recommends reading the previous chapters beforehand, in particular Chapter 3.

It should be possible for anyone with sufficient knowledge of natural language processing and basic knowledge about statistics to work their way through this thesis, though this may require considerable effort for a total novice. There are several good textbook in natural language processing, for example, [Grishman 86]. Anyone who wants to know the basics of statistics used in this thesis should refer to [Charniak 93] or [Krenn and Samuelsson 97].

Chapter 2 describes a probabilistic parser whose grammar is obtained from a syntactically tagged corpus. The distinguishing characteristic of the grammar is that it has only a small number of non-terminals. Both a two non-terminal grammar and five non-terminal grammar will be described. The parser is a bottom-up chart parser. As the number of grammar rules is enormous, e.g. thirty to forty thousand, the parsing algorithm has some tricks. A trial of this approach for Japanese and an application of the parser for continuous speech recognition will be presented. Some parts of the work reported in this chapter were previously published in [Sekine and Grishman 95], [Sekine et al. 97a] and [Sekine et al. 97b]. The parser and the evaluation program are available as public domain software from

[Sekine 96a] and [Sekine and Collins 97]. Also, the work was referenced in a book [Young and Bloothooft 97] and in a review [Bod and Scha 96].

Chapter 3 describes a study of the sublanguage notion based on corpora. First, statistical measures are used to find sublanguage from newspaper articles. Then multi-domain corpora are used for a study of sublanguage features in terms of lexical and syntactic characteristics. Finally, an application of sublanguage to continuous speech recognition will be described. Some parts of the work reported in this chapter were previously published in [Sekine 94a], [Sekine 94b], [Sekine 95], [Sekine et al. 95], [Sekine and Grishman 96], [Sekine et al. 97a], [Sekine and Grishman 97] and [Sekine 97]. Also the work was referenced in a book [McEnery and Wilson 96] and appeared in a chapter of a book [Jones and Somers 97].

Chapter 4 describes experiments with the parser based on the sublanguage notion. This combines the approaches described in the previous two chapters. Several sublanguage grammars are acquired from different domain corpora, and the performance of parsing texts from different domain with different grammars is observed. Parts of the work reported in this chapter were previously published in [Sekine and Grishman 97] and [Sekine 97].

Finally, in Chapter 5, closing discussions and possible future work will be presented.

Chapter 2

Corpus-based Parser

2.1 Introduction

The availability of large, syntactically-bracketed corpora such as the University of Pennsylvania Tree Bank (PennTreeBank) [Marcus 96] affords us the opportunity to automatically build or train broad-coverage grammars. Although it is inevitable that an annotated corpus would contain errors, statistical methods and the size of the corpus may be able to ameliorate the effect of individual errors. Also, because a large corpus will include examples of many rare constructs, we have the potential of obtaining broader coverage than we might with a hand-constructed grammar. Furthermore, experiments over the past few years have shown the benefits of using probabilistic information in parsing, and the large corpus allows us to train the probabilities of a grammar [Fujisaki 84] [Garside and Leech 85] [Chitrao and Grishman 90] [Magerman and Weir 92] [Black et al. 93] [Bod 93] [Magerman 95] [Collins 96].

A number of recent parsing experiments have also indicated that grammars whose production probabilities are dependent on the context can be more effective than context-free grammars in selecting a correct parse. This context sensitivity can be acquired easily using a large corpus, whereas human ability to compute such information is obviously limited. There have been several attempts to build context-dependent grammars based on large corpora [Chitrao and Grishman 90] [Simmons and Yu 91] [Magerman and Weir 92] [Schabes and Waters 93] [Black et al. 93] [Bod 93] [Magerman 95].

As is evident from the two lists of citations, there has been considerable research involving both probabilistic grammar based on syntactically-bracketed corpora and context-sensitivity. Some of these will be explained in the next section.

In this chapter, a parsing method which involves both probabilistic techniques based on a syntactically-bracketed corpus and context sensitivity will be presented. The idea is based on a very simple approach which allows us to create an efficient parser and to make use of a very large tree bank. Experiments for English with

some variations and a trial for Japanese using a similar idea will be reported. Also, an application of the parser to continuous speech recognition will be described.

2.2 Prior Work

There has been a good deal of work in corpus based approaches to parsing. When examining these methods, we can distinguish two different kinds of corpus, unannotated corpus and annotated corpus. Prior to the University of Pennsylvania Tree Bank (PennTreeBank), an annotated corpus [Marcus 96], [Marcus 93], the majority of the research in the field had been done using unannotated corpora in combination with some existing grammar. This was mainly because there had been no widely-known, publically and easily available syntactically annotated corpora. In this section, four projects which used unannotated corpora are described.

Since then, there have been several research effort which used syntactically annotated corpora, like the PennTreeBank (Wall Street Journal or ATIS domain). Six such projects will be described in this section. Some of them are focused on context sensitivity with different frameworks and some emphasize lexical dependency relationships. Since the same training data and evaluation data are used in many projects, we can compare the performance among them.

The approaches to corpus-based parsers described in this section are summarized in Table 2.1. Each project is represented by the first author and the year of

Researcher		annotated	context sensitive	lexical
Fujisaki	(84)	No	No	No
Chitrao	(90)	No/Yes(small)	Yes	No
Magerman	(91)	Use a parser	Yes	No
Pereira	(92)	Partially	No	No
Bod	(92)	Yes	Yes (Global)	No
Briscoe	(93)	Yes (hand)	Yes (LR)	No
Black	(93)	Yes	Yes	Yes
Magerman	(95)	Yes (WSJ)	Yes	Yes
Rayner	(96)	Yes	Yes (Global)	No
Collins	(96)	Yes (WSJ)	Limited	Yes
Sekine		Yes (WSJ)	Yes (Global)	No/Yes

Table 2.1: Summary of corpus-based parsing approaches

the publication. The first column indicates if the training corpus is annotated or unannotated. 'WSJ' means the PennTreeBank, Wall Street Journal section. The second column indicates if it uses context in the parsing. The use of context differs system by system. 'LR' means probabilistic LR parser, which is different from the

other parsers. 'Global' means that these methods try to use global context compared to the rest of the methods, which use relatively selective information about their context. The third column indicates if it uses lexical information. In the last row, the method proposed in this thesis is explained. It emphasizes context sensitivity and tries it with and without lexical information.

2.2.1 Iterative Learning Algorithm

One of the earliest experiments on corpus-based parsing was conducted by Fujisaki [Fujisaki 84] [Fujisaki et al. 89]. It involved probabilities of pure CFG rules computed from an unannotated corpus using the inside-outside algorithm and unsupervised iterative training. They use a hand crafted grammar as the base of the experiment. The core idea of the probability estimation is iterative training. The inside-outside algorithm, which is motivated by the forward-backward algorithm, was used to reduce the expensive computation. For an explanation of the inside-outside algorithm, see for example [Charniak 93]; only the iterative training will be explained here. The algorithm tried to assign a probability for each grammar rule, say $P_i(r_j)$ is the probability of rule r_j after the i-th iteration. Each grammar rule has an initial probability $P_0(r)$. In each iteration, all the sentences in the corpus are parsed and all possible parse trees are generated. The tree probability is calculated based on the rule probability in the previous iteration or the initial probabilities. Then we count up how often rules are used in the entire corpus by accumulating the tree probabilities in which rule r_j is used. We denote the count for rule r_j by $C(r_j)$. Then the new rule probabilities are defined by the following:

$$P_i(r_j) = \frac{C(r_j)}{\sum_{LHS(r_k)=LHS(r_j)} C(r_k)} \tag{2.1}$$

The idea is that after each iteration, the probabilities of relatively frequent rules may increase, whereas the probabilities of less frequent rules may decrease.

This looks appealing, but as was described in [Charniak 93] there is a local minimum problem. Charniak reported an experiment with the inside-outside algorithm using a grammar of all possible fixed length rules. In the experiment, 300 different randomly-set initial probabilities were assigned for the grammar, and 300 different final sets of probabilities were generated by the algorithm. The result was none of them were the same. Furthermore, out of the 300 trials, only one of the results was very similar to the desirable one which was derived by a known CFG, and the other 299 were stuck in non-optimal local minima.

An interesting way to force the initial probabilities into the correct part of the parameter space was proposed by [Pereira and Schabes 92]. They tried the inside-outside algorithm with partially annotated corpora. It starts with a grammar in Chomsky normal form containing all possible productions. They introduced a partially annotated corpus, in other words 'seeds', which are known correct relationships. The performance with the seeds was significantly better than that without

the seeds. This demonstrates the importance of annotated training data.

2.2.2 Context Sensitive Parsing

The previous parsers used the pure context free grammar formalism. The method described here tried to use context sensitivity in parsing.

Chitrao and Grishman [Chitrao and Grishman 90], [Chitrao 90] used the iterative processing using an unannotated corpus based on a hand crafted grammar. They used an unannotated corpus just like the Fujisaki's experiment. The specialties of the method were 'fine grained statistics' and 'heuristic penalties'.

'Fine grained statistics' tried to capture some context sensitivities. A rule can occur several places in several rules, but its appearance at certain locations of a certain rule may have a greater probability than the other places. The following example describes the situation:

```
X -> A B C | D B E
B -> P Q | R S
```

Suppose in the two grammar rules for `B`, `B -> P Q` is more likely in the context `A B C`, while `B -> R S` is more likely in the context of `D B E`. An ordinary probabilistic CFG grammar cannot capture this context dependency, but they proposed two methods to handle the phenomena. One is to rename the different instances of the same non-terminal so that each renamed non-terminal will now represent each different context. This seems a good method, but they did not pursue the idea because of difficulties on implementation. The other idea is to incorporate context information in the probability. The context dependent probabilities can be computed by the same iterative algorithm. It led to a more informed probabilistic grammar and the accuracy was improved by 13% in their experiment.

'Heuristic penalties' aimed to speed up the parser at a little sacrifice of accuracy. It penalized shorter hypotheses so that longer but possibly preferable hypotheses can get more credits in their chart parsing algorithm. By this modification, the parser lost the characteristic of being best-first and this resulted in a slight degradation of accuracy. However, the number of edges, which is a rough indicator of parsing speed, was reduced almost 20% by the method.

They also made experiments with a supervised method which will be described later in this section. Although only 107 annotated sentences are used and the benefit of the method is not very clear, the work should be noted as an early trial of the supervised method.

2.2.3 Towards Supervised Methods

The methods described up to now are unsupervised methods. They use existing grammars and try to assign probabilities for the rules by iterative approach. Two approaches which began the trend towards supervised methods are described in this

subsection. When the two experiments were conducted, large annotated corpora were not available. The first approach used an existing parser to create training data, and the other also used an existing parser along with human intervention to choose the appropriate analysis.

Magerman [Magerman and Marcus 91a] proposed a probabilistic parser, *Pearl.* It did not use an iterative training process; instead, probabilities were defined just by counting the usage of rules in a corpus. Sentences in ATIS domain corpus were parsed by the augmented version of the PUNDIT parser and probabilities are assigned for the un-augmented version of the PUNDIT grammar.

It used contextual information, namely the parts-of-speech of the surrounding words and rule of the parent node. The tree probability is calculated by the geometric mean of element tree probabilities instead of the product of the probabilities. It uses bottom-up chart parsing mechanism, so, it can handle idioms and a word lattice input, which improves the accuracy. The overall parsing accuracy was measured on 40 test sentences. It produced a parse tree for 38 input sentences and 35 of them were equivalent to the correct parse tree produced by the augmented version of PUNDIT. They concluded that comparable performance was achieved without the painfully hand-crafted augmentation of the grammar.

Briscoe and Carroll [Briscoe and Carroll 93] created a probabilistic LR parser with a unification-based grammar. As states of the LR table are related to the parse context, it becomes a context sensitive probabilistic model if the probability is calculated directly from the corpus, instead of derived from probabilistic CFG rules. In the training phase, human intervention was involved to choose the correct transition in LR table. They counted the frequency with which each transition from a particular state has been taken and converted these to probabilities such that the probabilities assigned to all the transitions from a given state sum to one. They reported results of a pilot study training on 150 noun definition from the Longman Dictionary of Contemporary English and tested on 55 new definitions of length less than 10 words. Out of 55 test sentences, 41 (74.5%) were parsed correctly with the highest ranked analysis, while in 6 cases, the correct analysis was the second or third most highly ranked. An interesting work on this line was reported in [Inui et al. 97].

2.2.4 History Based Parsing

All the approaches to grammar learning in this subsection and the subsections which follow used hand annotated corpora for their training. In other words, these are completely supervised methods. Some of them use the Wall Street Journal text of the PennTreeBank which will also be used in the experiments of this thesis.

A group at IBM [Black et al. 93] built a parser which used context information, including the dominating production and the syntactic and semantic categories of words in the prior derivation. Namely what it tried to calculate is the probability

of generating a node based on some information in the parent node.

$$p(Syn, Sem, R, H_1, H_2 | Syn_p, Sem_p, R_p, I_{pc}, H_{1p}, H_{2p}) \tag{2.2}$$

Here, subscript p indicates the parent node, Syn for syntactic category, sem for semantic category, R for the rule, I for the index as a child of the parent constituent, H_1 for the primary lexical head and H_2 for the secondary lexical head. So the context information they used was the information in the parent node. This was estimated by the combination of the following five factors. The third factor was calculated by decision tree method and the others were computed by n-gram models.

$$p(Syn | R_p, I_{pc}, H_{1p}, Syn_p, Sem_p) \tag{2.3}$$

$$p(Sem | Syn, R_p, I_{pc}, H_{1p}, H_{2p}, Syn_p, Sem_p) \tag{2.4}$$

$$p(R | Syn, Sem, R_p, I_{pc}, H_{1p}, H_{2p}, Syn_p, Sem_p) \tag{2.5}$$

$$p(H_1 | R, Syn, Sem, R_p, I_{pc}, H_{1p}, H_{2p}) \tag{2.6}$$

$$p(H_2 | H_1, R, Syn, Sem, R_p, I_{pc}, Syn_p) \tag{2.7}$$

The base grammar used in the experiment was a broad-coverage, feature-based unification grammar. It is a context free grammar with about 672 rules with 21 features. The experiment used 9,000 sentences, which is in a limited vocabulary of about 3,000 words for training. The parsing result was evaluated on 1,600 sentences. The History based grammar outperformed their probabilistic CFG, increasing the parsing accuracy rate from 60% to 75%, a 37% reduction in error. This is an interesting attempt to incorporate context in parsing. Another interesting point of the algorithm is that it combines grammatical clues and lexical clues into one mechanism by the probabilistic model and decision tree metric. The decision tree is a general tool in the sense that it can be trained so that it utilizes the most useful feature among several different kinds of features. (A good explanation of the decision tree method can be found in [Russell and Norvig 95].)

2.2.5 Decision Tree Parsing

Magerman, who was a member of the project explained in the previous subsection, tried to make more extensive use of the decision tree [Magerman 95]. It used Wall Street Journal of the PennTreeBank in the experiment. The parser had three decision tree models which take context into account. The three models are a part-of-speech tagging model, a node-expansion model, and a node-labeling model. The system creates a parse tree in a bottom-up and left-to-right fashion. At each node, it produced probabilities of the tag, the label, and the relation of the node to its upper constituent. The relation is one of the following five values:

- **right** - the node is the first child of a constituent

- **left** - the node is the last child of a constituent
- **up** - the node is neither the first or the last child of a constituent
- **unary** - the node is a child of a unary constituent
- **root** - the node is the root of the tree

An example of the tree representation is shown in Figure 2.1. The order of construction is indicated in the figure. The decisions were made based on lexical and

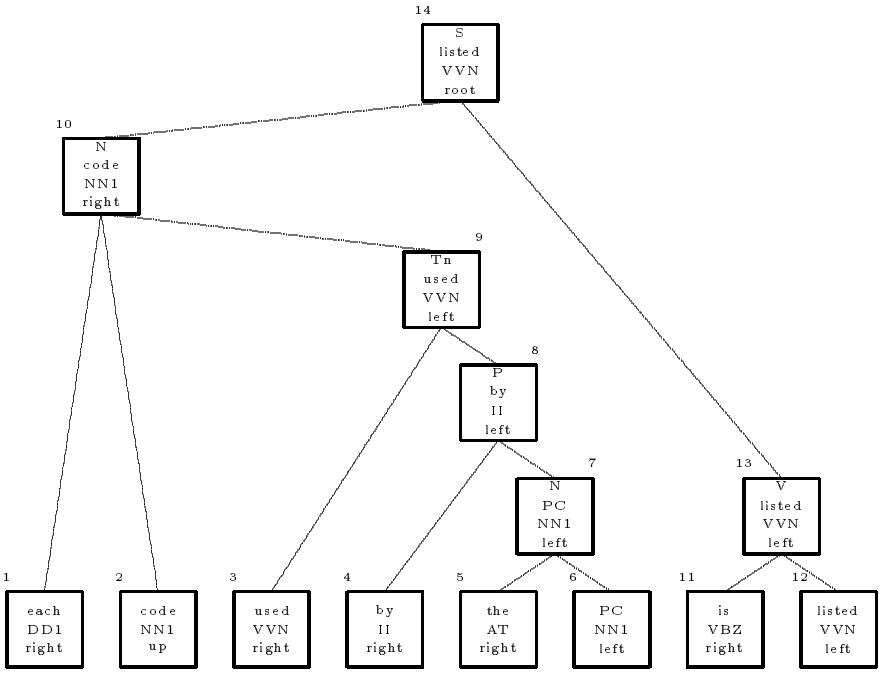

Figure 2.1: Example of sentence representation

contextual information of the parent and the child of the node. The number of features is large and hence problematic to utilize, but this problem was solved using the decision tree method which finds only the useful features. It searches for the parse tree with the highest probability, which is the product of the probabilities of each of the actions made in constructing the parse. The performance of the parser was outstanding. Its precision and recall for the sentences shorter than 40 words were 86.3% and 85.8%, respectively. This was the first parser to approach the current state-of-the-art performance on the task of parsing Wall Street Journal sentences.

2.2.6 Data Oriented Parsing

Bod [Bod 92] [Bod 95] [Bod and Scha 96] explored the idea of Data Oriented Parsing. In this framework, all possible tree fragments in a hand annotated corpus are regarded as rules of a probabilistic grammar. For an input sentence, the entire tree is constructed as a combination of the fragments so that the product of the probabilities is maximum. So it used global context information if it is possible. In order to deal with the combinational explosion of the number of derivations, they introduced the Monte Carlo Parsing algorithm, which estimates the most probable parse by sampling random derivations. It is an approximation but the error can be made arbitrarily small. They conducted an experiment using a set of 600 trees from the PennTreeBank ATIS corpus. From 500 training trees, they obtained roughly 350,000 distinct subtrees. The parsing result was evaluated on the remaining 100 trees. The input was parts-of-speech sequences, not sentences. Accuracy was reported in three categories as shown in Table 2.2.

- Parse accuracy: percentage of the selected parses that are identical to the corresponding test set parse
- Sentence accuracy: percentage of the selected parses in which no brackets cross the brackets in the corresponding test set
- Bracketing accuracy: percentage of the brackets of the selected parses that do not cross the brackets in the corresponding test set parse
- Coverage: percentage of the test sentences for which a parse was found

Parse accuracy	64%
Sentence accuracy	75%
Bracketing accuracy	94.8%
Coverage	98%

Table 2.2: Result of DOP

This algorithm takes context dependency into account. However, as they extracted all possible tree fragments, the number of possible derivations becomes enormous, so some technique to overcome the problem is needed [Sima'an 97].

2.2.7 Parsing by Lexical Dependency

This is another state-of-the-art parser, proposed by Collins [Collins 96]. It used an annotated corpus, Wall Street Journal of the PennTreeBank, like the method by Magerman. The difference is that this method heavily relies on the lexical information or lexical dependencies rather than context information. The philosophy

of this parser is similar to the idea of dependency grammar. A head is defined in every constituent and the dependencies are defined between the head and all the other siblings. Lexical dependency relationships are accumulated from the training corpus. For an input sentence, it uses the probability of the relationships to make the most probable complete dependency structure. In order to solve the sparseness problem, it employs a back-off strategy using parts-of-speech. It introduced a notion of BaseNP, which means non-recursive or lowest NP, as a pseudo token. Also, in order to capture some linguistic characteristics of English, six kinds of feature are added to the calculation. These are the order of two words, adjacency of the two words, whether there is a verb between the words, how many commas are between them, whether there is a comma just after the first word, and whether there is a comma just before the second word. The performance is comparable to the parser by Magerman. Its best labeled-recall was 85.3% and labeled-precision was 85.7%. The interesting comparative experiment reported was that if it ignores the lexical information but uses only parts-of-speech information, the performance decrease significantly to 76.1% recall and 76.6% precision.

Recently [Collins 97] reported a new version of the parser. It includes a generative model of lexicalized context-free grammar and a probabilistic treatment of both subcategorization and wh-movement. The recall and precision of the new parser are 88.1% and 87.5%. Also [Ratnaparkhi 97] proposed a parser using similar information based on maximum entropy model. The parsing strategy is unique and it can generate multiple trees in linear observed time. The performance is 87.5% and 86.3% of recall and precision. [Charniak 97] reported a parser based on similar framework.

2.2.8 Explanation-based Learning

This is an interesting framework for constructing grammars which is closely related to the framework presented in this thesis.

It is based on Explanation-Based Learning (EBL) firstly introduced in Artificial Intelligence, and applied to parsing by [Rayner 88], [Samuelsson 94a] and [Rayner 96]. The background idea is that grammar rules tend to combine frequently in some particular ways. Given a sufficiently large parsed corpus, it is possible to identify the common combinations of grammar rules and chunk them into "macro-rules". The result is a "specialized" grammar, which has a larger number of rules, but a simpler structure, allowing it in practice to parse quickly. It gains speed in a trade-of for the coverage. In [Rayner 96], they reported 3 to 4 times speed up at the price of about 5% loss of coverage, using 15,000 training data in the Air Travel Planning (ATIS) domain. Their idea of the macro rules is very similar to the idea of the small non-terminal grammars described in this thesis. However, their experiments are done on a small and relatively homogeneous domain. The performance is not compatible, because they reported only the indirect performance produced by the parse output. [Samuelsson 94b] proposed entropy thresholds which automatically

derive the macro-rules, but no improvement over the previous work was found.

2.2.9 (Lexicalized) Tree Adjoining Grammar

The relevance of this research to this thesis is not the method of grammar acquisition, but its formalism. The formalism of Tree Adjoining Grammar (TAG) and Lexical Tree Adjoining Grammar (LTAG) were proposed by Joshi and Schabes [Joshi et al. 75] [Joshi 87]. [Schabes et al. 88] [Schabes 90]. The idea behind TAG formalism is very similar to the idea of the parser described in this thesis and LTAG is actually a motivation of the modification towards lexicalized probabilistic parser explained in Section 2.9. TAG is a generalization of context-free grammar, that comprises two sets of elemental structures: initial trees and auxiliary trees. (See Figure 2.2, in which initial and auxiliary trees are labeled using α and β, respectively. Note that these examples are LTAG rules, because each elementary tree has a lexical item.) An auxiliary tree has a nonterminal node on its frontier that matches the nonterminal symbol at its root. These elementary structures can be combined

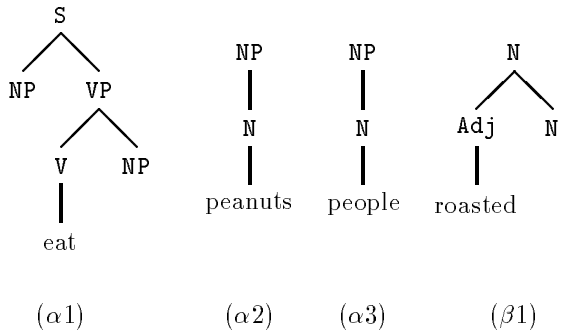

Figure 2.2: Examples of TAG elementary trees

using two operations, *substitution* and *adjunction*. The substitution operation corresponds to the rewriting of a symbol in a context-free derivation. For example, one could expand either NP in $\alpha 1$ by rewriting it as tree $\alpha 2$. The adjunction operation is a generalization of substitution that permits internal as well as frontier nodes to be expanded. One can adjoin $\beta 1$ into $\alpha 2$ at the node N.

A lexicalized tree adjoining (LTAG) grammar is a TAG in which each elementary structure (initial or auxiliary tree) has a lexical item on its frontier, known as an anchor. It can be rephrased that each lexical item has associated with it a set of structures that characterize the contexts within which that item can appear.

Several frameworks of probabilistic LTAG have been proposed [Resnik 92] [Schabes 92]. However, these seem to have difficulty to define the initial grammar rules. It is not straightforward to assign probabilities to the rules. The recent work by Joshi

[Joshi 94] tries to use n-gram statistics in order to find an elemental structure for each lexical item, which is called supertag. As a supertag contains structural information, assigning supertags for all the words in a sentence almost results in a parse of the sentence. The performance of a trigram supertagger was reported to be 90.0% on the WSJ corpus. Then a 'stapler' combines supertags to yield a parse tree. The performance was reported in terms of dependencies. Based on using 8,000 WSJ sentences for training and tested on 7,100 Brown corpus sentences, the accuracy of dependency relationships was 73.1% recall and 75.3% precision.

2.3 Initial Idea - Parsing by lookup

This section describes the initial idea which leads to the parsing described in this chapter.

Because of the existence of large syntactically-bracketed corpora and the advantage of context-sensitive parsing, we can contemplate a super-parsing strategy - parsing without parsing, or parsing by table look-up. This approach is based on the assumption that the corpus covers most of the possible syntactic structures for sentences. In other words, most of the time, you can find the structure of any given sentence in the corpus. If this assumption were correct, we could parse a sentence just by table look-up[1]. The idea is illustrated in Figure 2.3. The system first assigns parts-of-speech to an input sentence using a tagger, and then just searches for the same sequence of parts-of-speech in the corpus (or a tree dictionary). The structure of the matched sequence is the output of the system.

Now we have to see if the assumption is correct. The PennTreeBank is used for the investigation, which is one of the largest syntactically-bracketed corpora at the moment. However, it turned out that this strategy does not look practical. Out of 47,219 sentences in the corpus[2], only 2,232 sentences (4.7%) have exactly the same structure as another sentence in the corpus. This suggests that, if we apply the above strategy, we could find a match, and hence be able to produce a parse for only about 4.7% of the sentences in a new text. In other words, for 95.3% of input sentence, we can not make any output. A very rough estimation assuming Zipf's law [Zipf 49] [Zipf 65] in the distribution requires 10^{70} to 10^{140} sentences in order to get 80% of coverage.

[1] In this approach, ambiguities caused by lexical information, for example, prepositional attachment, are ignored.

[2] These are not the entire corpus but the 96% of the corpus which will be used for the grammar training. Also, minor modifications have been made, which will be described later.

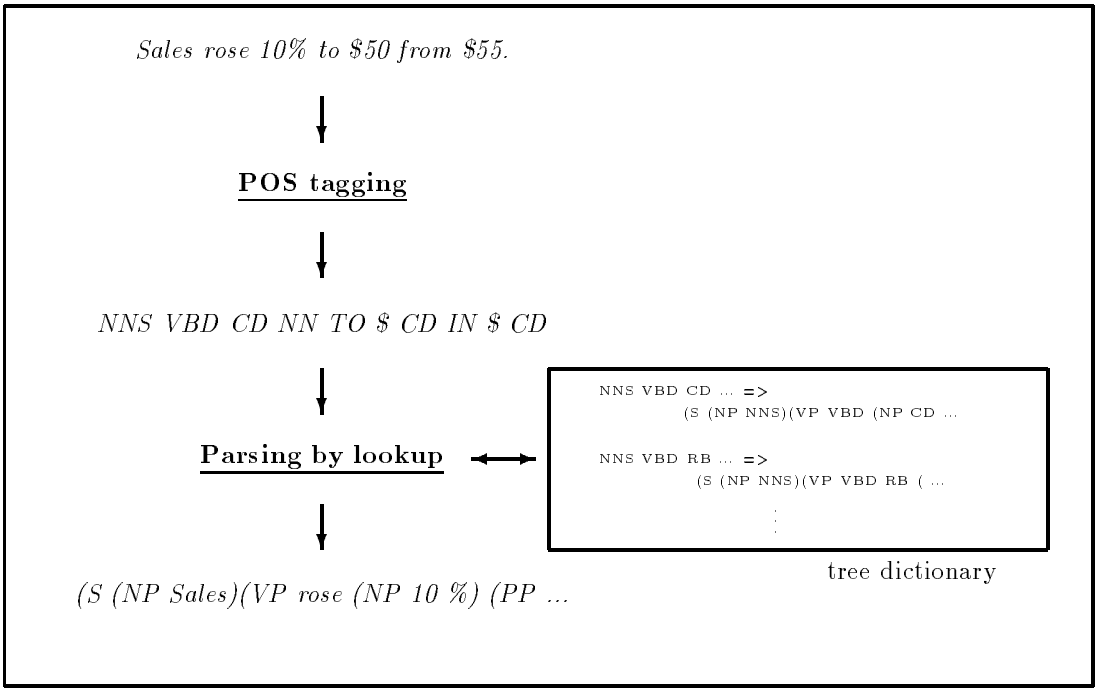

Figure 2.3: Parsing by table lookup

2.4 Two non-terminal grammar

2.4.1 Overview

Because of the observation described in the previous section, a compromise has to be made over the super-parsing idea. Instead of seeking a complete match for the part-of-speech sequence of an entire sentence, partial sequence matchings based on the two important non-terminals `S` (sentence) and `NP` (noun phrase), were introduced. The parser tries to find a nested set of `S` and `NP` fragments in the given sentence so that the whole sentence is derived from a single `S`. Then it applies the look-up strategy for each fragment. In other words, at first the system collects, for each instance of `S` and `NP` in the training corpus, its expansion into `S`'s, `NP`'s, and terminal categories; this is, in effect, a production in a grammar with non-terminals `S` and `NP`. It also records the full constituent structure for each instance. In analyzing a new sentence, it tries to find the best segmentation of the input into `S`'s and `NP`'s; it then outputs the combination of the structures of the chosen segments. To assure that this strategy is applicable, statistics are collected from the PennTreeBank (Table 2.3). From the table, we can see that there are a considerable number of multiple

Category	S	NP
Total instances	88,921	351,113
Distinct structures	24,465	9,711
Number of structures which cover 50% of instances	114	7
percentage of instances covered by structures of 2 or more occurrences	77.2%	98.1%
percentage of instances covered by top 10 structures	27.5%	57.9%

Table 2.3: Statistics of `S` and `NP` structures

occurrences of structures, unlike the case in the super-parsing strategy where only 4.7% of the instances have multiple occurrences. Also we can observe that a very small number of structures covers a large number of instances in the corpus. The most frequent structures for `S` and `NP` are shown below. The numbers on the left indicate their frequency. The definitions of the symbols are listed in Appendix A.

```
6483 (S NP (VP VBX NP))
4931 (S -NONE-)
4188 (S NP (VP VBX (SBAR S)))
1724 (S NP (VP VBG NP)
1549 (S S , CC S)
```

```
36470 (NP DT NNX)
34408 (NP NP (PP IN NP)
32641 (NP NNPX)
27432 (NP NNX)
17731 (NP PRP)
```

Although we see that many structures are covered by the data in the corpus, there could be ambiguities where two or more structures can be created from an identical part-of-speech sequence. For example, the prepositional attachment problem[3] leads to such ambiguities. A survey of the PennTreeBank shows that the maximum number of different structures for the same part-of-speech sequence is 7 for `S` and 12 for `NP`, and the percentage of the instances of `S` and `NP` with different structures are 7% and 12%, respectively. This is a little problematic, but by taking the most frequent structure for each ambiguous sequence, we can keep such mistakes to a minimum. For example, the prepositional phrase attachment problem is one of the cases of ambiguity. The parser always prefers noun attachment, because the frequency of noun attachment is more than that of verb attachment in the corpus. Some of these ambiguities should be resolved by introducing lexical or semantic information in the parsing[4]. This will be discussed in Section 2.8.

From these statistics, we can conclude that many structures of `S` and `NP` can be covered by the data in the PennTreeBank. This result supports the idea of parsing with two non-terminals, `S` and `NP` which segment the input, and the structure inside the segment is basically decided by table look-up. However, because non-terminals and hence segmentation ambiguities are introduced, the overall process becomes more like parsing rather than just table look-up.

2.4.2 Grammar

Guided by the considerations in the previous subsection, a grammar can be acquired from the PennTreeBank. The grammar has symbols `S` and `NP` as the only non-terminals, and the other non-terminals, or grammatical nodes, in the corpus are in effect embedded into the rules. In this thesis, the non-terminals in PennTreeBank which are not used as non-terminals in the grammar (for example, `PP` or `VP` in the two non-terminal grammar) will be called 'grammatical nodes', in order to distinguish them from non-terminals of the grammar. For example, the following is one of the extracted rules.

```
S -> NP VBX JJ CC VBX NP
  :structure "(S <1> (VP (VP <2> (ADJ <3>)) <4> (VP <5> <6>)))";
```

(where `S` and `NP` are non-terminals and the other symbols in the rules are terminals – part-of-speech tags of the PennTreeBank). By this rule, `S` is replaced by the sequence `NP VBX JJ CC VBX NP`, and in addition the rule creates a tree with grammatical nodes, three `VP`'s and one `ADJ`. When the parser uses the rule in parsing a sentence, it will generate the associated structure. For example, Figure 2.4 shows how the sentence `"This apple pie looks good and is a real treat"` is parsed. The first three words and the last three words in the sentence are parsed as usual, using

[3] Sentence like "I saw a girl with a telescope" has such problem. The prepositional phrase "with a telescope" can modify "a girl" (noun attachment) or "saw" (verb attachment).

[4] For example, sentence "I sold stock on Friday" should more likely be a verb attachment.

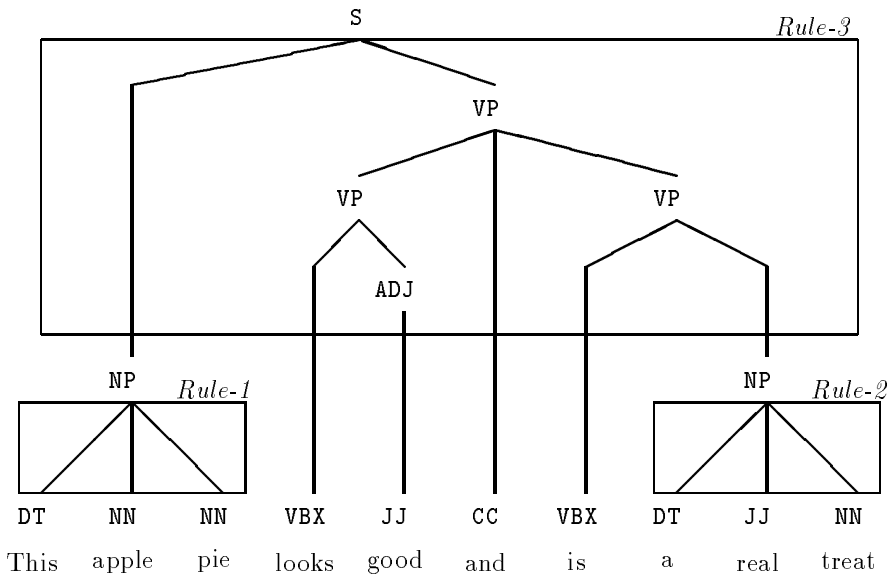

Figure 2.4: Example of parsed tree

the rules, `NP -> DT NN NN` (*Rule-1*) and `NP -> DT JJ NN` (*Rule-2*), respectively. The remainder is parsed by the rule, `S -> NP VBX JJ CC VBX NP` (*Rule-3*) alone. This rule constructs the structures under the root `S`. In short, the whole tree is generated based on the three rules, although there are more than three grammatical nodes or actual non-terminals in the tree.

2.4.3 Minor Modifications

Four kinds of minor modification are made to the grammar, in order to improve its coverage and accuracy. First, the punctuation mark at the end of each sentence is deleted. This is to keep consistency at the end of sentences, which sometimes have a period, another symbol or no punctuation in the PennTreeBank. Second, similar categories, in terms of grammatical behavior, are merged into a single category. This reduces the number of grammar rules and increases the coverage of the grammar. For example, present tense and past tense verbs play a similar role in determining grammatical structure. Third, sequences of particular categories are collapsed into single instances of the category. For example, sequences of numbers, proper nouns or symbols are replaced automatically by a single instance of number, proper noun or symbol. This modification also works to reduce the number of grammar rules. Finally, some new categories are introduced. This is because the PennTreeBank project tried to reduce the number of part-of-speech categories in order to ease the tagging effort. The PennTreeBank manual [Marcus 96] says that they combined categories, in cases where finer distinctions can be recovered based on lexical information. So, by introducing new categories for a set of words which have different behavior from the other words in the same category, we can expect to get more information and more accurate parses.

The following is the summary of modifications in the grammar:

1. Delete punctuation at the end of sentences

2. Merge Categories
   `VBX=(VBP, VBZ, VBD), NNPX=(NNP, NNPS), NNX=(NN, NNS)`

3. Collapse sequence into single instance
   `NNP, CD, FW, SYM`

4. Introduce new categories
   `@OF = of`;
   `@SNC =` Subordinating conjunction which introduces sentence (`although`, `because`, `if`, `once`, `that`, `though`, `unless`, `whether`, `while`);
   `@DLQ =` Pre-quantifier adverbs (`about`, `all`, `any`, `approximately`, `around`, `below`, `even`, `first`, `just`, `next`, `not`, `only`, `over`, `pretty`, `second`, `so`, `some`, `too`)

2.4.4 Tagging

As the first step in parsing a sentence, one or more part-of-speech tags are assigned to each input token, based on the tags assigned to the same token in the training corpus. This introduces tagging ambiguity. Each tag has a probability which will be used in the score calculation in parsing. The probability is based on the relative frequency of the tag assigned to the token in the corpus. The threshold for the probability is set to 5% in order to make the parser efficient. Tags with smaller probability than the threshold are discarded.

$$P_{tag}(t|w) = \frac{Frequency\ of\ word\ w\ with\ tag\ t}{Frequency\ of\ word\ w} \tag{2.8}$$

2.4.5 Score Calculation

The formulae for the probability of an individual rule P_{rule} and the score of a parsed tree S_{tree} will be defined in this subsection. The probability of a rule, $X- > Y$, where Y is a sequence, is based on the frequency with which X derives Y in the corpus, and the frequency of the non-terminal, X. The score for a parse tree is the product of probability of each rule used to build the tree together with square of the probability of the tag for each word in the sentence. The square factor results in putting more weight on tag-probability over rule-probability, which produce better results than assigning equal weights. The best parsed tree has the highest score among the trees possibly derived from the input.

$$P_{rule}(X- > Y) = \frac{Frequency\ with\ which\ X\ is\ expanded\ as\ Y}{Frequency\ of\ rules\ where\ LHS\ is\ X} \tag{2.9}$$

$$S_{tree}(T) = \prod_{R:\ rules\ in\ T} P_{rule}(R) \prod_{t:\ tags\ in\ T} (P_{tag}(t|w))^2 \tag{2.10}$$

2.4.6 Backup Grammar

Although the parser which will be described in Section 2.7 can handle a large grammar, it is unable to parse some long sentences, because of memory limitations. So a smaller grammar is prepared, for use in case the larger grammar can't parse a sentence. The small grammar consists of the rules having frequency more than 2 in the corpus. Because the number of rules is small, parsing is rarely blocked by memory limitations. The parsed result of this grammar is used only when the larger grammar does not produce a parse tree. Table 2.4 shows the numbers of rules in the larger grammar(G-0) and the smaller grammar(G-2). The number of rules in G-0 is smaller than the number of 'distinct structures' shown in Table 2.3, because if there are several structures associated with one sequence, only the most common structure is kept.

Category	Grammar-0	Grammar-2
S	23,386	2,478
NP	8,910	2,087
Total	32,296	4,565

Table 2.4: Number of rules

2.4.7 Experiment

For the parsing experiments, the WSJ corpus of the PennTreeBank is divided into two portions. 96% of it is used for training to extract grammar rules. The remaining part (1,989 sentences) is used for testing. The parsing results are shown in Table 2.5. Here, "G-0" is the parsed result using the grammar with all the produced rules,

Grammar	number of sentences	no parse	space exhausted	sentence length	run time (sec./sent.)
G-0	1989	20	293	19.9	13.6
G-2	1989	172	42	22.2	8.1

Table 2.5: Parsing statistics

"G-2" is the grammar with rules of frequency more than 2. "No parse" means the parser can't construct S for the input sentence, "space exhausted" means that the node space of the parser is exhausted in the middle of the parsing on 160MB machine. "Sentence length" is the average length of parsed sentences, and the run time is the parse time per sentence in seconds using a SPARC 5. Although the average run time is quite high, more than half of the sentences can be parsed in less than 3 seconds while a small number of long sentences take more than 60 seconds. The number of "no parse" sentences with G-2 is larger than that with G-0. This is because there are fewer grammar rules in G-2, so some of the sequences have no matching pattern in the grammar. It is natural that the number of sentences which exhaust memory is larger for G-0 than for G-2, because of the larger number of rules.

Next, the evaluation using Parseval method on parsed sentences is shown in Table 2.6. "Parseval" is the measurement of parsed result proposed by [Black et al. 91]. The result in the table is the result achieved by the G-0 grammar, supplemented by the result using the G-2 grammar, if the larger grammar can't generate a parse tree. These numbers are based only on the sentences which parsed (1,899 out of 1,989 sentences; in other words 90 sentences are left as unable-to-be-parsed sentences even using the back-up method). Here, "No-crossing" is the number of sentences which have no crossing brackets between the result and the corresponding tree in the Penn Tree Bank. "Average crossing" is the average number of crossings per sentence.

Total sentences	1899
No-crossing	643 (33.9%)
Average crossing	2.64
Parseval (recall)	73.43%
Parseval (precision)	72.61%

Table 2.6: Parsing evaluation

The performance is not as good as some state-of-art parsers described in Section 2.2. Note that they used lexical information, while the parser described in this section uses almost no lexical information. Compared to so-called traditional, or hand-made grammars, roughly speaking, the performance is similar or better. For example, Black [Black 93] cited the best non-crossing score using traditional grammars as 41% and the average of several systems as 22%.

2.5 Five Non-terminal Grammar

In the framework of the parser explained in the previous section, the two non-terminals were selected based on intuition. These are namely `S` and `NP`, which are generally regarded as the basic units in English. However, it is obvious that selecting the appropriate non-terminal set is one of the most crucial issues in the framework. Instead of the ones chosen by intuition, there could be an optimum set which gives the highest performance. This idea leads to the experiment of changing the non-terminal set, which will be described in this section.

This change may cause a trade-off between coverage and accuracy. If we use a small non-terminal set - the extreme case is only one non-terminal, `S` - it generally includes longer context, which would lead to better accuracy, but the coverage becomes very low. As is pointed out in Section 2.3, the coverage is 4.7% if only one non-terminal `top-S` is used. On the other hand, if we increase the number of non-terminals - the extreme case is to pick up all grammatical nodes defined in the PennTreeBank as non-terminals - then the coverage may be better, because the length of each rule becomes shorter. But accuracy may become lower, because we will lose context information.

In this section, experiments varying the non-terminal set in order to find an optimal set will be reported.

2.5.1 New Non-terminals

Two types of new non-terminals will be introduced.

One type is simply the grammatical nodes defined in the PennTreeBank. As described in Appendix A, there are 24 non-terminals in the PennTreeBank. In-

troducing some of these non-terminals may widen the coverage, but may lose the effect of context sensitivity, at the same time. In the experiment described here, five new non-terminals are individually added to the two basic non-terminals, S and NP. These five new non-terminals are VP, PP, ADJP, ADVP and SBAR. These are chosen because these are the major non-terminals in terms of frequency. For the sake of the explanation, names are given to the grammars, like 3-NT-(added-NT). For example, 3-NT-vp is the name of the grammar in which there are three non-terminals, S, NP and VP.

Also, for comparison, a grammar with all 24 non-terminals is created. It will be called all-NT.

The other type of new non-terminals are subclass of the current non-terminals. The idea is to categorize S and NP into different categories so that each split has different behavior. For example, in the PennTreeBank, S is used to label the top node of many sentences as well as portions of a sentence, such as a subordinate clause, a to infinitive or a quotation. It is conceivable that separating these types of S can achieve better performance, because the context of these may be different. In consequence, for instance, the new grammar would reject or assign low probability to a sentence like *which I like. In contrast, in the 2 non-terminal grammar (2NT), it might have high probability, because a rule "S -> WDT NP VBP" is a frequent construction in the corpus.

In the experiment, SS and TOINF are introduced for the non-terminal S. TOINF is introduced for the to-infinitive sentences. The PennTreeBank uses S for to-infinitive clauses, but obviously, these are different in behavior compared to the other S's. Also, SS is introduced for S other than top-S and TOINF. This aims to separate S's between S's for the entire sentence and the others.

With the similar idea, NPL is introduced. It is separated from NP by the following method. If the subtree of a NP does not contain NP among its descendents (not just as immediate children but as any descendents, as well), then it is defined as NPL, otherwise leave it as NP. (L stands for "lowest") The rationale of this is that simple noun phrases and complex noun phrases have different contexts in English. For example, the subject position prefers a simple noun phrase, and a noun phrase modified by a sentence clause may also be short. Note that because the grammar is a probabilistic grammar, it involves likelihoods rather than just restrictions. So it might be very important if split non-terminals have different probabilities for the rules. For example, assume there are rules S -> A with 0.01 probability, and SS -> A with 0.00001 probability. In the old grammar the rule for these grammar rules might be S -> A with 0.005 probability. This does not represent the real phenomena, which can be captured by the new grammar.

2.5.2 Experiment

Table 2.7 shows the parsing results using several different types of grammar. The second column shows the name which will be used to refer to this grammar in this

chapter. The third column shows the accuracy results, bracket recall and precision. The fourth column shows the number of fitted sentences which can not have a complete parse tree because of the memory limitation (160MB) or the grammar coverage. When this happens we generate a fitted parse tree by concatenating the most probable partial trees in order to make `S` at the top node. The issue of fitted parsing will be fully described in the next subsection. The quality of such parse trees is generally worse than a complete tree. The grammars are trained on the PennTreeBank sections 02 - 21, and tested on section 23. The accuracy measure is slightly different from the accuracy measure used in the previous section. In the previous section, the Parseval method [Black et al. 91] was used. In that measure, multiple layers of unary trees are collapsed to a single layer of unary tree. Also, the labeling is not considered in the matching[5]. However, the metric used in this section is the simple labeled bracketing precision and recall [Sekine and Collins 97]. From the experiments, the labeled bracketing metric generated about 1% to 3% worse result compared to the Parseval method.

Non-terminal	Name	Accuracy (recall/precision)	Number of fitted sentence
all grammatical nodes	all-NT	62.60 63.88	1137
S, NP	2-NT	71.89 72.92	123
S, NP, VP	3-NT-vp	70.51 72.94	241
S, NP, PP	3-NT-pp	68.64 69.53	574
S, NP, ADJP	3-NT-adjp	71.27 72.78	205
S, NP, ADVP	3-NT-advp	70.83 71.85	343
S, NP, SBAR	3-NT-sbar	70.92 73.03	192
S, NP, VP, PP	4-NT	67.04 68.44	706
S, NP, SS,	3-NT-ss	72.13 72.58	158
S, NP, NPL	3-NT-npl	74.20 73.51	15
S, NP, TOINF	3-NT-toinf	73.27 74.15	101
S, NP, SS, NPL, TOINF	5-NT	74.84 73.42	15

Table 2.7: Results on different non-terminal set

From the table, we can observe that the performance of the small non-terminal grammars is better than all-NL grammar. It is around 10% better in recall and precision, or 25% reduction in the error. This is a significant improvement.

Regarding the grammars with the new additional non-terminals, the results show that the introducing the additional non-terminals makes the accuracy worse (3-NT-vp etc.). This might mean the gain achieved by improving the coverage is outweighed

[5] Parseval is designed in order to compare different types of parsing output. The main objective of Parseval is to establish a standard metric for comparing different parsers based on different theories.

by the loss of context sensitivity and increased number of fitted sentences. It is also related to the size of training corpus, which will be discussed later.

Regarding the grammars with the split non-terminals, the table shows significant improvements. The grammar with three new split non-terminal (5-NT) gets 3% improvement in recall and about 1% in precision over the baseline two non-terminal grammar. At this range, 3% improvement is very important because the amount of possible improvement becomes relatively small. This result proves the assumption that these non-terminals are contextually different. In particular, the number of fitted sentences in `3NT-ss` and `3NT-toinf` are almost the same as that in `2NT`, indicating that the main improvements might come from context information. In `3NT-npl` and `5NT`, the number of fitted sentences are reduced, so it is not easy to determine if the improvement was due to the context information or the reduction in fitted sentences.

It is possible to extend the idea to other possible splits. Ultimately, there are a large number of possible splits and it is impossible to try all these grammars. We might need to use our intuition as a guidance to determine what splits might be useful.

Regarding the search space in the parsing, it can be seen from the table that introducing the split non-terminals, in particular `NPL`, reduces the search space, because the number of fitted sentences decreased significantly. The 5-NT grammar generates only 15 fitted sentences in contrast to 123 in the 2-NT grammar. This certainly helps to improve the accuracy, because the quality of a fitted parse is generally worse than completely parsed ones. Investigations in this area have to be done in future.

2.5.3 Fitted Parsing

A technique 'Fitted Parsing' was employed for these experiments. Fitted parsing is used when the parser can't produce a complete tree, because of memory limitation or lack of appropriate grammar rules. Fitted parsing combines partial trees or chunks so that the analysis of the entire sentence will be the list of chunks. Note that there could be a lot of possible combinations, but it chooses the most probable one for the output. The probability of an analysis is defined by the product of probabilities of all chunks. The categories of the chunks are limited to `S` and punctuation marks in the experiments. It is possible to add costs to the number of chunks in order to prefer a small number of large chunks rather than a large number of small chunks. However, this seems not necessary to improve the accuracy according to several trials. The details of the experiments are omitted here.

2.6 Size of Training Corpus

In this section, the issue of training corpus size will be discussed. This issue is very often discussed in corpus based methods, or data oriented learning in general [Russell and Norvig 95]. In order to assess the ability of the system, it is important to know the relationship between the training corpus size and the performance, or how much data is needed to achieve sufficient performance. Also, in this parsing framework, it is interesting to investigate the relationships depending on its grammar types (see Section 2.5 about the grammar types). If a grammar has a smaller number of non-terminals, it is likely to have a coverage problem at small training sizes, but once there is enough training data, then the accuracy may become better. On the other hand, if a grammar has a larger number of non-terminals, it is likely that there is less coverage problem even with the small training data, but the absolute accuracy may be worse with large training data compared to a grammar with a small number of non-terminals. Also, the relationship between the number of fitted sentences and the size of training corpus will be presented.

2.6.1 Experiment

In this experiment, the size of the training corpus ranges from 206 sentences to 47,219 sentences, approximately dividing the largest training corpus by two recursively. Then we conduct experiments using four types of grammar. In other words, four types of grammar are generated from a training corpus of eight different sizes. So, 32 parsing results are compared in terms of the training corpus size and the grammar type. The four grammar types are as follows.

- 2-NT (`S` and `NP`)
- 5-NT (`S`, `SS`, `NP`, `NPL` and `TOINF`)
- 4-NT (`S`, `NP`, `VP` and `PP`)
- all-NT

Before describing the experiments, several interesting statistics will be shown which were discovered during the preparation of the experiments.

Table 2.8 shows the average number of grammar rule instances generated per sentence in the largest training corpus. In other words, these numbers indicate how many grammar rules are needed in order to create a tree for a sentence. Comparing the number for 2-NT and all-NT, we find that about 55% of the grammatical nodes are `S` or `NP`, and 45% of the grammatical nodes are nodes other than `S` and `NP`, which are embedded in the long grammar rules in the 2-NT grammar. The number of instances in 2-NT is relatively larger than expected.

Figure 2.5 shows the relationship between the training size and the distinct number of grammar rules (number of types). From the figure, we can observe that even

Grammar type	Average number of grammar rule instance per a sentence
2-NT	10.3
5-NT	10.3
4-NT	16.4
all-NT	18.6

Table 2.8: Grammar types and number of instances per sentence

with 40,000 sentences, the curves are not saturating, although a very slight slowing down can be seen. This is not a good sign for the strategy, because it indicates that the coverage may not be saturated even if we increase the training corpus. The rate of growth ranges from about 0.21 rule per sentence (all-NT grammar) to 0.75 rules per sentence (5-NT grammar). So we may find a new rule in about five new sentences (all-NT grammar), or in almost every new sentence (5-NT grammar),

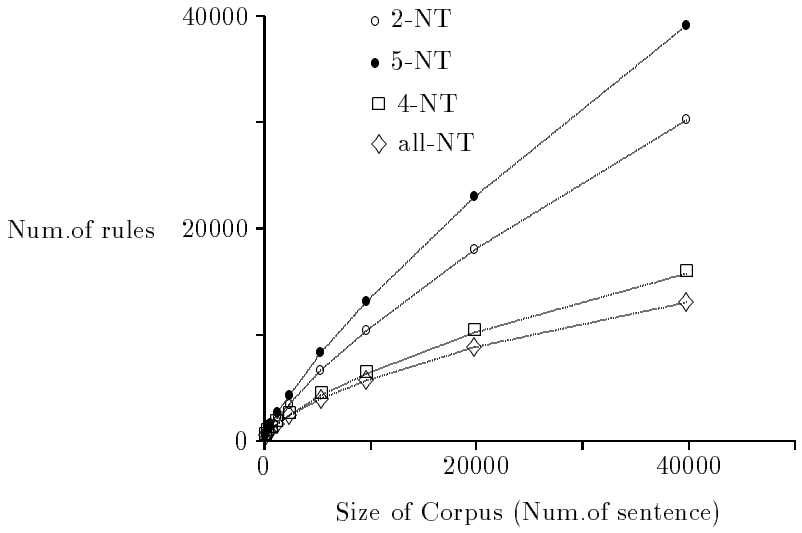

Figure 2.5: Corpus size and number of grammar

Now, the results of the parsing experiment using 32 different grammars are shown. Section 01 to 21 of PennTreeBank are used as the training and section 00 is uses as the testing. Figures 2.6 and 2.7 show the relationship between the size of training corpus and accuracy measures (recall and precision, respectively). Again, the labeled bracketing metric was used. All the curves generally show improvement in performance which saturates at certain levels. For example, the graphs of 2-NT grammar and 5-NT grammar are increasing at the small corpus size and reach the

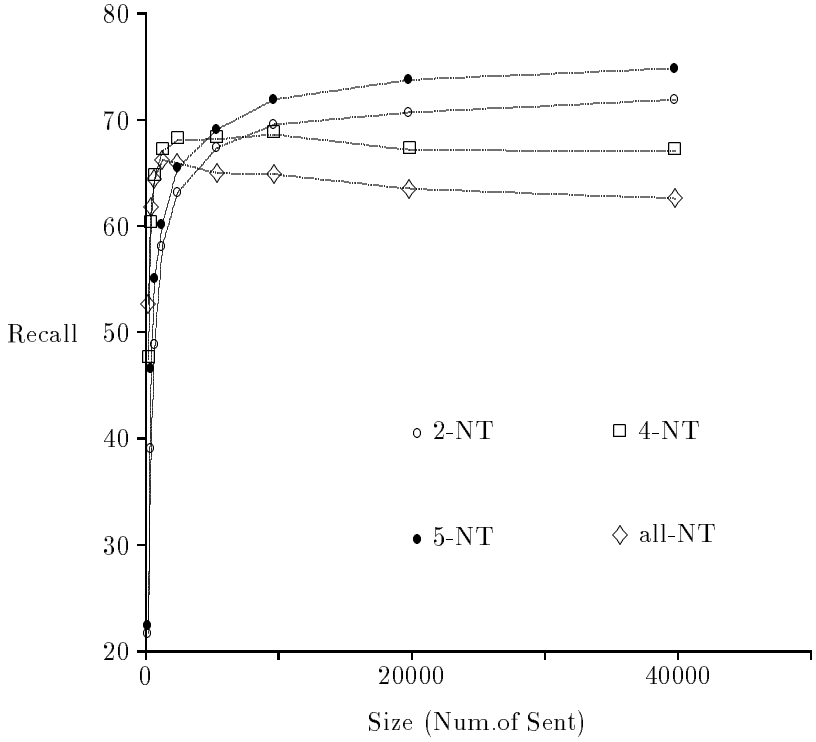

Figure 2.6: Size and recall

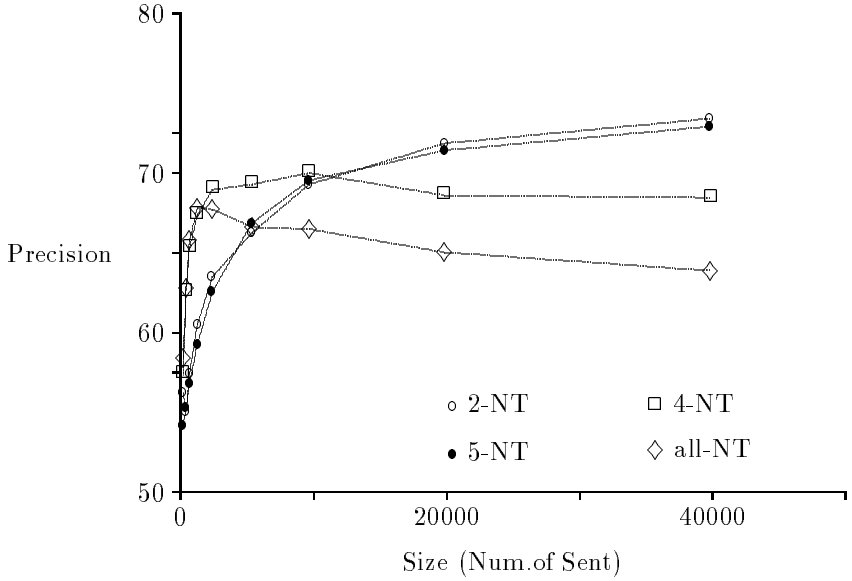

Figure 2.7: Size and precision

saturation points at around 6,000 sentences; the curves after this point are almost flat. The degradation of all-NT grammar is due to the memory size problem. 160MB machines are used in the experiment, but as the grammar has many short rules, they can fit many parts of a sentence. Then the number of active nodes in the chart parser grows rapidly, and the number of incomplete sentences also increases. Consequently, a lot of sentences can not have a complete tree for the entire sentences. Those sentences will be helped by the fitted parsing technique described in the previous subsection. For example, the number of sentences which use fitted parsing technique is 1,137 out of 1,921 test sentences with the largest all-NT grammar (Table 2.9). This issue will be discussed later.

Looking at Figure 2.6 and 2.7, it is surprising that the accuracy saturates at around 6,000 sentences with 2-NT and 5-NT grammars. This number is much smaller than common expectations. For all-NT and 4-NT grammars, the saturation point is much smaller, although the absolute accuracy is lower than that of the best two grammars. This is a very interesting and encouraging result for the corpus-based framework, because we don't need a very large training corpus in order to create a fairly good grammar. In particular, this can support the idea of domain dependent grammars. As the syntactic phenomena may differ in different domain, we may have to prepare corpus for each domain. The fact that a technique needs a small training corpus is critical in order to apply the technique to different domains. The issue of domain and the parser will be discussed in Chapter 4. Also the result that we don't need a large corpus supports the idea of applying the framework to other languages. Creating a large tree bank is not so easy; it is crucial that this approach needs only a small corpus in a new language. An application of the framework to Japanese will be reported in Section 2.10.

Table 2.9 shows the number of fitted parsing sentences for each of the 32 different grammars. For each grammar type, there is a minimum point of the number of fitted parsing sentences. This phenomena is caused by the combination of two sources of parse failure. These are the memory limitation and the grammar coverage. When the training data becomes larger, more grammar rules are extracted, and the parser creates more active nodes. Then there will be more sentences which can't complete a parse tree for a fixed memory size. On the other hand, when the training data is small, there is the coverage problem. As the grammar rules are extracted from a corpus, we may not have enough grammar rules to make a complete tree for an unseen sentence, even if the complete tree is not the correct tree. This is more likely to happen in the grammar with smaller training data, because the chance of having the rules needed to create a complete tree becomes smaller. This causes an increase in fitted sentences in the range of small training size. Figure 2.8 suggests how these causes combine to affect the number of fitted parsing sentences. Memory limitation causes fitted parsing at large training data (the right triangle), while grammar coverage causes the problem at small training data (the left triangle). The combination of these two triangles creates a minimum point somewhere in the

Size	39832	19840	9673	5376	2397	1261	681	404	206
2-NT	123	53	32	53	115	301	620	1008	1632
5-NT	15	8	30	81	238	466	619	921	1715
4-NT	706	614	361	237	94	76	118	240	656
all-NT	1137	1013	804	632	308	137	119	157	437

Table 2.9: Training size and number of fitted parses

middle, which can be seen from Table 2.9.

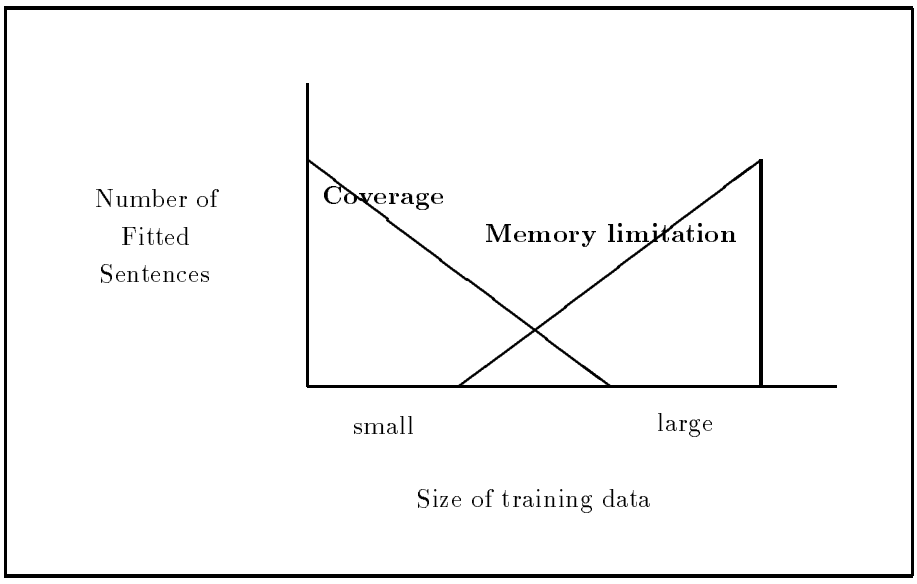

Figure 2.8: Image of two reasons for fitted parsing

2.7 Parsing Algorithm

In this section, the parsing algorithm is described. It is not easy to handle such a large grammar, i.e. a grammar with thirty or forty thousand rules. For example, a simple LR parser might need a huge table, while a simple chart parser might need a large number of active edges. A chart parser which can handle the large grammar was developed. There are three key techniques to overcome the difficulties. These are grammar factoring, best first search and Viterbi search algorithm. In this section the parsing algorithm, including these three key techniques, will be described.

2.7.1 Grammar Factoring

The first key technique is grammar factoring. The grammar rules are factored with common prefixes and can be represented in the form of a finite state automaton. In the system, it is implemented by a trie structure. Each arc is labeled with a grammar symbol of the right-hand side of grammar rules. Information about each grammar rule, including the left-hand-side symbol, score and structure, is stored at the node where the rule is completed. This could be a list if there is more than one rule for an identical sequence of symbols.

Now, an example of a grammar automaton is shown. The automaton shown in Figure 2.10 is generated by the grammar in Figure 2.9. Here, the number at the end of each rule indicates the score of the rule. In the figure of the automaton, the

```
S -> A B C : 30
S -> A B C D : 25
S -> A B F : 20
NP -> A B C : 40
NP -> A E : 25
```

Figure 2.9: Example of grammar rules

information in each node is the left-hand side symbol and score of each rule. For example, rule `S -> A B C D` is stored at the right-end node in the figure, which you can reach by the transitions `A B C` and `D`. Note that for the sequence of symbols `A B C`, there are two rules, for `S` and `NP`.

In practice, as the grammar has thousands of rules which start from the same terminal, for instance, `DT`, a simple chart parser has to have the same number of active edges when it finds a determiner in an input sentence. So usually, it needs an enormous number of active edges to parse even a short sentence. However, since active edges indicate grammar rules which can be extended after that point, we can replace the thousands of active edges by a single pointer to the corresponding node in the grammar automaton. Because a node in the automaton has arcs to all the possible successors, it is equivalent to having a number of active edges in a conventional chart parser.

Actually scores are stored slightly differently in the finite state automaton, in order to make search more efficient. This is related to the best-first-search algorithm explained in the next subsection, but the implementation is done at nodes in the finite state automaton.

Each node in the automaton has a partial grammar score so that the sum of the scores through the path of a grammar rule will be the score for the rule. This partial score is the minimum possible score for the rules following the node. Figure 2.11 shows the modified finite state automaton for the rules shown in Figure 2.9. In the figure, numbers in the upper half of nodes are partial scores for the purpose

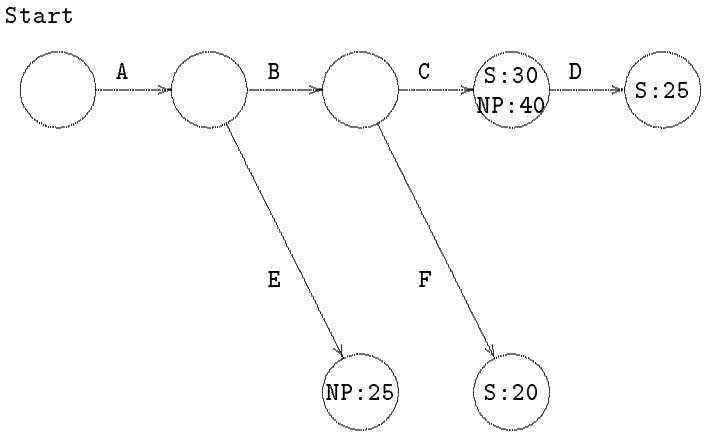

Figure 2.10: Automaton for the grammar rules

of the grammar score calculation. There is grammar information in the lower half, only when a grammar rule is completed at that node just as in the basic automaton. In addition to left-hand-side symbol of the rules, it has an additional score for the grammar rule which is completed at the node. The sum of the partial scores through the path of a rule is the score of the rule. For example, the score for rule `S -> A B C` is calculated as $20 + 0 + 0 + 5 + 5 = 30$. These partial scores are used in the best first search technique.

2.7.2 Best First Search

The second technique is the best first search. The parser has a heap, which keeps the active nodes to be expanded. The active nodes are stored in the order of their scores. So the top of the heap is the active node with the best score and which is the one to be expanded next. When a `top-S` is generated for the entire sentence, the parsing stops after checking if no other active node is capable to make a better tree. This best first search technique decreases the number of active nodes compared with that of a left-to-right style chart parser. The partial score in the grammar automaton explained in the previous subsection is used to have a better estimate of the score for each active node.

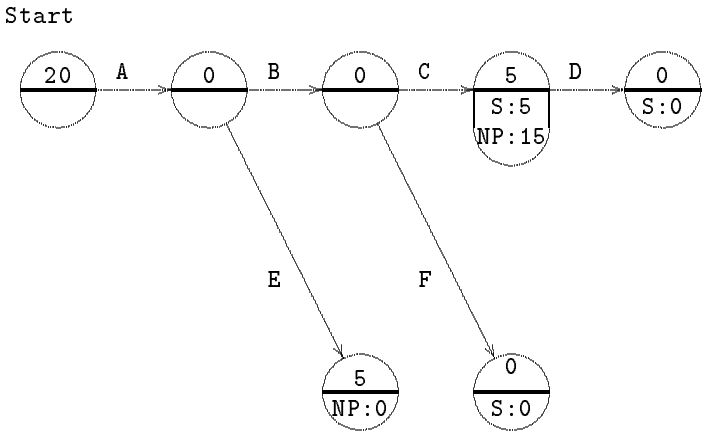

Figure 2.11: Modified automaton for the grammar rules

2.7.3 Viterbi Search

The third technique is the Viterbi search. Because we are looking for the best solution for each input sentence, we need only one complete category for each span. In other word, for each span, the only best candidate for each category has to be kept and the worse candidates of the same category can be discarded, as these will never be a part of the best solution.

This is a very efficient algorithm, if you are looking for only the best solution, because the parser creates fewer inactive nodes, and in consequence fewer active nodes.

2.7.4 Algorithm

In this subsection, the parsing algorithm is presented. The skeleton of the algorithm will be formally described at the end of this subsection; minor details are omitted. Brief explanations about heap operation and dictionary operation are presented.

The heap operations can be implemented by a binary tree so that the score of a node is always smaller than that of its children. The time complexity of heap operations is logarithmic in the number of data in the heap. (The reader who is interested may find details in [Cormen et.al 90]). There are four functions for the heap operations:

```
extract_heap()       : extract the best score ANode from heap
store_heap(p)        : store ANode p into heap
```

```
not_empty_heap()     : return 1 if heap is NOT empty
best_score_in_heap() : return the best score of ANode in heap
```

The words in the dictionary have part-of-speech and score information which will be used in the probability calculation along with grammar rule scores. The dictionary look-up is simplified in the algorithm, but in practice there are complicated issues like capitalized words, numbers, or unknown words.

In the algorithm, there are two structures, `ANode` and `Node`. `ANode` roughly corresponds to a set of active edges, and `Node` corresponds to an inactive edge in the standard chart parser algorithm. Both `ANode` and `Node` have their scores which are calculated based on the word scores and grammar scores underlying the node. In addition, `ANode` has a pointer to the grammar automaton which represents the portion of grammar rules already consumed. `Node` has its grammatical symbol and pointers to its child nodes. There is a variable to keep the best score for the entire sentence already discovered (`$best_score`). The score is basically minus the logarithm of the probability, so the smaller the score, the better the probability. Also there are variables, `$p` and `$q` for `ANode`, and `$n` and `$m` for `Node` structures. A function `score()` is used, which returns the score of the argument.

```
========== Parsing  Algorithm ================================
BEGIN

  $best_score := VERY_BIG;

  FOR all words in the sentence DO
    store_heap(ANode for each dictionary entry);
  END;

  WHILE not_empty_heap() and best_score_in_heap()<$best_score DO

    $p := extract_heap();

    IF there is a complete grammar rule at $p and score($p) +
       grammar score < score(Node at the same span and with the
       same category) THEN

      create new Node $n for $p;

      IF $n is a `S´ covering the entire sentence and
         score($n) < $best_score THEN

        $best_score := the score;
      ELSE

        store_heap(Anode for $n);

        FOR Anode $q which is finishing at the previous word to
          the span of $n and there is a rule which follows $n DO

          store_heap(Anode combining $q and $n);

        ENDFOR;
      ENDIF;
    ENDIF;

    FOR Node $m to which there is a grammar rule from $p DO
      store_heap(ANode combining $p and $m);
    ENDFOR;
  ENDWHILE;
END;
```

2.8 Integrating Lexical Dependency

So far, almost no lexical information was used. However, it is clear that lexical information is very important for parsing. For example, most prepositional attachment ambiguities can not be solved without lexical information[6]. Consider the following example:

```
<Input>   I sold stock on Friday.

<1> (S (NPL I) (VP sold (NP (NPL stock) (PP on (NPL Friday)))))
<2> (S (NPL I) (VP sold (NPL stock) (PP on (NPL Friday))))
```

It shows two analyses using the five non-terminal grammar. In the analysis in parse-1, the prepositional phrase '`on Friday`' modifies '`stock`' and these make a noun phrase. In the other analysis, the prepositional phrase modifies '`sold`'. Obviously the second analysis is generally appropriate. However, the parser creates the first analysis, because the combined probability of parse-1 is greater than that of parse-2, when only syntactical probabilities are used. Roughly speaking, in the prepositional attachment ambiguity, the prepositional phrase modifies the noun phrase more often than the verb. However, as is clear from the example, it depends on the words or lexical relationships. One idea for solving the problem is to use the frequency of the word pairs or dependencies in order to solve the ambiguity. For example, in the training corpus of the PennTreeBank, there are 11 instances of `on` modifying `sold`, but only 2 instances of `on` modifying `stock`. So this could be a clue to produce the correct answer for the ambiguity.

This is the motivation of the experiment described in this section. It might be more useful if we extend 'word pair' to 'word triplet', e.g. `sold-on-Friday` versus `stock-on-Friday`. Also it is possible to include information about the parent category etc., but as the initial trial, information on word pairs (lexical dependency) will be incorporated into the parser.

2.8.1 Model

The lexical dependencies are accumulated from the PennTreeBank. Since we will inevitably encounter a data sparse problem when we try to extract lexical information from the corpus, an interpolation model is used for smoothing. Namely, the probability of lexical dependency is calculated by Formula 2.11. Here, $C(w)$ is the count of word w appearing as a head in the corpus, $Pos(w)$ is the part-of-speech of word w and $C(a, b, d)$ is the number of appearances of the dependency relationship where a is the head and b is the argument and the dependency direction d, which is either left or right. The weight for each element is tuned to produce optimal result.

[6] Note that some of the ambiguities can not be solved even with lexical information; discourse or sublanguage information is needed.

$$
\begin{aligned}
P_{lex}(w,v,d) &= 0.97\frac{C(w,v,d)}{C(w)} \\
&\quad +0.01\frac{C(Pos(w),v,d)}{C(Pos(w))} \\
&\quad +0.01\frac{C(w,Pos(v),d)}{C(w)} + 0.01
\end{aligned} \tag{2.11}
$$

This probability is combined with the tree probability based on syntactic probabilities and lexical probabilities, explained in 2.4. Then the formula for the tree probability is

$$
P_{tree}(T) = P_{syntax}(T) \prod_{d:dependencies\ in\ T} P_{lex}(d) \tag{2.12}
$$

2.8.2 Finding the Head of a Constituent

In order to acquire lexical dependency relationships, we have to identify the head of constituents. The head is going to be the head of all dependencies to the other elements in the constituent. It is defined by heuristic rules introduced by [Magerman 95] and [Collins 96]. For example, in order to find the head of a prepositional phrase, the elements in the phrase are scanned from left to right, and the first preposition encountered in the scan is the head of the phrase. In some constituents, the scan may be from right to left, while in the other constituents, it is from left to right. The search is conducted using several categories; i.e. if there is no preposition, then foreign words or prepositional phrases are searched. If no item in the table is found, then the left or right-most item will be the head of the constituent. Appendix B shows the rules. This table is created based on the previous work by [Magerman 95] and [Collins 96].

2.8.3 Acquire Data from Corpus

From 96% of the PennTreeBank corpus (section 00, 01 and 03-24), 1,034,914 dependency relationships for 32,012 distinct head words are extracted. The dependency direction, i.e. if the argument is to the left or right of the head, is also recorded. Table 2.10 shows examples of dependencies for a singular noun 'impact' as a head. Only the dependencies with frequency greater than 1 are shown.

2.8.4 Experiment

An experiment was conducted on sentences in section 2 of the PennTreeBank, and the sentences are limited to those shorter than 40. A five non-terminal grammar is used as the basis. All the other parts of the PennTreeBank are used for the grammar acquisition and the lexical dependency extraction. The result is compared

Arg. on Left	Freq.	big	2
the	49	its	2
a	18	lasting	2
an	11	less	2
any	11	market	2
The	9	minor	2
much	6	profit	2
negative	6	ultimate	2
and	5	**Arg. on Right**	Freq.
“	4	on	58
adverse	4	of	40
financial	4	,	4
little	4	”	3
significant	4	from	3
long-term	3	as	2
,	2		

Table 2.10: Examples of dependencies for ‘impact’

with the result without lexical dependency. Table 2.11 shows the recall and precision

Experiment	Recall	Precision
Without lexical dependency	80.12	78.30
With lexical dependency	82.01	79.77

Table 2.11: Experiment about lexical dependency

of the parsing experiments based on the Parseval metric. We can see that the lexical dependency information improves about 2% in recall and 1.5% in precision. This is an encouraging result for pursuing this line of the experiment.

2.8.5 Future Direction

This is currently a very active research area in corpus based parsing. Recently, in [Collins 97] [Charniak 97] [Ratnaparkhi 97], very good parsing performances are reported using lexical dependencies on statistical parsers. The dependency information they used is relatively richer than that used in this experiment. One of the future directions is to implement the lexical dependency techniques described in these papers.

2.9 Lexicalized Grammar

This is another direction for improving the parser using lexical information. The idea was motivated by LTAG, which was explained in Section 2.2. The framework explained in this subsection tries to include lexical items directly into the current grammar rules, just like the modification from TAG to LTAG. For example, the following was found in the five non-terminal grammar. A very frequent rule `S -> NP VBX SS`, has two structures, `(S NP (VP VBX (SBAR SS)))` and `(S NP (VP VBX SS))`. Here the structure means a partial tree which should be created from the rule. In this case, there are two structures, because these two kinds of partial trees are found in the training corpus, both of which are actually frequent. The frequency of the first structure (let's call it 'structure-1') is 864 and the frequency of the second structure ('structure-2') is 138. An example sentence for structure-1 is `He said he is right` and an example sentence for structure-2 is `She helped students do better on the test`. For more investigation, instances which create the grammar were accumulated and the words used at the terminal slot (`VBX`) were analyzed. Table 2.12 shows the six most frequent verbs for each structure. It is surprising

WORD	Structure-1	Structure-2
said	642	1
says	114	0
say	40	0
reported	5	0
think	5	0
thinks	5	0
helped	0	11
saw	0	7
expects	0	7
is	1	6
make	0	5
sent	0	5
TOTAL	864	138

Table 2.12: Frequency of verbs in two structures

that, in structure-1, most of the verbs are 'say' verbs (796 out of 864; 92%) and the frequencies of these verbs in structure-2 are almost zero. So, it suggests that instead of having a grammar rule `S -> NP VBX SS` for structure-1, it might be a good idea to have a rule like `S -> NP` `` `say-verb' `` `SS`. There are two rationales for believing that this might improve the parsing accuracy. One is for the sentences with a 'say' verb. These sentences will be assigned that structure more certainly, and they will be assigned more accurate probabilities. The other is for the sentences with a non-'say'

verb. Because now structure-2 becomes the majority (only 68 instances for non-'say' verb in structure-1, compared to 138 instances for structure-2) and should receive more credit for the sentences with the verbs other than 'say' verbs. The parser should choose structure-2 as the output structure for the sequence. Consequently, these lexicalized rules should improve the parsing accuracy.

This framework could capture the structural preference which may not be captured by the lexical dependency strategy explained in the previous subsection. This method not just simply combines context sensitivity and lexical information but also includes lexical context sensitivity, as well.

2.9.1 Experiment

There is one important parameter in the lexicalized experiment, which is the frequency threshold. For example, it is possible to lexicalize all grammar rules, but then this may cause coverage problems. In the extreme case, if we make all rules lexicalized, then a sentence can be parsed only when the sequence of words is exactly matched by a combination of some rules. Also low frequency rules might be harmful, because of their peculiarity.

One of the strategies for avoiding these problem is to lexicalize rules with more than a certain frequency; in other words, we should set a frequency threshold to the lexicalized rules. Then the remaining instances which have the same structure are gathered and create a non-lexicalized grammar rule as a back-off. The probability of a rule is computed simply based on the frequency of the tree. In other words, any portion of the PennTreeBank is counted as an example of only one of these rules.

Threshold	Number of lexicalized rules	Recall	Precision	Fitted sentences
2	29018	76.21	74.33	22
3	15304	76.18	74.42	25
5	7626	76.19	74.50	29
10	3149	76.12	74.37	29
15	1841	76.05	74.42	28
20	1243	75.87	74.29	30
∞	0	74.84	73.42	15

Table 2.13: Results of lexicalized experiments

Six thresholds were selected and the results of the experiments are shown in Table 2.13. In this experiment, the labeled bracket recall/precision metric is used. From the table, we can see that there is more than 1% improvement in recall and precision from lexicalized rules.

Also, we can notice that there is a curve in the accuracy in Table 2.13. In terms of

the average of the recall and precision, the best result is obtained with a threshold of 5. If we set the threshold higher or lower, the result gets worse. This is the expected result, explained at the beginning of this subsection.

2.9.2 Future Direction

Actually, in structure-1, about 95% of the verbs are a kind of reporting verb (variations of 'say', 'report', 'announce', 'predict', 'suggest', 'claim' and so on). So one plausible idea is to build word clusters each of which has similar behavior. In order to build the cluster, there are two possibilities. One is to use an existing dictionary. For example, the COMLEX dictionary [Grishman et al. 94] has the class of "report-verb". We can use the verbs in this class for this particular instance. The other possibility is to use automatic clustering based on the data itself. This remains as future work.

2.10 Japanese Parser

It would be interesting to see if the same technique works in other languages. In particular, because it is reasonable to assume that the technique is more useful in strict-word-order-languages, like English, we want to investigate if it can work for free-word-order-languages, like Japanese or Thai. It has been well discussed that, in Japanese, lexical relationships are the most crucial information for analyzing syntactic structures, and there are probabilistic corpus based parser which use lexical information (for example, [Fujio and Matsumoto 97] [Shirai et al. 97]). However, there are several cases where context plays an important role, e.g sent-1 and sent-2 as follows:

```
sent-1) KARE-HA   HASHITTEIRU.
        he -subj   running
        (He is running.)
sent-2) KARE-HA   HASHITTEIRU  INU-WO   MITA.
        he -subj   running     dog -obj saw
        (He saw a running dog.)
```

In the first sentence, the subject `KARE` depends on `HASHITTEIRU`, however, in the second sentence, it depends on `MITA`. The difference is caused by the additional two words. In this section, a trial of acquiring a Japanese grammar which uses context information will be presented.

2.10.1 Corpus

In this experiment, a Japanese annotated corpus developed at Kyoto University [Kurohashi and Nagao 97] is used. It has part-of-speech (POS) tagging and dependency information. They used an automatic part-of-speech tagger, JUMAN

[Matsumoto et al. 97] and a dependency analyzer, KN Parser [Kurohashi 96] to build initial structures and then trained annotators corrected the results. The size of the corpus is about 10,000 sentences, which is about a fifth of PennTreeBank at the moment. Their target size is 200,000. Each segment (`BUNSETSU`) is identified along with its dependency information. It consists of the ID number of the target segment and the type of the dependency relation. There are three dependency types represented by letters, 'D' for normal dependency, 'P' for parallel and 'A' for appositive. We will use binary notation for dependency relations. (See Figure 2.13 for an example of the binary notation.) In the notation, a dependency relation can be found by the following procedure. Climb up the tree until it becomes the left child of a node for the first time. The head of the segment is the right-most leaf (segment) of the right child. The last segment of a sentence is never a dependent of any segment. It is always the head of the sentence.

Now, the evaluation metric of parsing in this experiment is defined. Since dependency is regarded as one of the best notations for expressing Japanese syntactic structures and also the corpus used it, the purpose of our experiment shall be to find the correct dependency relation (or target) for each fragment. The type of dependency is ignored in this experiment. There are two evaluation metrics. One is 'dependency accuracy'. It is the percentage of segments which are assigned the correct dependency targets. The other is 'sentence accuracy'. It is the percentage of sentences which are assigned the correct dependency relations for the entire sentence.

2.10.2 Simple (Stupid?) Idea

In this section, a simple but maybe stupid idea will be proposed. The emphasis is on utilizing the context information. It ignores all the content information of each segment and regards all segments as uniform. In other words, when we try to parse a sentence, we will use only the information of sentence length. For an input sentence of length N, we just search for the most frequent dependency pattern of length N in the training corpus. Here, dependency pattern means the set of dependency relations in the entire sentence.

This is a very simple algorithm to run. However, obviously it may not work, because sentences are ambiguous when given only that information. In order to see if this strategy works, the corpus was investigated in terms of dependency patterns and sentence length. Table 2.14 shows dependency patterns of length less than five and their frequencies in the entire corpus. Each dependency pattern is shown in the binary tree representation and each segment is shown by '`#`'. For example, the second instance (`(# (# #))`) means that both the first and the second segments depends on the third segment, while the third instance (`((# #) #)`) means the first segment depends on the second and the second segment depends on the third segment. It is clear that the simple strategy does not work well even for those short sentences because of the ambiguities. For example, for the sentences of the length three, if the system always returns `(# (# #))` pattern, the sentence accuracy would be about

Dependency pattern	Frequency
(# #)	276
(# (# #))	252
((# #) #)	242
((# #) (# #))	200
(# ((# #) #))	139
(((# #) #) #)	120
(# (# (# #)))	106
((# (# #)) #)	42

Table 2.14: Examples of dependency patterns

51% and dependency accuracy would be about 75.5%. For the sentences of the length four, if we take this simple strategy, sentence accuracy would be about 32% and dependency accuracy would be about 67%.

2.10.3 Cleverer Idea

So the ambiguity explained in the previous section has to be resolved. An obvious method is to introduce some information about each segment so that a sentence should not be represented by its length (N), but by a sequence of categories. It is well known that in Japanese, the last word of a segment contains significant information for identifying the correct dependency relation of that segment. For example, if we know that a segment ends with `FUKUJOSHI-NO`, a post-position semantically similar to 'of' in English, then it is likely that it depends on the next word. (Actually a survey in the later section shows that 95% of `NO` segments depend on the next segment in the corpus.) Also we found some interesting examples. In Japanese, `KAKUJOSHI-HA` and `WO` are very common post-positions, so we investigated the sentences of length three whose first segment ends with `HA` and second segment ends with `WO`. There are 12 instances of such sentences in the corpus and all 12 have the dependency structure `(# (# #))`; none of these has the structure `((# #) #)`. This is a good result for designing our parser. Remember, in the previous strategy, sentences of length three have fifty-fifty distribution between the two structures. However, now it can find a unique and complete solution, if a sentence is a type of "`HA - WO - any`".

In short, this is a cleverer idea for parsing. We should include some information about each segment. But now, how we can define the appropriate level of information in order to maximize the performance? It is clear that if we include too much information, we may face the coverage problem, because the finer the information

becomes, the less the chance that we can find a sequence matching an input sentence in the training data. For example, if we include all lexical information, the parser can produce an answer only when there is an exactly identical sentence in the corpus. On the other hand, if we have too little information, just like the simple strategy, we might have more ambiguity and then the accuracy becomes worse. The situation is illustrated in Figure 2.12.

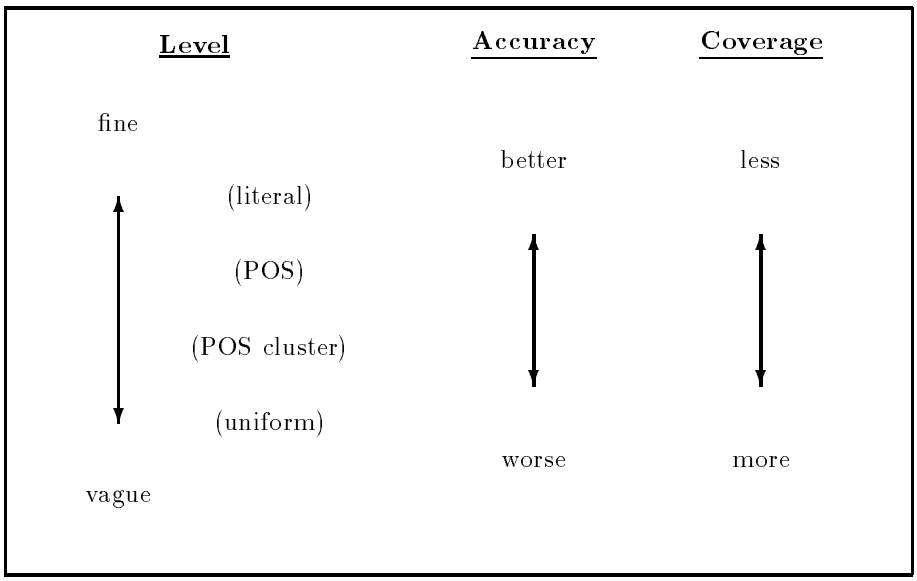

Figure 2.12: Trade-off between accuracy and coverage

The following scenario was tried to find the appropriate level of information. The basic idea is to start with a large number of initial categories, or parts-of-speech, then cluster or redefine these categories to maximize the performance of the parser. This procedure was not fully automatic: several steps involved human intervention and human intuition.

1. Decide on terminal units and non-terminals
   A segment (`BUNSETSU`) is used as a unit of analysis (terminal), as it is well known that the segment is a unit of syntactic structure and it is not that difficult to detect the correct segments automatically.

   Based on the experiments with the English parser, it is very important to define appropriate non-terminals. From these experiments, noun phrase and sentence group were considered to be the candidates for nonterminals. However, in Japanese, most noun phrases are enclosed in a segment, only the inner sentence is used as a non-terminal besides the top sentence category. It will be

called VP. and actually three types of VP are introduced, as will be explained later.

2. Choose initial categories
   Now, the set of initial categories for segments is defined. The categories are borrowed from the parts-of-speech defined in JUMAN or some lexical information of the last word of segments. The last word in each segment is used because it is understood as a good indicator of dependency relations. Note that the last word excludes symbols (i.e. comma (TOUTEN), bracket (KAKKO), quote (KAGI-KAKKO) or some other symbols); in such cases, the previous word is used for the moment. In a later stage, comma is taken into consideration for some cases as it provides useful information for parsing. The definition of the category (let's call it 'POS') came from the major classification definition in JUMAN for 13 categories, minor classification or a set of minor classifications for 6 categories, and 13 lexicalized categories and 2 other kinds of categories. These decisions were made by intuition and also by looking at the frequency of each category in the corpus.

3. Cluster categories
   Length three sentences in the training corpus were used for deciding on the category clusters. Iterative clustering by exhaustive search at each iteration was conducted. At each iteration, it selects the pair of categories which, when merged, maximize the estimated performance. This process was repeated until the performance starts declining. Here, the performance was measured by the score derived by the following method. It counts up the number of instances of the most common pattern for each sequence. For example, in a sequence `A B C` (`A`, `B` and `C` are categories), let's assume there are 8 instances of `(A (B C))` and 2 instances of `((A B) C)`. Then the system adds the number of instances of `(A (B C))` to the score. Notice that, in this example, we will eventually get 8 and lose 2 in the evaluation. The idea behind this is the probabilistic parser, which selects the most common patterns for a sequence. Assume one instance is removed from the corpus and the remainder are used for training, then the removed instance is parsed using the most common pattern in the training. The score approximately estimates the accuracy of that parse. If there is only one instance for a particular sequence, we give 0.5 accuracy, because we can't judge it without training data, because it can't be its own training data. This is a coverage problem. Using 490 sentences of length 3 in the corpus, the method mentioned above gave a score of 355.5 for the initial category set. However, when a new category is created by combining KAKUJOSHI-GA and FUKUJOSHI-HA, the score became 365, mainly because of increasing coverage. This pair produced the greatest increase among all the category pairs. After repeating this process 10 times[7], a score of 425 was achieved, which is locally

[7] Here, some human judgment was used, too. Some best pairs were not used if these are regarded

maximal. Note this may not be the global maximum because we used the iterative hill climbing technique.

4. Identify POS which depend on the next segment
   By the method above, good performance can be obtained for the sentences of length three, but it might not be optimal for the longer sentences. There is the coverage problem, because the longer the sentences, the less training data we have. For each POS, its frequency and the percentage of segments which depend on the next segment are computed. We found that several POS's almost always depend on the next segment. For example, the POS and the percentage of depending next segment are `SHIJISHI`, 92%, `SETSUZOKUJOSHI-NO`, 95%, and `RENTAISHI` 96%. As these are very likely to depend on the next segment, if an input sentence for the parser has one of the three categories, then the parser automatically makes it depend on the next segment. The combined segment has the category of the last segment (e.g. `SETSUZOKUJOSHI-NO JOSHI-HA` => `JOSHI-HA`).

   Also, there are several POS's whose frequency is very low. In such cases, having the POS in the grammar might become very harmful. The rule generated with a rare POS may work very strongly for a sentence with that POS, and it might be the case that the parser always generates structure using such rules. In order to avoid the problem, such POS's were eliminated. In parsing, if the input has such a POS, the parser makes the segment automatically depend on the next segment. Then the combined category has the category of the last segment of the sequence. Accordingly both methods help to increased the coverage and improved the accuracy.

5. Redefine categories by human intuition
   At this point, there are about 15 POS clusters. When looking at the result, our intuition is that all the categories of `JOSHI` have to be separated. In Japanese, `JOSHI` is a key element in parsing and different `JOSHI`'s have different behavior. For example, `FUKUJOSHI-HA` and `KAKUJOSHI-GA` behave in a similar manner in short sentences, but in terms of dependency, a segment with `HA` is likely to depend on a segment at longer distance than that for `GA`. So, all `JOSHI` categories are separated.

   Also, two sub-categories are added to the verb class, which are `RENTAI` (forms modifying nouns) and `RENYOU` (forms modifying verbs or adjectives), as the behaviors of these segments are different from each other and different from the rest of verb class. A clustering analysis is performed on the verb categories so that they are divided into several clusters based on their behavior. Two types of information were used in this analysis, distance to the dependency target and class of the target category. Three clusters are achieved, which

as accidental

roughly represent RENTAI (DOUSHI-TA), RENYOU (DOUSHI-TE) and the others (DOUSHI-HOKA). As was explained before, the nonterminal VP is also divided into three non-terminals based on the type of verb.

Comma TOUTEN is important information for parsing. All categories except KAKUJOSHI categories are divided into two categories, with and without comma. This modification improved the performance of the parsing.

6. 34 POS
   Some other minor modifications were performed, and finally, there were 34 POS categories. Each class will be represented by the representative POS in the category.

KAKUJOSHI-KARA	FUKUJOSHI-HA	SAHEN-MEISHI
KAKUJOSHI-GA	FUKUJOSHI-HA-TOUTEN	SAHEN-MEISHI-TOUTEN
KAKUJOSHI-DE	FUKUJOSHI-HOKA	MEISHI-HOKA
KAKUJOSHI-TO	FUKUJOSHI-HOKA-TOUTEN	MEISHI-HOKA-TOUTEN
KAKUJOSHI-NI	DOUSHI-TA	FUKUSHI
KAKUJOSHI-NO	DOUSHI-TA-TOUTEN	FUKUSHI-TOUTEN
KAKUJOSHI-HE	DOUSHI-TE	HANTEISHI
KAKUJOSHI-MADE	DOUSHI-TE-TOUTEN	HANTEISHI-TOUTEN
KAKUJOSHI-YORI	DOUSHI-HOKA	SETSUZOKUSHI
KAKUJOSHI-YO	DOUSHI-HOKA-TOUTEN	SETSUZOKUSHI-TOUTEN
KAKUJOSHI-WO	KEIYOUSHI	
SETSUZOKUSHI-NO	KEIYOUSHI-TOUTEN	

2.10.4 Grammar and Parser

In this subsection, examples of the acquired grammar rules and a parse tree will be shown. In order to simplify the explanation, probabilities are omitted. Three examples of the grammar rules are:

```
Rule-1) S -> VP-1
Rule-2) VP-1 -> VP-1 JOSHI-WO JOSHI-HA DOUSHI-TA
              :struct "((# #)(# #))"
Rule-3) VP-1 -> KEIYOUSHI JOSHI-GA DOUSHI-TA
              :struct "((# #) #)"
```

As was mentioned in the introduction, the idea of the grammar is context sensitivity. For example, by Rule-2, a combination of three dependencies (elementary binary trees) are generated at once. The first element, VP-1 (inner sentence; ending with a base form verb), depends on the second element, JOSHI-WO. The second element, JOSHI-WO depends on the last element, DOUSHI-TA. Also, the third element, JOSHI-HA depends on the last element. Using these three rules, Rule-1, 2 and 3, the tree shown in Figure 2.13 can be generated.

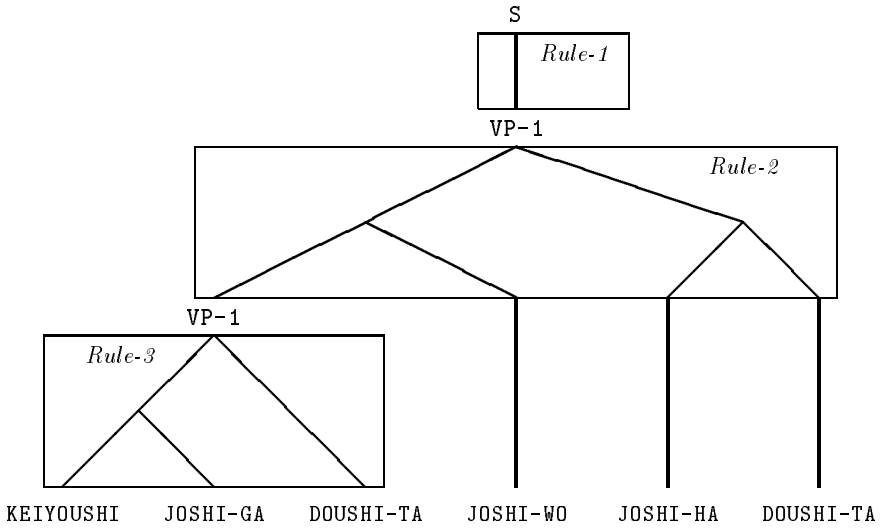

Figure 2.13: Example tree

The parser tries to find a combination of fragments where the top node `S` spans the entire input and the product of the probabilities of all the rules becomes maximal. In parsing, the same parsing engine used in the English version was used. Only a few modifications were needed: deleting the English specific morphological analysis and modifying the display component to print Japanese characters.

2.10.5 Experiment

The experiment was conducted using about 9500 sentences for training and the remaining 1246 sentences for the test. In this experiment, the input consisted of hand tagged POS segments. The number of rules we acquired from the training corpus are shown in Table 2.15. Here, `S` rules are the rules whose LHS are `S`, top node sentence. Also, `VP-1` for verbs (base form and `TA`), `VP-2` for verbs (modifying `YOUGEN`, `TE` and conditional) and `VP-3` for other verbs. On average, a sentence is parsed in 0.07 second in real time on a SPARC station-20 and the process size is 64MB.

The performance is compared with the 'default strategy'. In that strategy, all segments except the last segment depend on the next segment. This is a simple but fairly good strategy as many segments in Japanese are very likely to depend on the next segment. (The result of the experiment shows that two thirds of the segments in the corpus depend on the next segment.) The dependency accuracy classified by

Rules	type	instance
S	1849	9479
VP-1	6484	15143
VP-2	1827	5658
VP-3	827	1206
Total	10987	31486

Table 2.15: Number of rules

sentence length is shown in Table 2.16. The average accuracy of the parser was

Length	Freq	default	parser	Length	Freq	default	parser
2	(40)	**100.00**	**100.00**	18	(23)	62.40	**62.66**
3	(49)	69.39	**89.80**	19	(20)	59.17	**62.78**
4	(72)	64.81	**86.11**	20	(10)	63.68	**65.79**
5	(107)	65.42	**77.57**	21	(14)	64.64	**69.29**
6	(88)	65.68	**78.18**	22	(11)	63.20	**67.53**
7	(102)	64.38	**73.86**	23	(9)	62.12	**62.63**
8	(80)	64.82	**70.36**	24	(8)	60.87	**62.50**
9	(99)	64.02	**71.46**	25	(8)	63.02	**64.06**
10	(123)	65.13	**71.18**	26	(3)	**65.33**	58.67
11	(83)	63.73	**65.54**	27	(3)	**71.79**	64.10
12	(60)	64.09	**70.45**	28	(5)	64.44	**68.15**
13	(49)	64.29	**70.07**	29	(1)	**64.29**	50.00
14	(52)	62.87	**67.31**	31	(1)	53.33	**56.67**
15	(52)	64.84	**69.92**	37	(1)	**69.44**	63.89
16	(37)	62.52	**67.93**	38	(1)	**62.16**	59.46
17	(28)	63.17	**66.52**	41	(1)	**62.50**	57.50
						64.09	**69.64**

Table 2.16: Dependency accuracy

69.64% which is about 5.5% better than that of the default strategy. From the table, we can observe that the result is relatively better for short sentences. The accuracy for the sentences of length less than 10, which is the average length, is 75.39%, whereas the accuracy for the sentences of length longer than 9 is 67.36%. Comparing the result with that of the default strategy, the difference are larger in short sentence. The issue of coverage might cause this tendency. In the training corpus, there are a plenty of short sentences and it might be reasonable to assume that the variation of dependency structures for short sentences is largely covered. However, because the number of long sentences is smaller and the number of variations for long sentences is larger, the variations for long sentences are less covered by the training corpus.

Also, for parsing long sentences, because the rules acquired from short sentences have high probabilities, the rules acquired from long sentences may be blocked by these rules.

The sentence accuracy is shown in Table 2.17. Obviously getting all the depen-

Strategy	Number of sent.	Longest sent.
default	79/1246(6.34%)	6
parser	242/1246(19.42%)	15

Table 2.17: Sentence accuracy

dencies correct is a very difficult task, but the result of the parser (19.42%) is quite good. About one out of five sentences gets the correct dependency relations for the entire sentence, while the default strategy gets only 6.34%. Most of them are generated when it finds exactly the same pattern in the training corpus. Although more investigation is needed, this result suggests that if we have more data we may be able to achieve better performance, because there might be more of a chance to find the exact match in the training corpus. This assumption is supported by another observation. The dependency accuracy of about 90% was achieved for the sentences in the training corpus. This shows both the potential and the limitation of the method. When 90% was achieved, each sentence in this test set has the exact match in the training corpus. So, if we have a very large training corpus, the dependency accuracy could be improved from 69.64% to close to 90%, because it is much more likely to find the exact match in the training corpus. Note that we may not be able to achieve 90% even with an unlimited size corpus, because of increasing ambiguities. The limitation, the other side of the coin, is that even if we have a corpus of unlimited size, the dependency accuracy may have a ceiling of 90%. The remaining 10% was caused by ambiguities which can not be solved by syntactic context information. Also, the combining of rules using probabilities may be harmful to the goal of producing correct analyses for long sentences. A combination of short but quite common subsequences could have a better probability than a rare frequency, long and exact match sequence. Some techniques which are not used in the current parser are needed. One of them and the most important issue might be lexical information. It will be discussed in the next subsection.

2.10.6 Discussion

Although the parser shows good performance, in particular for short sentences, it is obvious that there is room to improve. First, let's start by considering what was achieved. The input in our algorithm is a POS sequence, and does not contain any lexical information. In other words, it is like a trial of parsing for the following sequence (Here, `PP` means a post position, `verb-noun` means a verb of a noun

modification form, and `verb-base` is a base form verb.):

```
JOSHI-HA KEIYOUSHI-TOUTEN SETSUZOKUJOSHI-NO JOSHI-WO KEIYOUSHI
 PP(HA)     Adj-comma          of              PP(WO)     Adj

   DOUSHI-TE SETSUZOKUJOSHI-NO JOSHI-TO JOSHI-DE DOUSHI-TA
   verb-noun        of          PP(TO)   PP(DE)  verb-base
```

The author believes that unless we have very good luck, we cannot parse the sentence correctly. A test was conducted by four linguists or computational linguists, two of them made 6 correct relations, one made 7 correct relations and the other made 8 correct relations among the 9 dependency relations. However, once we know the sentence is[8]:

```
[MURAYAMA SHUNOU HA] [YOUKA,] [CHOUTOUHA NO]
  ["SHINTOU JUNBIKAI" WO] [HOSSOKUSASETA] [SHAKAITOU NO]
  [YAMAHANA KAICHOU TO] [SHUNOUKANTEI DE] [KAIDAN SHITA]
```

all of the examinees can solve all the dependency relations. This indicates that lexical information is crucial in order to find the correct analysis. It is a future work to include this information in the parser.

2.11 Application to Speech Recognition

In this section, an application of the parser to continuous speech recognition will be described. The input for a continuous speech recognition system is speech, like reading newspaper articles or radio broadcast news. The system tries to output a transcription of the input. The techniques used in these system can be divided into two categories; acoustics and language modeling. An acoustic component takes the speech as its input and usually outputs the graph of candidate words (lattice) to a language component along with scores. The language component calculates the likelihood (score) of each word sequence and decides on the most plausible output based on acoustic and language scores. Acoustic techniques are irrelevant to this thesis; only the language component will be discussed. The most popular technique for the language model is the n-gram model. It assigns a score to a word based on the previous $n-1$ words. Most of the current systems use $n = 3$. The idea behind this is that we can guess the likelihood of words based on the previous two words. For example, the word '`right`' is more likely than '`write`' if the previous two words are '`you are`'.

However, this technique has an apparent limitation. It cannot take longer dependency or syntax into account. Consider if the previous example is in the following context (note that to find a comma is not easy in the speech recognition task):

[8] The sentence is a bit modified to make it short

```
I have not seen an author as brilliant as you are
                [right/write] to our magazine please
```

Here, we need syntactic knowledge to know that '`write`' is more appropriate than '`right`'. This is the motivation for the experiments in this section.

2.11.1 Structure

SRI's speech recognition system was used as the source of transcription hypotheses [Sankar et al. 96]. The scheme of the experiment is shown in Figure 2.14. The n-best sentences, the most likely top n candidate sentences for an utterance, is the input data for the parser. The parser assigns its own score for each sentence. The scores produced by SRI's acoustic and language models are linearly combined with the parsing score. Then the hypothesis with the highest total score is selected as the final output.

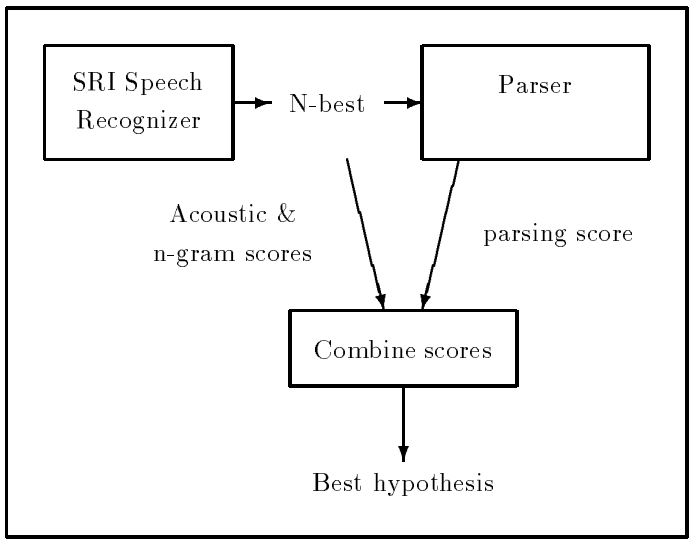

Figure 2.14: Structure of the system

2.11.2 Binary Comparison

First, in order to assess the ability of the parsing technique in speech recognition, a 'binary comparison' experiment was conducted. From the N-best sentences, the best candidate based on SRI's acoustic and language model scores (which will be called 'SRI-best'), and the correct sentence ('correct') are manually extracted. Both of the sentences for each utterance are parsed and the scores are compared. The

difference in the parsing score is compared with the difference in the trigram score as shown in Table 2.18. Table 2.19 shows the breakdown of the numbers based on cross comparison. In the tables, only those sentences where the correct sentence is in SRI 's N-best and the correct sentence is not SRI's best sentence are reported. The hope is that the parser will consistently prefer the correct sentence over SRI's best, and indeed in 60% of the cases the correct sentence had the better parsing score. Some

Parser	favors correct	39	(60%)
	favors SRI-best	26	(40%)
trigram	favors correct	25	(38%)
	favors SRI-best	40	(62%)

Table 2.18: Binary comparison between parser and trigram

	trigram favors correct sent.	trigram favors SRI-best	total
Parser favors correct	16	23	39
Parser favors SRI-best	9	17	26
total	25	40	65

Table 2.19: Break down of the comparison

sentence pairs along the scores are listed in Appendix C. In the remainder of this subsection, the category of the result will be indicated by using the position in Table 2.19 (i.e. top-right or bottom-left). In the bottom-left category — examples which are not desirable for the parsing model — some bugs in the grammar were found, as well as some inevitable cases. In these cases, local evidence is more important than wide syntactic context. For example, in the third pair of sentences in Appendix C, the parser prefers `the parent company shareholders` rather than `the parent company's shareholders`. This is because the part-of-speech sequence `DT NN NN NNS` is more likely than `DT NN NN POS NNS` (here, `DT`=determiner, `NN`=singular noun, `NNS`=plural noun and POS=possessive). However, if you look at the words, the correct sentence is at least as plausible as the other hypothesis (as the trigram model predicted). Several instances of this kind can be found in the bottom-left category.

By looking at the 23 instances in the top-right category — where the parser predicted correctly while the trigram model did not — a number of encouraging examples can be found. Six examples are listed in Appendix C. For example, in the first sentence, starting with `macdonnell`, SRI's best candidate has no verb, yet the trigram score for the candidate is better than that for the correct sentence. In the

second sentence `they say...`, there are too many verbs in SRI's best candidate. This is exactly the result to be expected by introducing a parser. In other words, in some cases, wide context is more important for picking the correct words than local (trigram) context .

The other categories (16 top-left and 17 bottom-right in the table) are harmless; adding parsing score to trigram score in these cases does not affect the ranking of the two sentences. Many such cases are to be expected because syntactic context often includes local evidence.

Outside of this table, an interesting example was found. It concerns out-of-vocabulary (OOV) words (in particular, proper nouns). An example is shown in Appendix C under "other" category. It contains an OOV sequence of long proper nouns ("noriyuki matsushima"), but as these nouns are not in the vocabulary, the speech system produced an unusual sequence of words ("nora you keep matsui shima"). The trigram score for the correct hypothesis can not be calculated, but the parser assigned a much better score to the correct sentence. So, it may be interesting for future work to use the parsing technique in order to identify these mistakes on OOV words.

2.11.3 Evaluation

Although some promising evidence was found in the binary comparison experiment, no improvement in speech evaluation was found when the parsing scores were linearly combined with the other sentence scores. This is understandable, because now there are 19 competitors (20-best was used) rather than a single competitor in the binary experiment; there could be some other hypothesis which is syntactically more plausible but includes more word errors.

It is a future work to conclude the usefulness of the parsing technique to this application. Research to find a method which utilizes the technique should be conducted.

Chapter 3

Sublanguage

3.1 Introduction

There have been a number of theoretical studies devoted to the notion of sublanguage. Most of them claim that the notion is important in processing natural language text, owning to lexical, syntactic or semantic restrictions, etc. A number of these studies have analyzed actual texts to try to verify the claim [Kittredge 82] [Grishman and Kittredge 86] [Slocum 86] [Biber 93]. Some of these studies and the definition of 'sublanguage' will be discussed in Section 3.2.

Several successful natural language processing systems have explicitly or implicitly addressed the sublanguage restrictions. For example, TAUM-METEO [Isabelle 84] is a machine translation system in which only sentences in the weather forecast domain are dealt with. It works remarkably well and has been used commercially. It is believed that the system was successful because of the limitation of the task, including the size of vocabulary and grammar. This is the benefit of the sublanguage notion and we would like to generalize the success of TAUM-METEO. However, it might be unrealistic to build such system for each domain by hand. One way to solve the problem is to find the means for automatic or semi-automatic linguistic knowledge acquisition from a limited domain.

Owing to the appearance of large machine-readable corpora, there are now new opportunities to address the issue. Section 3.4 reviews one such corpus, the Brown corpus. The explosion of large corpora has lead to a flowering of research on linguistic knowledge acquisition from corpora [CL Special Issue I 93] [CL Special Issue II 93] ; some were introduced in the previous chapter. Among them, several studies have mentioned the importance of the sublanguage notion, for example [Grishman and Sterling 92] [Sekine 92]. Although these are still small experiments in terms of coverage, they addressed an important problem of knowledge acquisition for sublanguage. The experiments reported in this chapter encourage the idea and the author believes pursuing this line of research could lead to a breakthrough for future NLP systems.

There are several other problems concerning the notion of sublanguage. One of them is the automatic definition and dynamic identification of the sublanguage of a text. In conventional sublanguage NLP systems, the domain of the system was defined manually. For example, the sublanguages of "weather forecasts", "medical reports" or "computer manuals" are, in a sense, artificially or intuitively defined by humans. This is actually one method to define the sublanguage of the text, and it often work well. However, it is not always possible and sometimes it may be wrong. For example, in a large vocabulary speech recognition system, which may have to handle a variety of sublanguages, it is impossible to tune it to each sublanguage in advance. Also, even if we take a human-defined domain as a sublanguage, the domain might contain a mixture of linguistic phenomena. For example, the "computer manual" domain could range from an "introduction of word processor for novices" to a "UNIX reference manual". In short, it might be a good idea to define sublanguage so that it will maximize the performance of particular application. Furthermore, even if we can define sublanguage for a system, we have to have a means to identify the sublanguage to which the text belongs. Section 3.3 shows experiments along these lines.

Another interesting problem of sublanguage is syntactic variation. Because of the unavailability of syntactically annotated corpora until recently, there had been little research on this issue. In those days, some easily detectable syntactic features were used in the investigations, e.g. relative clause frequency, active-passive ratio etc, but there was not a global and objective measure. Syntactic variations are very important, since they can affect the performance of a parser, the most important component in most NLP systems. If syntactic variations are very wide across sublanguages, we may have to prepare different grammars for different sublanguages. Section 3.5 and 3.6 demonstrate some syntactic variations among different domains. The results motivated the experiments described in the next chapter.

Finally, applications of the sublanguage notion will be described in Section 3.7 and in the next chapter. In Section 3.7, a topic coherence model for continuous speech recognition (CSR) systems will be presented. Most of the current CSR systems employ the trigram model as their language model, which only depends on the previous two words to assign probabilities to the current word candidates. Obviously, this model is not sufficient, because a text has a coherency of word usage based on the topic of the text. The topic is sometimes important for choosing the correct word from multiple candidates. The experiment reported in that section shows that the idea is actually useful and it improved the accuracy of a large vocabulary CSR system.

3.2 Related Work

The studies in this field can be divided into two categories. One is 'qualitative study' of sublanguage. Many such studies have been motivated by linguistic sur-

veys or studies of texts in a few domains based on human introspection. This type of study started early on. However, because of the availability of corpora and improving computer power, 'quantitative study' has emerged. It tries to measure the features of sublanguage by quantitative means. Some studies were motivated by some application systems, e.g. text categorization, text clustering or information extraction. A few studies of each category are summarized in this section. The first two subsections are related to the 'qualitative studies', although the notion 'inductive definition' explained in Section 3.2.1 is closely related to the 'quantitative studies' explained in Section 3.2.3.

3.2.1 Definition of Sublanguage

The definition of sublanguage has been well discussed in the last decades. We can find two kinds of definitions, although the difference has not received serious attention. One definition is inferable from Harris [Harris 68]:

> *Certain proper subsets of the sentences of a language may be closed under some or all of the operations defined in the language, and thus constitute a sublanguage of it.*

From this definition, we can infer that a sublanguage can be defined empirically by observing language behavior. The other type of definition is exemplified by the following citation from Bross, Shapio and Anderson [Bross et al. 72]:

> *Informally, we can define a sublanguage as the language used by a particular community of speakers, say, those concerned with a particular subject matter or those engaged in a specialized occupation.*

Let us call the former definition an *'inductive definition'*, since a sublanguage can be defined by observation of data. The latter definition, on the other hand, will be called a *'deductive definition'*, since it is based on the principle that a particular community of speakers would define a sublanguage.

Many practical projects have been using the deductive definition intentionally or unintentionally. For example, the TAUM project took two subject domains for their MT applications; one is weather forecasts and the other is aircraft maintenance manuals. This is a deductive definition of sublanguage, because they took a particular subject as their target. However, the deductive definition is not always possible and sometimes it may be wrong, as was discussed earlier. The inductive definition is related to quantitative analyses, since it is empirical. This will be described later.

3.2.2 Qualitative Analyses of Sublanguage

Lehrberger [Lehrberger 82] summarized qualitatively the characteristics of sublanguage into six categories. This analysis is also interesting in terms of quantitative

analyses. Since qualitative analyses and quantitative analyses are just like two sides of a coin, many of the characteristics in this analysis are actually the topics of quantitative analyses.

1. Limited subject matter

   In the deductive definition of sublanguage, this is a motivation rather than a characteristic. Limited subject matter leads to the characteristics enumerated in the following. An interesting question is how many sublanguages exist in a given language. This may be impossible to answer, because there are an almost infinite number of subject matters and some of them can not be determined a priori but emerge gradually through the use of a language in various fields, for example, "the language of sports-casting" or "the language in private e-mail exchanges on a particular topic".

2. Lexical, syntactic and semantic restrictions

   Lexical restrictions can be easily found by the observation of a sublanguage text. For example, in instructions for aircraft maintenance, there are 571 idioms, 443 of which are 'technical' idioms specific to the subject matter, like 'aspect ratio' or 'nose gear'. Also, in the corpus there are words which characterize the subject domain, like 'aileron', 'motor' etc, and other general words do not occur at all, like 'hope', 'think', 'I' or 'me'.

   In the aircraft maintenance instructions, the following types of sentences never occur; direct questions, tag questions, sentences with simple past tense, or exclamatory sentences.

   An example of a lexical semantic restriction is that, in aeronautics, the noun *dope* refers to a chemical compound used to coat fabrics employed in the construction of aircraft, whereas in pharmacology it may refer to narcotics. Also there are limitations of polysemy and case frame of verbs.

3. 'Deviant' rules of grammar

   This refers to rules describing sentences which, though quite normal in a given sublanguage, are considered ungrammatical in the standard language. It also refers to rules describing cooccurrence restrictions within a sublanguage that do not exist in the standard language.

4. High frequency of certain constructions

   Imperative sentences abound in a maintenance manual, because it is mainly concerned with instructing the users in the performance of certain actions.

In the aircraft maintenance instructions, there are many adjectives which never occur in predicate position. Examples of these are 'actual', 'chief', 'nickel-cadmium' and 'piston-type'. Also, the corpus contains many long strings of nouns or nouns and adjectives, like 'external hydraulic power ground test quick-disconnect fittings' or 'fan nozzle discharge static pressure water manometer'.

5. Text structure

For example, aircraft maintenance instructions are divided into numbered sections each of which deals with a specific part of the aircraft. This is important because these section divisions could resolve a polysemy of a word. For example, 'capacity' refers to volume in the hydraulic system and to farad in the electrical system or electronic equipment.

6. Use of special symbols

English texts in various fields share the alphabet a,b,c, ..., x, y, z, but '$\exists$' occurs in those dealing with mathematical logic, and 'æ' in phonology texts.

3.2.3 Quantitative Analysis of Sublanguage

Two studies in the field will be summarized in this subsection. There are many other such studies, for example in [Kittredge 82] [Grishman and Kittredge 86].

Slocum [Slocum 86] reported the experiments which tried to identify sublanguages on syntactic grounds. Four texts were prepared; two were extracted from operating and maintenance manuals, and the other two were sales brochures. These were parsed by their METAL parser. For each grammatical category, the following information was recorded: the number of applications of rules attempted by the parser, number of rules successfully applied and number of appearances of the phrase-type in `S`. Then, observing the data, they found two subclasses of texts, as desired, corresponding to the two categories. The brochure texts exhibited more syntactic phenomena than manual texts, although manual texts were not simply a subset of the other. The experiments suggest the possibilities of sublanguage identification and further the possibilities of sublanguage specific processing in order to maximize the performance of systems.

Biber [Biber 93] referred to the sublanguages as 'registers'. First, he compared the part-of-speech tags of grammatically ambiguous words between two different registers: fiction and exposition. Some examples are shown in Table 3.1. Similarly, comparison of probabilities for tagged sequences and percentages for prepositional phrases attached as noun modifiers and verb modifiers were reported to support the claim of different features between different registers.

word	tag	Fiction (%)	Exposition (%)
admitted	past tense	77	24
	passives	17	67
	perfects	6	0
	adjectives	0	9
trust	noun	18	85
	verb	82	15
until	preposition	19	38
	subordinator	81	62

Table 3.1: Comparison of tagging of words

Next, multidimensional differences among registers were observed in English. The study shows that there are systematic patterns (dimensions) of variation among registers. Each dimension comprised a set of linguistic features that co-occur frequently in texts. The dimensions were identified from quantitative analysis of the distribution of 67 linguistic features in corpora. Five major dimensions were identified by a factor analysis and interpreted as 1) Informational versus Involved production, 2) Narrative versus Non-narrative concerns, 3) Elaborated versus Situation-Dependent references, 4) Overt Expression of Persuasion, and 5) Abstract versus Non-abstract Style. In the plot of nine spoken and written registers in a two dimensional graph of 1) and 3), reasonable results were found. For example, 'Academic prose' was the extreme of informational and elaborated, whereas conversation was the extreme of involved and situated.

Experiments of automatic classification of new texts into the registers, based on a discriminant analysis using the five dimensions were reported. These were quite successful; the accuracy ranges from 62% to 93% depending on the conditions of the experiments.

3.3 Experiment with Perplexity

In this section, experiments of sublanguage definition and identification will be reported. As was discussed in the introduction section, it might be useful if we have some objective means to define sublanguage in order to make the sublanguage definition process automatic and to make the result better for some applications. Here, a method to find sublanguage clusters using perplexity is proposed. The result is evaluated by the experiment of sublanguage identification.

3.3.1 Overview

The experiments in this section can be divided into two parts: the definition of sublanguage and the identification of a new text to the sublanguage.

In the both experiments, a newspaper corpus (7.5MB of San Jose Mercury, 2147 articles) is used. Each article in the corpus is regarded as a unit of data, and a sublanguage will be formed by gathering similar units in term of word appearance. This is almost identical to the text clustering technique, which has been well studied in the field of information retrieval [Willett 88]. However the aim of the text clustering in information retrieval is slightly different from that in this thesis. They try to make text clusters which are useful for human information retrieval purposes, so the linguistic features of the clusters are not so important. In contrast, the purpose here is to find a sublanguage which is useful for NLP systems, so the linguistic features of clusters should receive more attention, although as an initial experiment, the evaluation was done on word frequency only.

3.3.2 Perplexity and Expected Perplexity

One of the problems in making clusters is how to define the number and size of clusters. The number of clusters can range from 1, with all the articles in the single cluster, to the total number of articles, with each cluster containing just one article. The problem here is that there is no objective means to decide the number and the size. This problem is the crucial issue for automatic sublanguage definition, because otherwise we need manual intervention or artificial thresholds. In order to explore this problem, the following statistics are proposed.

Perplexity, more precisely 'uni-gram perplexity' (PP) is a notion from Information Theory (see [Charniak 93] or [Krenn and Samuelsson 97]). It can be formally defined based on unigram Entropy H:

$$H = -\sum_{w} p(w) log_2(p(w)) \tag{3.1}$$

$$PP = 2^H \tag{3.2}$$

Here $p(w)$ is the probability of a token w in the source. This probability may be estimated by dividing the number of instances of the token by the number of tokens in the text. This is just an estimate, because the probability of a word in a language can be only 'estimated' from observation. From this estimated probability, we can get estimated entropy and perplexity. Roughly speaking, perplexity indicates the amount of information in the source. Under the condition that the sizes of two texts are the same in terms of number of tokens, then we can roughly say that the text with larger perplexity has the greater variety of tokens.

We can also calculate perplexity for a set of texts, treating the set as a single text. If two texts have a large number of overlapping tokens, which means the two texts are similar in terms of word distribution, the perplexity for the combined text

will be smaller than that of a text which has tokens chosen at random from the entire corpus. In short, if the perplexity for a text set is small in comparison to perplexity for a random text, we may say that the set has a sublanguage tendency.

In order to observe perplexities of texts, clusters are made based on similarities of articles. Clusters are grown from one initial article by adding similar articles in the order of their similarity to the initial article[1]. The definition for the similarity measure between article A_i and A_j is the following:

$$Similarity(A_i, A_j) = \frac{1}{|A_i||A_j|} \sum_{\{w|w \in A_i \wedge w \in A_j\}} \frac{1}{logDF(w)} \tag{3.3}$$

Here, $|A|$ is the number of distinct words in article A, $DF(w)$ is the document frequency, or the number of articles which contain word w. It uses the several thresholds. A cut off for document frequency 200 (tokens which have document frequencies more than 200 are not taken into account), minimum 2 occurrences in the article (only the tokens which occur 2 or more times in the article are considered), and minimum 2 tokens overlap (relationship would be established if the two article have 2 or more tokens overlapping). Those numbers were empirically determined. Each cluster grows in this way around each article in the corpus.

In order to make objective measurements, we want to compare these estimated perplexity values with the perplexity of random text. Because the estimated perplexity depends on the size of the sample text, we need to compare the cluster to a text of the same size obtained by selecting tokens at random from the entire corpus. The expected perplexity used in the experiment is the average of three of these 'random text' trials.

3.3.3 Observations on the Graph

Figure 3.1 shows four examples of the relationship between number of tokens in a cluster and the ratio of its perplexity to the random expected perplexity. The data points in the graph starting from the left to right indicate the data for the first article, the first article and the closest article combined, and so on. As the number of tokens becomes larger, the ratio moves toward 1.0, because the cluster is approaching the set of all articles combined. However, in all of the four examples, there is a minimum point at a small number of tokens. This could happen if the combining texts have a larger number of overlapping words than expected. Now, this minimum point suggests one of the definitions of sublanguage in terms of word distribution. This phenomena is observed not only for the four examples shown, but also for many of the other clusters. Observation of these articles shows a promising pattern. For example, in Example 2, all 9 articles from the beginning until the minimum point are obituaries, while the next three are real estate articles. In Example 1, the first article and 54% of articles up to the minimum point are college basketball articles. The

[1] This method is different from any of the standard clustering methods explained in Section 3.5

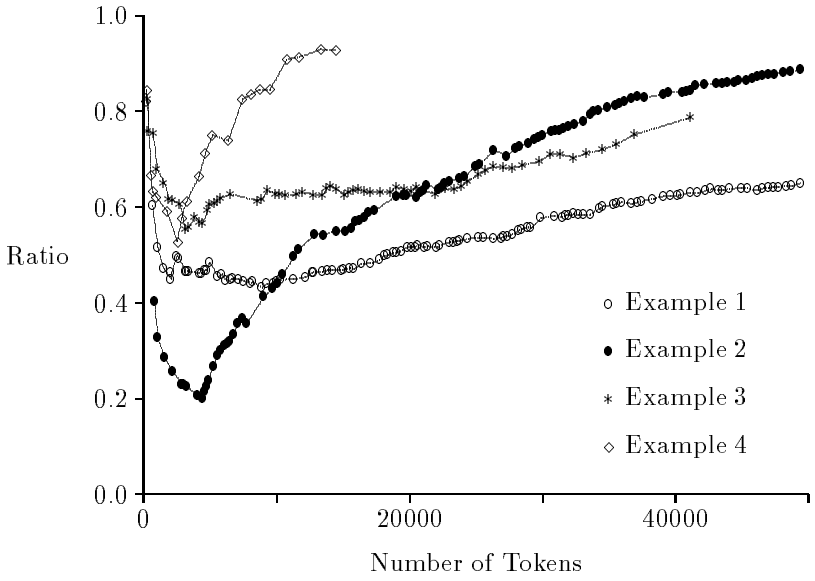

Figure 3.1: Ratio of perplexities

rest are either high-school and professional basketball (24% and 8%, respectively), hockey (8%) or football (4%) articles. On the other hand, only 16% of the next 25 articles and 6% of the rest of the articles up to the 185th are college basketball articles. These examples intuitively support the claim that the minimum point of each line at the graph would be an objective indicator of a sublanguage.

3.3.4 Sublanguage Clusters

Let's take the articles from the beginning until the minimum point as a sublanguage. These sublanguages are used in the experiment in the next subsection. As there are 2147 articles in the corpus and each cluster grows up from a single article, the same number of clusters exist (these are not disjoint and some of them are identical). The number of articles in a cluster ranges from one to 31, and the average number is 4.26 articles.

3.3.5 Sublanguage Identification for a New Text

In the experiment of sublanguage identification, 244 test articles are taken from the San Jose Mercury corpus. These are not used for sublanguage definition using the method described above. In this subsection, the method for identifying the closest cluster for each test article will be described.

For each test article, similarity measures are calculated for all the clusters to find the closest one. Basically, the similarity measure is the same as the one in the previous experiment. The similarity between a test article and a cluster is set equal

to the maximum similarity between the test article and any of the articles in the cluster. If there is a tie, the cluster which contains the smallest number of articles is chosen. This calculation is not expensive, because only a limited number of words are taken into account in the calculation, based on the document frequency cut off (50 in the experiment). The number of tokens to be examined in the calculation is normally much smaller than the number of tokens in the test articles. Also, the number of clusters to be examined for each word is less than the document frequency cut off. So this calculation is almost linear in the number of words in the test article, provided that there is enough space to store the word indices.

In this experiment, the parameters for similarity calculation were set slightly differently from the ones in the definition experiment. The document frequency cut off is 50 instead of 200 in the sublanguage definition experiment, minimum occurrence in an article is 1 and minimum token overlap is set to 3, based on empirical adjustments.

3.3.6 Result of Identification

In this subsection, the experiment of sublanguage identification is described. The evaluation measure is based on how many tokens in the test article also exist in the closest cluster, i.e. the number of tokens which overlap between the test article and the closest cluster. A straightforward measurement is its coverage, which is how many tokens are overlapping as compared to the number of tokens in the test article. This figure could be useful, but it is difficult to make a comparative observation because of lack of the normalization. The number of overlapping tokens is compared with the expected number of overlapping tokens. The expected number is computed by taking two sets randomly which have the same number of tokens as the test article and the selected cluster, then calculating the average number of overlapping tokens between them. This expected number (E) can be calculated by Formula 3.4.

$$E = \sum_{w} \frac{N_t \cdot f_w}{N} \left(1 - \left(1 - \frac{N_c}{N}\right)^{f_w}\right) \tag{3.4}$$

Here, N_t is the number of tokens in the test article, N_c is the number of tokens in the cluster, N is the number of tokens in the entire corpus, and f_w is the frequency of word w in the corpus. In the equation, the first factor is the expected frequency of word w in the article, the second factor is the probability that word w appears at least once in the cluster. The expected number of overlapping tokens can be classified into certain frequency ranges. Formula 3.5 defines $E(r)$, the expected number of overlapping tokens in the range of r, using $n_t(r)$, which is the number of tokens of the frequency range in the test article.

$$E(r) = n_t(r) \cdot \frac{\sum_{w \ in \ r} \frac{N_t \cdot f_w}{N} \left(1 - \left(1 - \frac{N_c}{N}\right)^{f_w}\right)}{\sum_{w \ in \ r} \frac{N_t * f_w}{N}} \tag{3.5}$$

High frequency words like "the" or "of" are relatively common regardless of the topic or sublanguage. On the other hand, low frequency words, which are often highly related to the topic, are expected to co-occur together in the same articles. So, we can anticipate in this evaluation that low frequency words overlap between test articles and the closest clusters more often than expected values. (Note that, because words with document frequency of less than 50 are used in the similarity calculation, those words must be specially considered in the evaluation.) To observe such details, the result is classified based on document frequency of words in the entire corpus. Table 3.2 shows an example of the results. In this example, the number of tokens in the article is 129 and the number of tokens in the cluster is 1265 and the cluster consists of 5 articles. The first column shows the number of

Document Frequency	Number of words	Overlap words	Coverage (%)	Ratio
1-49	20	6 (1.0)	30.0	6.01
50-99	11	5 (2.1)	45.5	2.40
100-199	25	15 (8.2)	60.0	1.82
200-299	10	9 (5.7)	90.0	1.57
300-499	10	7 (7.5)	70.0	0.94
500-999	12	12 (11.7)	100.0	1.03
1000-	41	41 (41.0)	100.0	1.00
Total	129	95 (67.6)	73.6	1.23

Table 3.2: An example of the result

tokens in the test article. The second column shows information about overlapping tokens. For example, 95 tokens out of 129 in the entire article are overlapping with the closest cluster. In the document frequency range from 100 to 199, 15 tokens are overlapping, while 8.2 tokens are expected to be overlapped. The third column shows the coverage of overlapping tokens in the test article. The overall coverage was 73.6% in this sample. The figure in the fourth column indicates that tokens in the article are overlapping 1.23 times the expected value. Also 6.01 and 2.40 times the expected number of overlapping tokens are found in the article in document frequency ranges from 1 to 49 and from 50 to 99, respectively. As mentioned before, the result for words whose document frequency ranges between 1 to 49 can not be evaluated directly by the figure. As the method to find the closest cluster is basically to find the closest article, so the words in the closest article are used in the cluster search. The number of tokens overlapping between the test article and the closest article in document frequency 1 to 50, i.e. number of tokens used in the similarity calculation, is 3 for this sample (not shown in the table). This means that out of 6 overlapping tokens between the test article and the closest cluster, 3 tokens are found in the closest article and the other 3 tokens are found in the rest of the articles

in the cluster. The additional 3 tokens as well as some of the overlapping tokens in the other ranges are the benefit of the clustering.

The average of the coverage and the ratio throughout the 236 test articles are shown in Table 3.3. Note that, 8 articles can't find their closest clusters because of the thresholds. The second column of the table shows the percentage of over-

Document Frequency	Coverage (%)	Ratio
1 - 49	16.7	7.98
50 - 99	26.4	1.94
100 - 199	34.5	1.90
200 - 299	45.9	1.03
300 - 499	57.7	1.19
500 - 999	76.6	1.00
1000 -	95.8	0.98
Total	51.1	1.18

Table 3.3: Average coverage and ratio

lapping tokens in the test articles, and the third column shows the average ratio of overlapping tokens to expected overlapping tokens. The table shows the success of the experiment. For example, tokens of document frequency range from 50 to 99 are found with 1.94 times the expected value in the closest cluster. Also the ratio in range from 100 to 199 is 1.90, which is well above 1.0. These results can be explained by saying that lower frequency words co-occur together more than the random expectation. On the other hand, ratio at higher frequency is about 1.0. It means the number of these words are almost same as the random expectation. This is understandable, because many high frequency words are closed class words and these words can occur regardless the topic. Actually, a close observation shows that almost all of the words of document frequency more than 1000 are closed class words, like "the", "of" or "it" and occur in many of the articles. There are only 68 words which have frequencies over 1000. (The total number of distinct words is 39217 and the total number of articles is 2147).

3.4 Brown Corpus

In the experiments in the remainder of the chapter and in the next chapter, the Brown corpus [Francis and Kucera 64/79] will be used. A brief review of the Brown corpus will be presented in this section. It is a syntactically tagged corpus consisting of several domains. One of the important features is that the way of tagging is uniform throughout the corpus. This is the crucial feature for the purpose of the sublanguage

studies because otherwise the different sublanguages could not be easily compared. Actually, the PennTreeBank version of the Brown corpus [Marcus 93] is used in the experiments.

The following description is quoted from the manual.

> This Standard Corpus of Present-Day Americana English consists of 1,014,312 words of running text of edited English prose printed in the United States during the calendar year 1961. So far as it has been possible to determine, the writers were native speakers of American English. Although all of the material first appeared in print in the year 1961, some of it was undoubtedly written earlier. However, no material known to be a second edition or reprint of earlier text has been included. The Corpus is divided into 500 samples of 2000+ words each. Each sample begins at the beginning of a sentence but not necessarily of a paragraph or other larger division, and each ends at the first sentence- ending after 2000 words. (In a few cases the count erroneously extended over this limit.) The samples represent a wide range of styles and varieties of prose. Verse was not included on the ground that it presents special linguistic problems different from those of prose. (Short verse passages quoted in prose samples are kept, however.) Drama was excluded as being the imaginative recreation of spoken discourse, rather than true written discourse. Fiction was included, but no samples were admitted which consisted of more than 50% dialogue. Samples were chosen for their representative quality rather than for any subjectively determined excellence.

The corpus consists of 15 domains as summarized in Figure 3.2 and fully shown in Appendix D. Each sample consists of about the same size of text in terms of the number of words (2000 words), although a part of the data is discarded because of format errors.

The top level sections shown in Figure 3.2 are used as the definition of domains. Each domain will be represented by the letter used for the section in the Brown corpus (e.g. `A` to `R`). It remains to be seen if the sections are sufficiently uniform and the best for the purpose of the sublanguage experiments. Remember the discussion in the introduction of this chapter. Some sections contain several different topics (see Appendix D); for example section `A` ranges from political or society reportage to cultural articles and section `J` is also comprised of several different fields. However, as an initial experiment, the sections "as is" are used for the experiments.

There are 24 parts-of-speech and 14 non-terminal symbols in the original PennTreeBank corpus. Besides the PennTreeBank classes, some new parts-of-speech and non-terminal symbols are introduced as was mentioned in the previous chapter.

I.	Informative Prose (374 samples)		
	A.	Press: Reportage	(44)
	B.	Press: Editorial	(27)
	C.	Press: Reviews	(17)
	D.	Religion	(17)
	E.	Skills and Hobbies	(36)
	F.	Popular Lore	(48)
	G.	Belles Letters, Bibliography, Memories, etc	(75)
	H.	Miscellaneous	(30)
	J.	Learned	(80)
II.	Imaginative Prose (126 Samples)		
	K.	General Fiction	(29)
	L.	Mystery and Detective Fiction	(24)
	M.	Science Fiction	(6)
	N.	Adventure and Western Fiction	(29)
	P.	Romance and Love Story	(29)
	R.	Humor	(9)

Figure 3.2: Categories of the Brown corpus

3.5 Comparison of Structure Distributions

In this and the next section, the characteristics of sublanguage in terms of syntactic structure are studied using the Brown corpus. In this section, a global analysis which compares distributions of syntactic structures across domains is reported. In the next section, a trial to find sublanguage specific structures is described.

3.5.1 Extract Subtrees

In order to represent the syntactic structure of each domain, the distribution of partial trees is used. A partial tree is a tree with a depth of one, in other words, a partial tree has only a root node and immediate leaves. For example, from the following parsed tree, 7 partial trees can be extracted. In each bracket, the first symbol is the root of a tree and the rest are children of the root.

```
(S (NP DT NN NN) (VP (VP VBZ (ADJ JJ)) CC (VP VBZ (NP DT JJ NN))))

  (S NP VP)
  (NP DT NN NN)
  (VP VP CC VP)
  (VP VBZ ADJ)
  (ADJ JJ)
```

```
(VP VBZ NP)
(NP DT JJ NN)
```

Partial trees are extracted from each section of the Brown corpus and the distribution of partial trees is computed based on its frequency divided by the total number of partial trees in each section. For example, the following are the top eight most frequent partial trees in domain `A` (Press: Reportage). The figure at right is the percentage of the partial tree in the section. There are several syntactic categories defined in the previous chapter.

```
(PP IN NP)      8.40 %
(NP NNPX)       5.42 %
(S S)           5.06 %
(S NP)          4.28 %
(NP DT NNX)     3.81 %
(PP OF NP)      3.54 %
(NP NNX)        2.84 %
(NP PRP)        2.79 %
```

Also, for the purpose of comparison, the distribution of lexical items is computed for each domain. For the lexicon, the same words with different parts-of-speech are distinguished. For example, "file (Noun)" and "file (Verb)" are different lexical items.

3.5.2 Compute Cross Entropy

'Cross Entropy' is a measure of how well a probabilistic model represents a test set. Let $P_T(i)$ be the probability of item i in the test set, $P_M(i)$ be the probability of i in the model, then the cross entropy of the test set to the model, $CE(T, M)$, is calculated by Formula 3.6.

$$CE(T, M) = -\sum_i P_T(i)\ log\ P_M(i) \tag{3.6}$$

The smaller the value, the more accurately the model represents the test set. If the model is identical to the test set, the value is equal to the entropy of the set itself (see [Charniak 93] for details).

For each pair of the distributions, cross entropy is computed by taking the distribution as the probability. Figure 3.3 shows the cross entropy of syntactic structure across domains. For example, 5.554 at column `H` and row `B` shows that the cross entropy of modeling by the distribution on `H` data and testing on `B` data is 5.55. As was explained in the previous section, the lower the cross entropy, the better the model is representing the distribution of the test data. So, the cross entropy to itself is always the best among the tests. Figure 3.4 shows the cross entropy of the lexicon across domains. These matrices themselves are not easy to observe. In the next subsection, analyses of these matrices using a clustering technique are described.

T\M	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R
A	5.13	5.35	5.37	5.41	5.41	5.37	5.42	5.52	5.45	5.51	5.52	5.46	5.53	5.55	5.43
B	5.47	5.19	5.47	5.43	5.50	5.46	5.49	5.55	5.51	5.55	5.58	5.47	5.60	5.60	5.44
C	5.51	5.49	5.16	5.48	5.55	5.52	5.53	5.65	5.59	5.57	5.65	5.48	5.65	5.63	5.49
D	5.48	5.39	5.42	5.12	5.45	5.39	5.45	5.58	5.48	5.49	5.52	5.39	5.54	5.54	5.38
E	5.50	5.48	5.51	5.47	5.20	5.44	5.53	5.62	5.48	5.58	5.59	5.51	5.58	5.61	5.50
F	5.31	5.29	5.32	5.26	5.29	5.10	5.30	5.48	5.31	5.36	5.37	5.32	5.38	5.40	5.30
G	5.34	5.30	5.31	5.27	5.34	5.29	5.12	5.47	5.36	5.35	5.41	5.31	5.41	5.37	5.31
H	5.56	5.44	5.57	5.56	5.58	5.61	5.62	5.09	5.53	5.82	5.88	5.69	5.89	5.89	5.67
J	5.39	5.37	5.40	5.35	5.35	5.34	5.40	5.46	5.15	5.52	5.57	5.48	5.58	5.59	5.45
K	5.32	5.25	5.26	5.24	5.31	5.25	5.27	5.55	5.41	4.95	5.14	5.13	5.15	5.17	5.13
L	5.32	5.26	5.32	5.27	5.32	5.23	5.30	5.64	5.45	5.12	4.91	5.13	5.09	5.13	5.11
M	5.59	5.47	5.47	5.42	5.55	5.52	5.55	5.75	5.74	5.41	5.39	4.98	5.41	5.36	5.31
N	5.29	5.25	5.29	5.26	5.28	5.22	5.28	5.61	5.43	5.10	5.06	5.11	4.89	5.12	5.10
P	5.43	5.36	5.40	5.35	5.40	5.36	5.33	5.68	5.55	5.23	5.21	5.18	5.21	5.00	5.22
R	5.57	5.46	5.50	5.43	5.56	5.49	5.59	5.79	5.69	5.43	5.41	5.34	5.44	5.47	5.07

Figure 3.3: Cross entropy of syntactic structure across domains

T\M	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R
A	7.42	8.32	8.44	8.57	8.68	8.57	8.73	8.57	7.85	8.69	8.78	8.38	8.84	8.84	8.47
B	8.19	7.19	8.17	8.16	8.37	8.33	8.38	8.29	8.50	8.44	8.56	8.11	8.57	8.59	8.20
C	8.59	8.40	7.24	8.43	8.62	8.57	8.65	8.68	8.80	8.61	8.73	8.28	8.81	8.78	8.30
D	8.27	7.97	8.06	6.85	8.23	8.11	8.04	8.21	8.36	8.27	8.44	7.90	8.40	8.32	7.99
E	8.65	8.45	8.48	8.44	7.34	8.50	8.67	8.54	8.70	8.72	8.80	8.36	8.77	8.77	8.41
F	8.42	8.27	8.27	8.24	8.36	7.32	8.36	8.53	8.60	8.45	8.56	8.14	8.55	8.53	8.18
G	8.33	8.11	8.13	7.96	8.28	8.15	7.27	8.35	8.40	8.30	8.42	8.08	8.43	8.38	8.09
H	8.20	8.06	8.30	8.24	8.26	8.36	8.43	6.97	8.17	8.73	8.79	8.26	8.85	8.87	8.40
J	8.42	8.22	8.24	8.23	8.32	8.37	8.42	8.15	8.25	8.61	8.68	8.23	8.74	8.82	8.35
K	8.32	8.18	8.17	8.23	8.45	8.31	8.35	8.76	8.68	7.00	7.89	7.76	7.94	7.96	7.88
L	8.37	8.23	8.25	8.34	8.51	8.37	8.42	8.82	8.80	7.85	6.85	7.75	7.91	7.95	7.91
M	8.58	8.36	8.34	8.26	8.68	8.53	8.76	8.81	8.98	8.17	8.18	6.67	8.16	8.26	7.98
N	8.50	8.38	8.40	8.39	8.57	8.39	8.57	8.89	8.98	7.95	7.96	7.80	6.98	8.00	7.97
P	8.39	8.26	8.24	8.21	8.47	8.31	8.36	8.90	8.90	7.84	7.86	7.77	7.92	6.88	7.89
R	8.58	8.35	8.26	8.27	8.59	8.44	8.63	8.80	8.96	8.21	8.27	7.90	8.31	8.31	6.87

Figure 3.4: Cross entropy of lexicon across domains

3.5.3 Clustering

First, a brief introduction of the standard clustering techniques using a distance matrix is mentioned [Gnanadesikan 77]. Given a set of items and a distance between each pair of items (not necessarily Euclidean distance), similar items are clustered based on the distance. Because there is no unique definition of grouping, the method of clustering is rather heuristic and several different methods have been proposed. The methods are largely divided into non-overlapping clustering methods and overlapping clustering methods. By non-overlapping methods, an item can belong to only one cluster, and by overlapping methods, an item can belong to several clusters. Also, there are several methods used to define distance between two clusters. Three major methods are 1) shortest distance, which takes the shortest distance among distance combinations between elements as the distance of the clusters, 2) average distance and 3) longest distance. The definition for 2) and 3) are analogue to the definition for 1). Here, non-overlapping clustering with average-distance clustering method is used.

Clustering makes it easier to understand the cross entropy data shown in the previous subsection. (Figures 3.3 and 3.4). The distance between two domains is calculated as the average of the two cross-entropies in both directions. Figure 3.5 shows the clustering result based on syntactic structure (Figure 3.3). Figure 3.6 shows the clustering result based on lexical data (Figure 3.4). From the results,

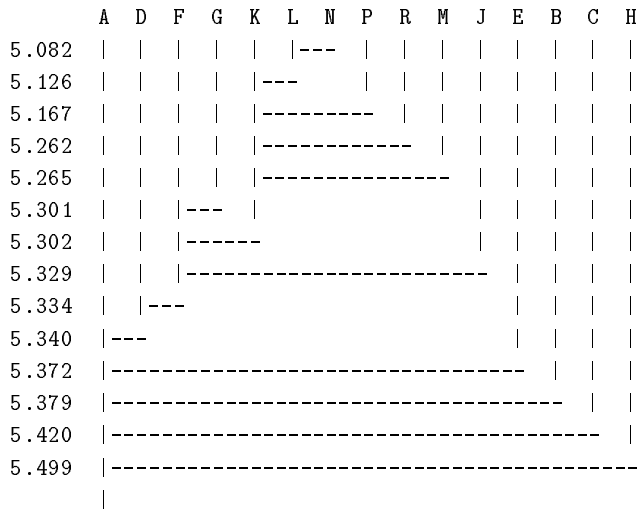

Figure 3.5: Clustering result based on grammar distances

we can clearly see that domains `K`, `L`, `N` and `P` are very close to each other. These are namely, "General Fiction", "Mystery and Detective Fiction", "Adventure and

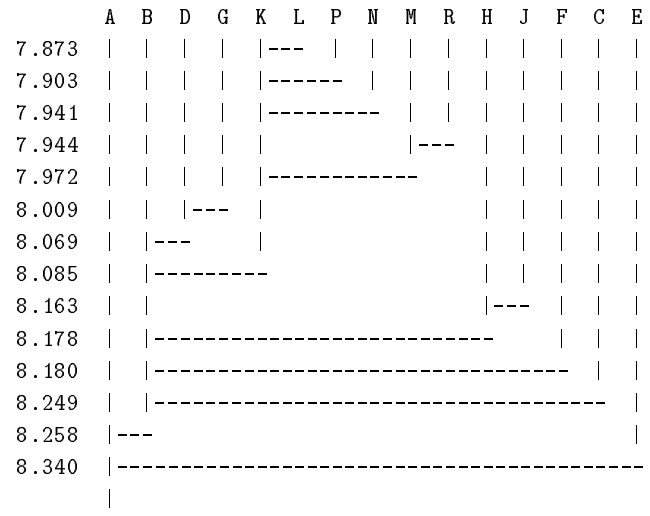

Figure 3.6: Clustering result based on lexical distances

Western Fiction" and "Romance and Love Story". This is intuitively true.

It is also interesting that the clustering results based on syntactic structure and lexical information are slightly different. For example, A (Press reportage) and E (Skills and hobbies) are far apart in lexical data, however on a syntactic basis, they are somewhat closer to each other. This is also intuitively understandable.

Lexical knowledge is very often used in text categorization. That is justifiable, because the purpose of most text categorization is to classify text in order to find semantically similar texts. However the purpose here is to improve the accuracy of natural language processing based on sublanguage knowledge. So, the fact that the two results are slightly different may suggest that the traditional technique of text categorization may not be the best method for the purpose. This is a very interesting and important result.

Other similarity results were obtained from some simple statistics. Table 3.4 shows some statistics for each domain; the items are numbered to correspond to the column in the table.

1. Number of sentences
2. Number of tokens
3. Average sentence length in tokens
4. Number of lexical items
5. Average frequency of a lexical item

6. Number of partial trees extracted in the experiment

7. Average number of partial trees per token

8. Number of distinct partial trees

Domain	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
A	4224	82943	19.64	13368	6.20	75677	0.91	3838
B	2706	51547	19.05	9396	5.49	48741	0.95	3007
C	1563	34246	21.91	8272	4.14	30628	0.89	2523
D	2393	33284	13.91	6188	5.38	31334	0.94	2393
E	3687	66277	17.98	11241	5.90	62046	0.94	3579
F	4442	89792	20.21	13813	6.50	85526	0.95	3999
G	6497	143611	22.10	17616	8.15	137394	0.96	5527
H	2476	58209	23.51	7656	7.60	53825	0.92	2982
J	6951	152909	22.00	16078	9.51	139177	0.91	5588
K	3661	54587	14.91	8886	6.14	54747	1.00	3151
L	3231	43872	13.58	6551	6.70	46107	1.05	2452
M	755	11324	15.00	3072	3.69	11074	0.98	1127
P	3625	54298	14.98	7998	6.79	56620	1.04	2995
R	898	17372	19.35	4751	3.66	16687	0.96	1627

Table 3.4: Some statistics across domain

From the table, we can clearly distinguish fiction domains (K to R) from non-fiction domains (A to J) based on 'Average length of sentence' (3) and 'Average number of partial trees per token' (6). The average length of sentences is shorter in fiction domains compared to that in non-fiction domains, although domain R (Humor) has relatively longer sentences. The average number of partial trees per token is larger in fiction domains than that in non-fiction domains. It's more than one partial tree per token in most of the fiction domains, which means the structure are deeper, and less than one partial tree per token in non-fiction domains, which means the structure are flatter. The fact that R (Humor) and M (Science Fiction) are slightly different from the other fiction domains is true also in the experiments based on the cross entropy and clustering, shown in Figure 3.3 and Figure 3.4.

3.6 Domain Specific Structures

3.6.1 Relatively Frequent Partial Trees

In the previous section, a global comparison based on syntactic structure was reported. It demonstrates a clear distinction between the fiction domains and the

non-fiction domains. However, it was not clear what kinds of structure characterize a particular domain. In this section, a study focused on the characteristics of each domain is conducted. Domain specific syntactic structures (partial trees) are extracted based on the following two conditions:

1. Relative frequency of a partial tree in a domain is at least 5 times greater than that in the entire corpus.

2. Its frequency in the domain is more than 5.

The first condition identifies the idiosyncrasy of the partial tree. The second condition is used in order to delete noise, because low frequency partial trees, many of which are erroneous or uninteresting, often satisfy the first condition.

3.6.2 List of Relatively Frequent Partial Trees

Appendix E shows the lists of relatively frequent partial trees which satisfy the two conditions mentioned in the previous subsection. Some examples and notes are added for some interesting results.

It obviously demonstrates that each domain has many idiosyncratic structures. Many of them are interesting to see and can be explained by our linguistic intuition. This result supports the idea of domain dependent grammar, because these idiosyncratic structures should be treated only by the grammar in that domain.

Even though the second condition tries to reduce the noise by discarding low frequency partial trees, several types of noise remain in the list. The main reason for the noise is the annotator's 'propensity'. For example, the manner of annotation for some identical sequences of parts-of-speech in different domains is sometimes different. It is not so difficult to find these, because they tend to happen in relatively common word sequences and we can compare them with the same instances in other domains.

Also, there are some types of noise which are caused by tagging errors. As in the process of the annotation, the sentences are tagged first by an automatic tagger and then human operators correct errors. So some consistent mistakes can be found, which were ignored by the operators. The most typical one is that of a capitalized word which is tagged as a proper noun, even if it is not. This kind of mistake is also easy to detect from the list.

Another kind of noise comes from how a sentence is segmented. For example, a compound sentence like `I said ``Yes.  I like it.´´`, usually is cut into two sentences `I said ``Yes.` and `I like it.´´`. Although it seems that this type of segmentation is fairly consistent, it is unavoidable that some peculiarities are introduced.

3.7 Application to Speech Recognition

In Section 2.11, an experiment to enhance continuous speech recognition with the parser was explained. That experiment tried to utilize syntactic information within a sentence. The experiment in this section is another effort to improve the accuracy of continuous speech recognition using long distance dependency. As was explained previously, most of the current continuous speech recognition systems use a tri-gram model as their language model. Here, a longer context, beyond sentence boundaries, is used. It is based on the notion of sublanguage, or topic coherence properties. The idea is that if we can find the sublanguage or the topic of the speech input, we can adjust the knowledge of the system to the sublanguage and may improve the recognition accuracy. In particular, since the usage of words is different in different sublanguages or domains (Section 3.5), this idea may have great potential.

3.7.1 Similar Approaches

There have been several attempts in the last few years to make use of topic coherence properties in order to improve recognition accuracy. Table 3.5 shows some of the recent work which use topic coherence techniques, including cache model and topic clustering methods. Because the evaluations were made on different test sets and

Site (Year)	Description	Improvement (word error rate)	References
IBM (91)	cache model		[Jelinek 91]
CMU (94)	trigger model	19.9 $\rightarrow$ 17.8	[Rosenfeld 94]
BU (93-94)	clustering (4 topic LM)	11.3 $\rightarrow$ 11.2	[Ostendorf et al. 95]
CMU (96)	hand clustering (5883 topic)	0.1,0.6% improve. in 2 story	[Seymore 97]
SRI (96)	clustering (4 topic LM)	33.1 $\rightarrow$ 33.0	[Weng 97]
CU (96)	cache model	27.7 $\rightarrow$ 27.5	[Woodland et al. 97]
Sekine	sublanguage and cache model	24.6 $\rightarrow$ 24.0 9.7 $\rightarrow$ 9.4	

Table 3.5: Approaches of topic coherent model in speech recognition

on different conditions, a direct comparison is not possible. In general, we can find improvements using these techniques, although many are relatively small (except for the CMU experiment (94), which was conducted on unique conditions). We can summarize the techniques in three categories:

- Cache
  Use information on previously uttered words to supplement the language model. It utilizes the property that the same words are repeatedly used in an article.

- Dynamic Topic Adaptation (trigger, sublanguage)
  Dynamically consult a database to build a language model for the topic based on the previous utterances. The data can be structured in advance (trigger model) or the raw text data can be retrieved and analyzed on demand (sublanguage model), but in either case the set of topics is not defined in advance.

- Clustering Language Model
  Prepare language models for several topics which are defined in advance (automatically or by hand). Then find the topic of the current segment and use the language model of the topic (or possibly a mix of several language models, also combined with the general language model).

3.7.2 Overview

In the system, the N-best hypotheses produced by the SRI speech recognition system are used, along with their acoustic and language model scores. The structure is the same as the that in the experiment explained in Figure 2.14 in Section 2.11. Note that none of SRI's language models take long-range dependencies into account. Their scores are combined with the score produced by the sublanguage component and the cache model score. Then the hypothesis with the highest combined score is selected as the output of the system. The relative weights of the eight scores are determined by an optimization procedure on a training data set.

The basic idea of the sublanguage experiment can be categorized into the 'dynamic topic adaptation' explained in the previous subsection. It tries to find the topic or sublanguage of the text, using the previously uttered sentences. Then a large corpus is used to make a language model for that specific topic and the language model, essentially unigram statistics, is used to improve the recognition accuracy.

3.7.3 Sublanguage Component

The sublanguage component performs the following four steps:

1. Select keywords from previously uttered sentences
2. Collect similar articles from a large corpus based on the keywords
3. Extract sublanguage words from the similar articles
4. Compute scores of N-best hypotheses based on the sublanguage words

A sublanguage analysis is performed separately for each sentence in an article (after the first sentence). There are several parameters in these processes, and the values

of the parameters used for this experiment will be summarized at the end of each process. Generally several parameter values were tried and the values shown in this paper are the best ones on the training data set.

A large corpus is used in the experiment as the source for similar articles. This corpus includes 146,000 articles, or 76M tokens, from January 1992 to July 1995 of North American Business News which consists of Dow Jones Information Services, New York Times, Reuters North American Business Report, Los Angeles Times, and Washington Post. This corpus has no overlap with the evaluation data set, which is drawn from August 1995 North American Business News.

Now, each process of the sublanguage component will be described in detail.

Select Keywords

The keywords which will be used in retrieving similar articles are selected from previously dictated sentences. The system which will be described here is an incremental adaptation system, which uses only the information the system has acquired from the previous utterances, which may be incorrect. So it does not know the correct transcriptions of prior sentences or any information about subsequent sentences in the article. For a comparison, a supervised adaptation experiment was conducted. In this mode, the correct transcriptions of the previous utterances are used.

Not all of the words from the prior sentences are used as keywords for retrieving similar articles. As is the practice in information retrieval, several types of words were filtered out. First of all, closed class words and high frequency words are excluded. Because these types of words appear in most of the documents regardless of the topic, it is not useful to include these as keywords. On the other hand, very low frequency words sometimes introduce noise into the retrieval process because of their peculiarity. Only open-class words of intermediate frequency (actually frequency from 6 to 100,000 in the corpus of 146,000 articles) are retained as keywords and used in finding the similar articles. Also, because the N-best sentences inevitably contain errors, a threshold is set for the appearance of words in the N-best sentences. Specifically, the requirement was enforced that a word appear in at least 15 times in the top 20 N-best sentences (as ranked by SRI's score) to qualify as a keyword for retrieval.

Parameter	Value
Maximum frequency of a keyword	100000
Minimum frequency of a keyword	6
N-best for keyword selection	20
Minimum no. of appearances of a word in N-best	15

Collect Similar Articles

The set of keywords is used in order to retrieve similar articles according to Formulae 3.7 and 3.8. Here $Weight(w)$ is the weight of word w, $F'(w)$ is the frequency of word w in the 20 N-best sentences, M is the total number of tokens in the entire corpus, $F(w)$ is the frequency of word w in the entire corpus, $AScore(a)$ is article score of article a, which indicates the similarity between the set of keywords and the article, and $n(a)$ is the number of tokens in article a.

$$Weight(w) = F'(w) * log(\frac{M}{F(w)}) \tag{3.7}$$

$$AScore(a) = \frac{\sum_{w \in a} Weight(w)}{log(n(a))} \tag{3.8}$$

Each keyword is weighted by the product of two factors. One of them is the frequency of the word in the 20 N-best sentences, and the other is the log of the inverse probability of the word in the large corpus. This is a standard metric of information retrieval based on the assumption that the lower frequency words provide more information about topics [Sparck-Jones 73]. For each article, article scores ($AScore$) for all articles (target) in the large corpus are computed as the sum of the weighted scores of the selected keywords and are normalized by the log of the size of the target article. This score indicates the similarity between the set of keywords and the article. The most similar 50 articles were collected from the corpus. These form the "sublanguage set", which will be used in analyzing the next sentence in the test article.

Parameter	Value
Number of articles in sublanguage set	50

Extract Sublanguage words

Sublanguage words are extracted from the collected sublanguage articles. This extraction was done in order to filter out topic-unrelated words. Here, function words are excluded, as was done in the keyword selection, because function words are generally common throughout different sublanguages. Next, to find strongly topic related words, words which appear in at least 3 out of the 50 sublanguage articles were extracted. Also, the document frequency in sublanguage articles has to be at least 3 times the word frequency in the large corpus:

$$\frac{DF(w)/50}{F(w)/M} > 3 \tag{3.9}$$

Here, $DF(w)$ is the number of documents in which the word appears. We can expect that these methods eliminate words less related to the topic, so that only words strongly related to the topic are extracted as the sublanguage words.

Parameter	Value
Minimum number of documents containing the word	3
Threshold ratio of word in the set and in general	3

Compute Scores for N-best Hypotheses

Finally, the scores for the N-best hypotheses generated by the speech recognizer were computed. The top 100 N-best hypotheses (according to SRI's score) are re-scored. The sublanguage score assigned to each word is the logarithm of the ratio of document frequency in the sublanguage articles to the word frequency of the word in the large corpus. The larger this score for a word, the more strongly the word is related to the sublanguage found through the prior discourse.

The score for each sentence is calculated by accumulating the score of the selected words in the hypothesis. Here $HScore(h)$ is the sublanguage score of hypothesis h.

$$HScore(h) = \sum_{w\ :\ Selected\ words\ in\ h} \log(\frac{DF(w)/50}{F(w)/M}) \tag{3.10}$$

This formula can be motivated by the fact that the sublanguage score will be combined linearly with general language model scores, which mainly consist of the logarithm of the tri-gram probabilities. The denominator of the log in Formula 3.10 is the unigram probability of word w. Since it is the denominator of a logarithm, it works to reduce the effect of the general language model which may be embedded in the trigram language model score. The numerator is a pure sublanguage score and it works to add the score of the sublanguage model to the other scores.

Parameter	Value
N-best to be re-scored	100

3.7.4 Cache model

A cache model was also used in the experiment. Not all the words in the previous utterance were used as cache words. Rather several types of words were filtered out and all of the "selected keywords" as explained in the last section were used for the cache model. Scores for the words in cache ($CScore(w)$) are computed in a similar way to that for sublanguage words. Here, N' is the number of tokens in the previously uttered N-best sentences.

$$CScore(h) = \sum_{w\ :\ words\ in\ cache} \log(\frac{F'(w)/N'}{F(w)/M}) \tag{3.11}$$

3.7.5 Experiment

There are 4 different conditions in the experiment.

1. Multiple microphone; incremental adaptation (P0)
2. Single microphone; incremental adaptation (C0)
3. Multiple microphone; supervised adaptation (C4A)
4. Single microphone; supervised adaptation (C4B)

'Single microphone' means that the speech was recorded using the close-talking Sennheiser HMD-410 microphone. 'Multiple microphone' means that the recoding was done using three secondary microphones that were unknown to the systems. 'Incremental adaptation' means that the data used for retrieving similar articles are derived from the previously recognized sentences. Since the recognizer produces N-best sentences, the top 20 sentences were used as the basis for the predictions. 'Supervised adaptation' means that the correct transcriptions of the previous sentences are used as the basis for selecting similar articles from the corpus and for the cache. P0, C0, C4A, C4B are labels to represent each experiment condition.

The parameters of the sublanguage component are tuned and the weight of each component is decided by the training optimization. The evaluation of the sublanguage method is done by comparing the word error rate of the system with sublanguage scores to that of the SRI system without sublanguage scores.

The absolute improvement using the sublanguage component over SRI's system, for example, on P0 is 0.6%, from 24.6% to 24.0%, as shown in Table 3.6. The figures in the table was based on the formal evaluation conducted at NIST [Speech Workshop 96]. The absolute improvement looks tiny; however, there is a

System	P0	C0	C4A	C4B
SRI	24.6 %	9.7 %	24.6 %	9.7 %
SRI + SL	24.0 %	9.4 %	24.3 %	9.3 %

Table 3.6: Formal result of speech recognition

limit to the improvement we can obtain, because the N-best sentences don't always contain the correct candidate. This will be discussed in the next subsection.

3.7.6 Discussion

Inevitably, this evaluation is affected by the performance of the base system. In particular, the number of errors produced by the base system and the minimum number of errors obtainable by choosing the N-best hypothesis with minimum error for each

sentence, are important. (Let's call the latter error "MNE" for "minimal N-best errors".) The difference between these numbers indicates the possible improvement which can be achieved by re-scoring the hypotheses. Note that the detailed error analysis shown in this section is based on unofficial figures, which were evaluated internally and are different from the numbers shown in Table 3.6. Also, for simplicity, only the P0 evaluation result is discussed in this subsection. Table 3.7 shows the

	Number of errors	Word Error Rate
SRI system	1522	25.37 %
MNE	1147	19.12 %
Possible Improvement	375	6.25 %

Table 3.7: Word errors of the base system and MNE

number of errors and word error rate of the SRI system and MNE. The difference of the two numbers represents the possible improvement by re-scoring the hypotheses. It is unreasonable to expect the sublanguage model to fix all of the 375 word errors (non-MNE). For one thing, there are a lot of word errors unrelated to the article topic, for example function word replacement ("a" replaced by "the"), or deletion or insertion of topic unrelated words (missing "over"). Also, the word errors in the first sentence of each article are not able to be fixed[2].

The absolute improvement using the sublanguage component over SRI's system is about 0.65%, from 25.37% to 24.72% (again as an internal figure), as shown in Table 3.8. That is, the number of word errors is reduced from 1522 to 1483. This means that 10.40% of the possible improvement was achieved (39 out of 375). Although the absolute improvement looks tiny, the relative improvement excluding

System	Word Error Rate	Num.of Error	Improvement exclude MNE
SRI	25.37 %	1522	
SRI + SL component	24.72 %	1483	10.40 %

Table 3.8: Word error rate and improvement

MNE, 10.40 %, is quite impressive, because, as was explained before, there are some types of error which can not be corrected by the sublanguage model.

Figure 3.7 shows an example of the actual output of the system. (This is a relatively badly recognized example.) The first sentence is the correct transcription, the

[2]Note that, in the experiment, a few errors in initial sentences were corrected, because of the weight optimization based on the eight scores which includes all of the SRI's scores. But this effect is very minor and these improvements are offset by a similar number of errors introduced for the same reason.

second one is SRI's best scored hypothesis, and the third one is the hypothesis with the highest combined score of SRI and sublanguage/cache models. This sentence is the 15th in an article on memory chip production. As you can see, a mistake in SRI's hypothesis, `membership` instead of `memory` and `chip`, was replaced by the correct words. However, other parts of the sentence, like `hyundai corporation and fujitsu`, were not amended. This particular error is one of the MNE, for which there is no correct candidate in the N-best hypotheses. Another error, `million` or `day` instead of `billion`, is not a MNE. There exist some hypotheses which have `billion` at the right spot, (the 47th candidate is the top candidate which has the word). The sublanguage model works to replace word `day` by `million`, but this was not the correct word.

```
1) in recent weeks hyundai corporation and fujitsu limited
   announced plans for memory chip plants in oregon at
   projected costs of over one billion dollars each

2) in recent weeks CONTINENTAL VERSION SUGGESTS ONLY limited
   announced plans for MEMBERSHIP FINANCING FOR IT HAD
   projected COST of one DAY each

3) in recent weeks CONTINENTAL VERSION SUGGESTS ONLY limited
   announced plans for memory chip plants in WORTHINGTON
   PROJECT COST of one MILLION each

   1) Correct Transcription, 2) SRI´s best, 3) SRI + SL
```

Figure 3.7: Example of transcription

3.7.7 Future Work on Topic Coherence Model

The results for the word error rate suggest that the sublanguage technique can be very useful in improving the speech recognition rate. However, the actual improvement in word error rate is relatively small, partially because of factors which could not be controlled, of which the problem of MNE is the most important. One of the methods for increasing the possibility of improvement is to make N (of N-best) larger, thus including more correct hypotheses in the N-best. Although SRI provides us N-best for up to 2000, parameter optimization showed us that 100 is the optimal number for this parameter. This result can be explained by the following statistics. Table 3.9 describes the number of MNE as a function of N for the training data set and evaluation data set. Also in brackets, the maximal possible improvement for each case is shown. According to the table, the number of MNE

N	MNE for evaluation	MNE for training
1	1522	1258
50	1163 (359)	991 (267)
100	1147 (375)	960 (298)
200	1134 (388)	947 (311)
500	1116 (406)	935 (323)
1000	1109 (413)	930 (328)
2000	1107 (415)	929 (329)

Table 3.9: N-best and word error

decreases rapidly for N up to 100; however, after that point, the number decreases only slightly. For example, in the evaluation data set, increasing N from 500 to 2000 introduces only 9 new possible word error improvements. This small number might give the component greater opportunity to include errors rather than improvements.

Improvements could be drawn from better adjustment of the parameter settings. There are parameters involved in the similarity calculation, the size of the sublanguage set, the ratio threshold, etc. To date, the tuning was done by manual optimization using a relatively small number of trials and a very small training set (the 20 articles which have N-best transcriptions). It is necessary to use automatic optimization methods and a substantially larger training set. With regard to the size of sublanguage set, a constant size may not be optimal. It might be interesting to use the technique described in Section 3.3. It selects the size of sublanguage automatically by seeking the minimum ratio of the document set perplexity to the estimated perplexity of randomly selected document sets of that size. This remains as a future work.

It might be worthwhile to reconsider how to mix the sublanguage and cache score with SRI's language model score. SRI provides language model scores for each sentence hypothesis, not for each word. However, if their language score can be computed with high confidence for a particular word, then the model should have relatively little weight. On the other hand, if the language model has low confidence for a word, sublanguage should have a heavy weight. In other words, the combination of the scores should not be done by a combination at the sentence level, but should be done at the word level.

Chapter 4

Sublanguage and Parsing

4.1 Introduction

In this chapter, experiments to observe domain dependency for parsing will be reported. The chart parser explained in Section 2.7 is used and grammars and dictionaries are extracted from the Brown corpus. About the Brown corpus, please refer Section 3.4. Texts from different domains are parsed by different kinds of grammars. Although one may raise some concern about the definition of the domain, which we discussed in the previous chapter, if the parsing results show a tendency toward improved accuracy when the text is parsed by the grammar from the same or similar domains, it exhibits the usefulness of the sublanguage notion.

Two experiments will be described. The first is an individual experiment, where texts from 8 domains are parsed with 4 different types of grammars. These are grammars acquired from the corpora of the same domain, all domains, non-fiction domains and fiction domains.

The other parsing experiment is an intensive experiment, where we try to find the best suitable grammar for texts in two particular domains. Also we try to see the relationship to the size of the training corpus. We use the domains of 'Press Reportage' and 'Romance and Love Story' in this intensive experiment.

Finally, discussion drawn from the experiments will be presented.

4.2 Individual Experiment

Table 4.1 shows the parsing performance for domains `A`, `B`, `E`, `J`, `K`, `L`, `N` and `P` with four types of grammars. In the table, results are shown in the form of 'recall/precision'. Each grammar is acquired from roughly the same size (21 to 24 samples) of corpus. All the grammar are 5NT grammar without any lexicalization, see Section 2.5. The grammar of all domains is created using a corpus of 3 samples each from the 8 domains. The grammar of non-fiction and fiction domains are created from a corpus

of 6 samples each from 4 domains. Then text of each domain is parsed by the four types of grammar. There is no overlap between training corpus and test corpus.

Document / Grammar	Itself	All	non-fiction	fiction
A. Press: Reportage	66.62/64.14	64.39/61.45	65.57/62.40	62.23/59.32
B. Press: Editorial	67.65/62.55	64.67/61.78	65.73/62.69	63.03/60.36
E. Skills/Hobbies	64.05/60.79	65.25/61.51	65.26/62.18	62.87/59.04
J. Learned	67.80/65.59	65.87/63.90	65.57/64.58	63.04/60.77
K. General fiction	70.99/68.54	71.00/68.04	70.04/66.64	71.79/68.95
L. Mystery/Detective	67.59/65.02	68.08/66.22	67.32/64.31	68.89/66.55
N. Adventure/Western	73.09/71.38	72.97/70.27	70.51/67.90	74.29/72.23
P. Romance/Love story	66.44/65.51	64.52/63.95	62.37/61.55	64.69/64.50

Table 4.1: Parsing accuracy for individual section

We can see that the result is always the best when the grammar acquired from either the same domain or the same class (fiction or non-fiction) is used. We will call the division into fiction and non-fiction as 'class'. It is interesting to see that the grammar acquired from all domains is not the best grammar in most of the tests. (Only for E, is it very close to the best result.) In other words, if the size of the training corpus is the same, using a training corpus drawn from a wide variety of domains does not help to achieve better parsing performance. This shows the domain dependency of parsing.

For non-fiction domain texts (A, B, E and J), the performance of the fiction grammar is notably worse than that of the same domain grammar or the same class grammar. In contrast, the performance on some fiction domain texts (K and L) with the non-fiction grammar is not so different from that of the same domain. Here, we can see some relationships between these results and the cross entropy observations described in Section 3.5. The matrix shown in Figure Figure 4.1. is a subset of the matrix in Figure 3.3. In the matrix, the cross entropies where any of the fiction domains are models and any of the non-fiction domains are tests (right-top in the matrix) have much higher figures than the cross entropies where any of the non-fiction domains are models and any of the fiction domains are tests (right-top in the matrix) This means that the fiction domains are less suitable for modeling the syntactic structure of the non-fiction domains compared to the reverse case. This observation explains the result of parsing very nicely.

It is not easy to consider why, for some domains, the result is better with the grammar of the same class rather than the same domain. One rationale we can think of is based on the comparisons described in Section 3.5. For example, in the clustering experiment based on syntactic structure shown in Figure 3.5, we have seen that domains K, L and N are very close and make a cluster at the earliest stage and domain P is the last one to join the cluster. So it may be plausible to say that

T\M	A	B	E	J	K	L	N	P
A	5.132	5.357	5.418	5.452	5.513	5.526	5.538	5.558
B	5.475	5.190	5.500	5.519	5.550	5.587	5.600	5.604
E	5.506	5.483	5.204	5.484	5.587	5.593	5.583	5.617
J	5.395	5.370	5.350	5.158	5.529	5.571	5.583	5.596
K	5.326	5.258	5.317	5.418	4.955	5.145	5.150	5.174
L	5.322	5.269	5.322	5.456	5.122	4.914	5.095	5.132
N	5.299	5.259	5.282	5.431	5.102	5.068	4.899	5.122
P	5.439	5.367	5.405	5.550	5.235	5.213	5.211	5.004

Figure 4.1: Cross entropy of syntactic structure across domains

the grammar of the fiction domains is mainly representing K, L and N, and because it may contain a wide range of syntactic structures, it produced a good performance for each of these domains. For domain P, the fiction domain grammar may not fit well. In other words, this is a small sampling problem. Because only 24 samples or texts of 48000 words are used, a single domain grammar tends to cover a relatively small part of the language phenomena. On the other hand, a corpus of very similar domains could provide wider coverage.

4.3 Intensive Experiment

In this section, several parsing experiments on texts of two domains are reported. The texts of the two domains are parsed with several grammars, e.g. grammars acquired from different domains, classes and combined, and also different sizes of the training corpus. The size of the training corpus is an interesting and important issue, which was also discussed in Section 2.6. The experiments demonstrated that the smaller the training corpus, the poorer the parsing performance. However, for example, we don't know which of the following two types of grammar produce better performance: a grammar trained on a smaller corpus of the same domain, or a grammar trained on a larger corpus including different domains.

We can imagine, in a practical situation, that we have a relatively big general corpus and a small domain corpus. Experiments which simulate such situations were conducted. That is an all-domain corpus was used as a general training corpus, and each domain corpus was added to the training corpus; then grammars were extracted from several combinations of these two kinds of corpora.

Table 4.2 shows the parsing performance on press reportage text using grammars of different domains. Table 4.3 shows the parsing performance on Romance and Love Story text using grammars of different combinations. Here "ALL" means the grammar acquired from the combination of all 8 domain corpora (the 192 sample

corpus contains all 24 samples of all domains), "FIC" means the fiction domain, "NNF" means the non-fiction domain, and letters represent single domains. The number in parentheses indicates the size of the training corpus; more precisely it is the number of samples in terms of the Brown corpus used for the training. The size of a specific domain corpus is increased by replicating the same corpus, because of the corpus availability limitation. We can observe an interesting result from

Grammar	Recall	Precision
A (24)	66.62	64.14
ALL (24)	64.39	61.46
ALL (24) + A (24)	66.73	64.73
ALL (192)	66.74	65.52
ALL (192) + A (24)	66.77	65.61
ALL (192) + A (48)	67.15	65.96
ALL (192) + A (72)	67.08	65.88
ALL (192) + A (120)	67.09	65.82
NNF (24)	65.57	62.40
NNF (24) + A (24)	67.90	65.55
FIC (24)	62.23	59.32
FIC (24) + A (24)	66.35	64.51

Table 4.2: Parsing accuracy for Press Reportage domain

the experiments combining the general corpus and domain corpus. In all cases except ALL(192), (which already includes the entire domain-specific corpus), adding the domain-specific corpus improves the accuracy quite substantially. Figure 4.2 shows the relationship between the degree of combination and parsing accuracy. The graph draws two lines of the parsing experiment based on the data of ALL(192) through ALL(192)+A(120) in Table 4.2 and that for P in Table 4.3. In the graph, the proportion of the domain-specific corpus in the training corpus is increasing from left to right. Note that the rightmost points indicate the parsing accuracy using the same domain grammar alone, which means the recall of A(24) for the Press Reportage and the recall of P(24) for the Romance domain. We can see an interesting result from the experiments of combining the general corpus and domain corpus. There is a maximum on parsing accuracy in the middle of the mixture in the Press Reportage domain. Although it is a relatively small improvement, it is worthwhile, because the results came from the exactly the same knowledge sources, the difference is the mixture. In the Romance and Love Story domain, the pure general domain grammar has the maximum accuracy, but note that it already includes the entire portion of the Romance and Love Story domain corpus.

Now the experiments concerning training size are reported. Figure 4.3 and Figure 4.4 show the graph of recall and precision of the parsing result for the Press

Grammar	Recall	Precision
P (24)	66.44	65.51
ALL (24)	64.52	63.84
ALL (24) + P (24)	67.40	66.97
ALL (192)	67.96	69.22
ALL (192) + P (24)	67.48	68.78
ALL (192) + P (48)	67.38	68.52
ALL (192) + P (72)	67.57	68.71
ALL (192) + P (120)	67.38	68.42
NNF (24)	62.37	61.55
NNF (24) + P (24)	67.38	66.44
FIC (24)	64.69	64.50
FIC (24) + P (24)	67.19	66.96

Table 4.3: Parsing accuracy for Romance domain

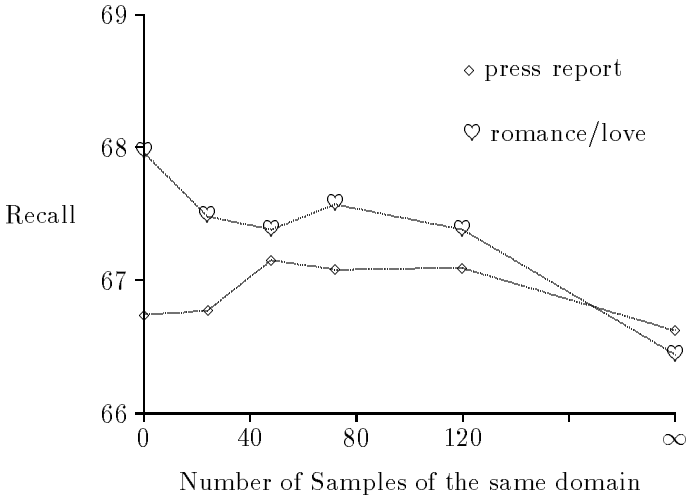

Figure 4.2: Degree of combination and the accuracy

Reportage text (domain A). Also, Figure 4.5 and Figure 4.6 show the graph of recall and precision of the parsing result for the Romance and Love Story text (domain P). The same text is parsed with 5 different types of grammars acquired from training corpora of different sizes. Because of corpus availability, it is impossible to create single domain grammars based on a large training corpus. The accuracy as a

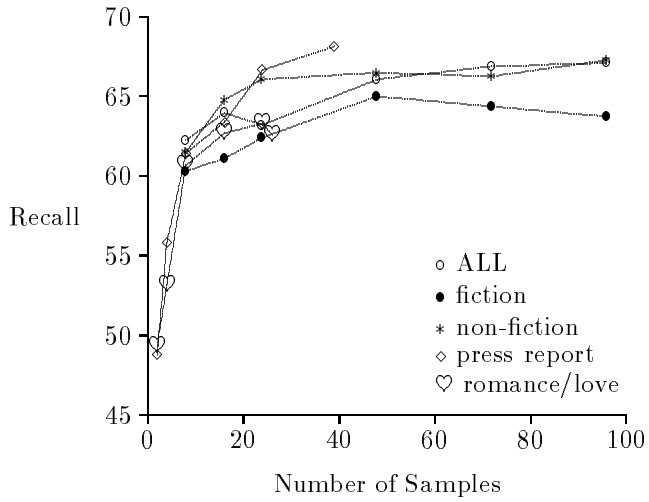

Figure 4.3: Size and Recall (Press Report)

function of the size of training corpus is monotonically increasing, with the slope gradually flattening. Note that the small declines of some graphs at large number of samples are mainly due to the memory limitation for parsing which was explained in Section 2.5. It is very interesting to see that the saturation point of any line on the graph is about 10 to 30 samples. That is about 20,000 to 60,000 words, or about 1,000 to 3,000 sentences. In the romance and love story domain, the precision of the grammar acquired from 8 samples of the same domain is only about 2% lower than the precision of the grammar trained on 26 samples of the same domain. We believe that the reason why the performance in this domain saturates with such a small corpus is that there is relatively little variety in the syntactic structure of this domain.

The order of the performance is generally the following: the same domain (best), the same class, all domains, the other class and the other domain (worst). The performance of the grammars of all domain and the other class are very close in many cases. In the romance and love story domain (Figure 4.6 and 4.5), the grammar acquired from the same domain made the solo best performance. The accuracies based on the grammars of the same domain and based on the other grammars are quite different. The results for the press reportage (Figure 4.4 and 4.3) are not so obvious, but the same tendencies can be observed.

In terms of the relationship between the size of training corpus and domain

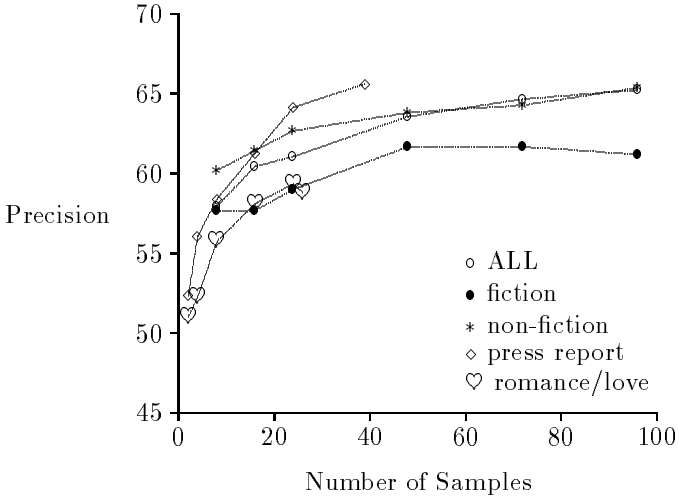

Figure 4.4: Size and Precision (Press Report)

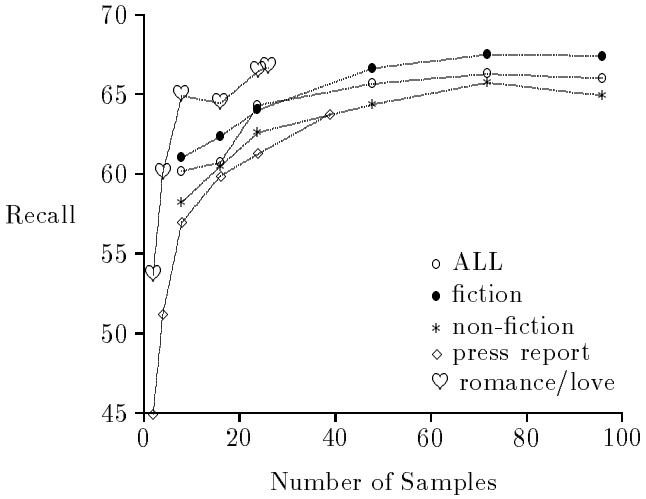

Figure 4.5: Size and Recall (Romance)

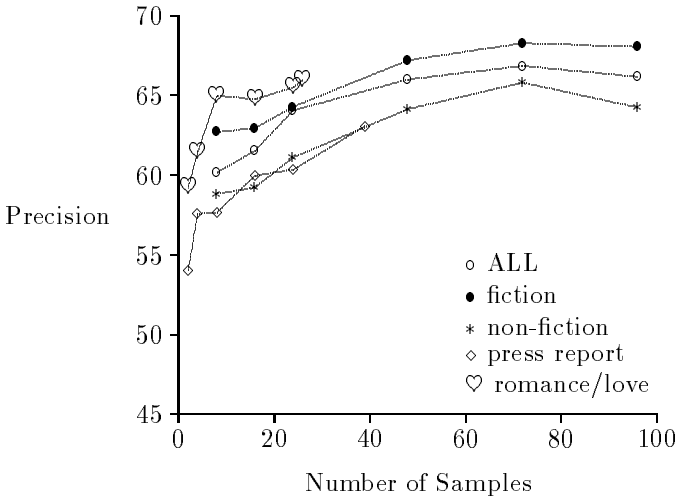

Figure 4.6: Size and Precision (Romance)

dependency, the performance of the grammar acquired from 24 samples of the same domain ('baseline grammar'), and that of the other grammars shall be compared. In the press reportage domain (Figures 4.4 and 4.3), about three to four times larger corpus of all domains or non-fiction domains is needed to match the performance of the baseline grammar. It should be noted that a quarter of the non-fiction domain corpus and one eighth of the all domain corpus consists of the press report domain corpus. In other words, the fact that the performance of the baseline grammar is about the same as that of 92 samples of the non-fiction domains means that in the latter grammar, the rest of the corpus gives only a little improvement for the parsing performance. In the romance and love story domain (Figure 4.6 and 4.5), the fiction domain grammar and all domain grammar quickly catch up to the performance of the baseline grammar. Less than twice as large a fiction domain corpus is needed to achieve the performance of the baseline grammar. From these two results and the evidence from the previous section, we can say that if you have a corpus consisting of similar domains, it is worthwhile to include the corpus in grammar acquisition, otherwise it is not so useful. We need to further quantify these relationships between the syntactic diversity of individual domains and the difference between domains.

4.4 Discussion

One of our basic claims from the parsing experiments is the following. When we try to parse a text in a particular domain, we should prepare a grammar which suits that domain. This idea naturally contrasts with the idea of robust broad-coverage parsing [Carroll and Briscoe 96], in which a single grammar should be prepared for parsing any kind of text. Obviously, the latter idea has the great advantage that you

do not have to create a number of grammars for different domains nor do you need to consider which grammar should be used for a given text. On the other hand, it is plausible that a domain specific grammar can produce better results than a domain independent grammar. Practically, the increasing availability of corpora provides the possibilities of creating domain dependent grammars. Also, it should be noted that we don't need a huge corpus to achieve a fairly good quality of parsing.

Undoubtedly the performance depends on the parsing strategy, the corpus and the evaluation methods. However, the results of the experiments suggest the significance of the notion of domain in parsing. The results would be useful for deciding what strategy should be taken in developing a grammar for a 'domain dependent' NLP application system.

Chapter 5

Conclusion

In this thesis, research on the following topics has been presented.

- A corpus-based probabilistic parser
  A novel approach to corpus-based parsing was proposed. It exhibited good performance and several strategies to improve the parser were tried. A Japanese parser based on the same idea was developed.

- Sublanguage study
  Domain dependency of language, in particular concerning syntactic structure, was studied. Several statistics and studies of sublanguage were introduced.

- Speech Recognition experiment using parser and sublanguage technique
  The parsing and sublanguage techniques were applied to speech recognition. An experiment using the parser demonstrated some promising examples. The sublanguage technique improved the speech recognition performance.

- Sublanguage and Parser
  Parsing experiments based on the knowledge extracted from different domains showed the domain dependency of parsing. This empirical result serves as an important suggestion for future work in domain dependent natural language processing.

New approaches, challenging applications and empirical studies were reported on these topics. However, none of the work is by any means completed. We can commence explorations in several directions from the work done in this thesis. These future projects were mentioned in each chapter, but here, the author would like to conclude the thesis by listing his major concerns.

- In order to improve the parsing accuracy, more trials should be conducted. This includes investigation of new non-terminal categories and back-off methods to more general grammar rules.

- In parsing, a richer lexical dependency model should be incorporated, since it has been shown that it is useful information for achieving a good parsing performance.
- A mechanism to reduce the required memory size for the parser should be considered. In particular, it is necessary to eliminate fitted sentences, because the quality of fitted sentences are generally worse. A technique like CYK parsing can be used.
- In sublanguage studies, the relationships between the nature of domain and the effect in processings should be analyzed. For example, how the domain specific structures affect parsing performance is one of the interesting topics.
- In the speech recognition experiment, although there were some examples where the parsing could help to improve the performance, the overall performance was not improved. Means of transferring these successes into better overall performance need to be discovered.

Appendix A

Symbol Definition

Definition of parts-of-speech Symbols

Categories with '*' mark are new categories which are not defined in PennTree-Bank.

CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential *there*
FW	Foreign word
IN	Preposition
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NNX	Noun*
NN	Noun (singular)
NNS	Noun (plural)
NNPX	Proper noun*
NNP	Proper noun (singular)
NNPS	Proper noun (plural)
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PP$	Possessive pronoun

RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol (mathematical or scientific)
TO	*to*
UH	Interjection
VB	Verb, base form
VBX	Verb, present tense, past tense*
VBP	Verb, present tense
VBZ	Verb, present tense, third singular
VBD	Verb, past tense
VBG	Verb, gerund/present participle
VBN	Verb, past participle
WDT	*wh*-determiner
WP	*wh*-pronoun
WP$	Possessive *wh*-pronoun
WRB	*wh*-adverb
#	Pound sign
$	Dollar sign
,	Comma
:	Colon, semi-colon
-LRB-	Left bracket character
-RRB-	Right bracket character
''	Straight double quote
`	Left open single quote
``	Left open double quote
'	Right close single quote
''	Right close double quote
@OF	*of**
@SNC	subordinate conjunction*
@DLQ	quantifier modifier*

Definition of grammatical node Symbols

ADJP	Adjective phrase
ADVP	Adverb phrase
CONJP	Conjunction phrase
INTJ	Interjection
NP	Noun phrase
PP	Prepositional phrase
PRT	Parenthetical
S	Simple declarative clause
SBAR	Clause introduced by subordinating conjunction
SBARQ	Direct question introduced by *wh*-word/phrase
SINV	Declarative sentence with subject-aux inversion
SQ	Subconstituent of SBARQ excluding *wh*-word/phrase
VP	Verb phrase
WHADVP	*Wh*-adverb phrase
WHNP	*Wh*-noun phrase
WHPP	*Wh*-prepositional phrase
X	Constituent of unknown or uncertain category
SS	S but not the top S*
TOINF	To infinitive*
NPL	Lowest noun phrase (BaseNP)*

Appendix B

Table for Finding Head

NT	Direction	Order of categories to be searched
S	right-to-left	VP S SS SBAR ADJP UCP SINV SBARQ SQ NP NPL
SS	right-to-left	VP S SS SBAR ADJP UCP SINV SBARQ SQ NP NPL
ADJP	right-to-left	QP ADJP $ JJ JJR JJS DT FW CD RB RBR RBS RP
ADVP	left-to-right	ADVP RB @DLQ RBR RBS FW IN TO CD JJ JJR JJS
CONJP	left-to-right	CC RB IN
FRAG	left-to-right	FRAG
INTJ	right-to-left	INTJ
LST	left-to-right	LS :
NAC	right-to-left	NNX NN NNS NNPX NP NPL NNPS NNP NAC EX $ CD QP PRP VBG JJ JJS JJR ADJP FW
NP	right-to-left	NP NPL NNX NN NNS NNP NNPS NAC EX $ CD QP PRP VBG NX DT JJ JJR JJS ADJP FW RB RBR RBS SYM PRP$
NPL	right-to-left	NNX NN NNS NNP NNPS NAC EX $ CD QP PRP VBG NX DT JJ JJR JJS ADJP FW RB RBR RBS SYM PRP$
NX	right-to-left	NX NNX NN NNS NNP NNPS
PP	left-to-right	IN @OF TO FW PP
PRN	right-to-left	S SS NP NPL SINV PP ADVP
PRT	left-to-right	RP @DLQ
QP	right-to-left	CD NNX NN NNS JJ JJR JJS RB RBR RBS DT $
RRC	left-to-right	VP S SS NP NPL ADVP ADJP PP
SBAR	right-to-left	SBAR @SNC TOINF IN WDT VP WHADVP WHADJP WHNP WHPP WP RBS DT @DLQ S
SBARQ	right-to-left	SQ S SS SINV SBARQ FRAG X
SINV	right-to-left	VP VBZ VBD VBP VB MD SQ S SS SINV ADJP
SQ	right-to-left	VP VBX VBZ VBD VBP VB MD SQ

```
UCP    left-to-right  UCP
VP     right-to-left  VBN VBX VBD MD VBZ VBP VP VB VBG ADJP NP NPL
WHADJP right-to-left  WHADJP WHADVP WRB CC JJ ADJP
WHADVP left-to-right  WRB CC
WHNP   right-to-left  WDT WP WP$ WHADJP WHPP WHNP WRB @SNC DT @DLQ
                      NP NPL
WHPP   left-to-right  IN @OF TO FW PP
X      left-to-right  X
TOINF  left-to-right  TO
AUX    right-to-left  NP NPL
NEG    right-to-left  RB
NNPX   left-to-right  NNPX
FW     left-to-right  FW
CD     left-to-right  CD
SYM    left-to-right  SYM
```

Appendix C

Examples of Binary Comparison

Scores of parser (left) and trigram (right) are shown in parentheses. Smaller numbers are better.

```
C: correct sentence      S: SRI-best candidate
```

Parser and Trigram both favor SRI-best

```
C: some dealers of foreign cars also lowered their
                                      japanese prices (448,655)
S: some dealers of foreign cars also lowered the
                                      japanese prices (424,614)

C: the problem isn´t gridlock he says the wheels are
                                    out of alignment (598,646)
S: the problem is in gridlock he says the wheels are
                                    out of alignment (567,625)
```

Trigram favors Correct, but Parser favors SRI-best (Bad example)

```
C: board would review distributing the remaining shares in the
 gold subsidiary to the parent company´s shareholders (1328,1306)
S: board would review distributing the remaining shares in the
 gold subsidiary to the parent company   shareholders (1254,1333)
```

Trigram favors SRI-best, but Parser favors Correct (Good example)

```
C: mcdonnell douglas corporation has built
                              helicopter parts ... (1360,1548)
S: mcdonnell douglas corporation and bell
                              helicopter parts ... (1404,1491)

C: they are interested in commodities as   a new asset class
                                             van says (521,731)
S: they are interested in commodities says a new asset class
                                             van says (560,720)

C: weary of worrying about withdrawal   charges
                           if you want to leave ... (1132,1273)
S: weary of worrying about withdraw all charges
                           if you want to leave ... (1210,1202)

C: this scenario as they say on t.v. is
                                based on a true story (550,649)
S: this scenario as a    say on t.v. is
                                based on a true story (576,644)

C: indirect  foreign ownership is limited to 25% (613,723)
S: in direct foreign ownership is limited to 25% (695,709)

C: even some lawyers now refer clients to mediators
                 offering to review the mediated agreement and
                           provide advice if needed (1045,1255)
S: even some lawyers now refer clients to mediators
                offering to review the mediated agreement can
                           provide advice if needed (1067,1253)
```

Others

```
C: the may figures show signs of improving sales said
                                    noriyuki matsushima (951,?)
S: the may figures show signs of improving sales said
                           nora you keep matsui shima (1211,?)
```

Appendix D

Brown Corpus

```
CATEGORIES of Brown Corpus (In total 500 samples)
================================================

I. Informative Prose     (374 samples)

  A. Press: Reportage  (44)
      Political   (Daily 10, Weekly 4, Total 14)
      Sports      (       5,        2,        7)
      Society     (       3,        0,        3)
      Spot News   (       7,        2,        9)
      Financial   (       3,        1,        4)
      Cultural    (       5,        2,        7)

  B. Press: Editorial  (27)
      Institutional      (7,        3,       10)
      Personal           (7,        3,       10)
      Letters to the Editor (5,     2,        7)

  C. Press: Reviews (17)
      Theatre, books, music, dance
                         (14,       3,       17)

  D. Religion  (17)
      Books                                ( 7)
      Periodicals                          ( 6)
      Tracts                               ( 4)

  E. Skills and Hobbies  (36)
```

Books (2)
Periodicals (34)

F. Popular Lore (48)
Books (23)
Periodicals (25)

G. Belles Letters, Bibliography, Memories, etc (75)
Books (38)
Periodicals (37)

H. Miscellaneous (30)
Government Documents (24)
Foundation Reports (2)
Industry Reports (2)
College Catalog (1)
Industry House Organ (1)

J. Learned (80)
Natural Science (12)
Medicine (5)
Mathematics (4)
Social and behavioral Sciences (14)
Political Science, Law, Education (15)
Humanities (18)
Technology and Engineering (12)

II. Imaginative Prose (126 Samples)

K. General Fiction (29)
Novels (20)
Short Stories (9)

L. Mystery and Detective Fiction (24)
Novels (20)
Short Stories (4)

M. Science Fiction (6)
Novels (3)
Short Stories (3)

N. Adventure and Western Fiction (29)

Appendix E

Examples of Idiosyncratic Structures

A. Press: Reportage

ratio	freq./total (in domain)	freq./total (in corpus)	rule (Example)
9.40	11 / 75662	14 / 905142	NP -> NNPX NNX NP
9.30	7 / 75662	9 / 905142	NP -> NP POS JJ NNPX
8.70	8 / 75662	11 / 905142	S -> NP VBX VP NP PP
8.44	12 / 75662	17 / 905142	NP -> DT $ CD NNX
8.30	77 / 75662	111 / 905142	NP -> NNPX NP
7.98	8 / 75662	12 / 905142	NP -> NP , NP -LRB- NP -RRB-
7.98	6 / 75662	9 / 905142	NP -> DT NNX NP PP
7.61	7 / 75662	11 / 905142	S -> NP MD VP PP PP
7.56	60 / 75662	95 / 905142	S -> NP :
7.48	10 / 75662	16 / 905142	NP -> NNPX .
7.18	6 / 75662	10 / 905142	NP -> DT NNPX NNX NNX NNX
6.98	7 / 75662	12 / 905142	NP -> JJ NNX NP
6.67	29 / 75662	52 / 905142	NP -> NNX NP
6.53	6 / 75662	11 / 905142	S -> NP VP PP NP
6.53	6 / 75662	11 / 905142	NP -> JJ VBG
6.44	14 / 75662	26 / 905142	NP -> NNPX , NNPX
6.44	7 / 75662	13 / 905142	ADJP -> VBN SBAR
6.20	29 / 75662	56 / 905142	NP -> NNPX NNPX NNPX
5.98	8 / 75662	16 / 905142	NP -> NP POS JJS NNX
5.98	7 / 75662	14 / 905142	X -> NP , PP
5.72	22 / 75662	46 / 905142	ADJP -> CD
5.52	6 / 75662	13 / 905142	NP -> DT NNPX CD NNX
5.46	63 / 75662	138 / 905142	NP -> CD NNPX

```
 5.20  10 /  75662   23 / 905142   NP  ->   ADVP $ CD
 5.20  10 /  75662   23 / 905142   ADVP  ->   RB NP
 5.18  13 /  75662   30 / 905142   NP  ->   DT @DLQ CD NNX
 5.18  13 /  75662   30 / 905142   NP  ->   CD PP PP
 5.13  12 /  75662   28 / 905142   NP  ->   NP POS JJ NNX NNX
 5.07  25 /  75662   59 / 905142   NP  ->   CD RB
 5.04   8 /  75662   19 / 905142   NP  ->   DT NNPX NNPX NNPX

----- Example ------------------------------------
 8.44   NP  ->   DT $ CD NNX
                   "the $40,000,000 budget"
                   "a $250 award"
                   "a 12,500 payment"
 8.30   NP  ->   NNPX NP
                   "Vice President L.B. Johnson"
                   "Atty. Gen. J. Joseph Nugent"
                   "First Lady Jacqueline Kennedy"
 7.56  S  ->   NP :
                   "Austin, Texas --"
                   "Geneva, June 18 --"
                   "Washington (AP) --"
 5.46   NP  ->   CD NNPX
                   "51st Street"
                   "734 Hartford Avenue"
                   "3300 Lake Shore Dr."
```

It is easy to notice that there are a number of rules which have `NNPX` (proper noun) in this list. The only domains which have more than three rules with `NNPX` are domain A, C and H. Domain H (Miscellaneous section) is rather difficult to analyse, but it is intuitively understandable that domain A and C have relatively frequent rules with `NNPX`. These are names of people, companies, places, etc. which are mentioned in press reports or reviews.

Also, we can find some typical rules of the domain, like `S -> NP :`, which is the first line of news reports, rules with numbers (`CD`), in particular with `$` sign. (There are many rules with numbers in domain J (Learned section), which is also intuitively understandable (mathmatic or enumerations), but none of the rules in that domain have `$` sign.)

B. Press: Editorial

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
18.57  34 /  48733   34 / 905142   S  ->   PP :
18.57   7 /  48733    7 / 905142   SQ  ->   NP SQ
18.57   6 /  48733    6 / 905142   X  ->   NP : `` S
18.57   6 /  48733    6 / 905142   CONJP  ->   @DLQ
13.00   7 /  48733   10 / 905142   NP  ->   DT ADJP NP NNX
11.98  20 /  48733   31 / 905142   NP  ->   NNPX JJ
11.46  29 /  48733   47 / 905142   NP  ->   DT ADJP NP
11.14   6 /  48733   10 / 905142   NP  ->   DT `` ADJP NNX ´´
11.14   6 /  48733   10 / 905142   NP  ->   DT `` ADJP ´´ NNX
10.83   7 /  48733   12 / 905142   X  ->   DT ADJP
10.50  13 /  48733   23 / 905142   S  ->   X : S
 9.66  13 /  48733   25 / 905142   NP  ->   NP `` NP ´´
 9.29  11 /  48733   22 / 905142   NP  ->   DT NP NNX PP
 8.67   7 /  48733   15 / 905142   NP  ->   ADJP NP
 8.29  25 /  48733   56 / 905142   NP  ->   DT NP NNX
 6.97   6 /  48733   16 / 905142   ADVP  ->   DT NNX SBAR
 6.84   7 /  48733   19 / 905142   NP  ->   CD IN NNX PP
 6.65  29 /  48733   81 / 905142   NP  ->   LS .
 5.39   9 /  48733   31 / 905142   SBAR  ->   SINV
 5.20   7 /  48733   25 / 905142   S  ->   CC ADVP , S
----- Example ------------------------------------
18.57  S  ->   PP :
                  "To the editor:"
                  "To the editor of New York Times:"
13.00  NP  ->   DT ADJP NP NNX
                  "A former World War 2, command"
                  "The first Rhode Island towns"
11.46  NP  ->   DT ADJP NP
                  "a new Department of Highways"
                  "the all-powerful Los Angeles Times"
11.14  NP  ->   DT `` ADJP NNX ´´
                  "an ``immediate and tangible reality´´"
                  "the ``anti-party group´´"
11.14  NP  ->   DT `` ADJP ´´ NNX
                  "an ``autistic´´ child"
                  "a stair-step´´ plan"
 8.67  NP  ->   ADJP NP
                  "northeast Kansas"
```

Compared to the previous domain (Press: reportage) we can find that there are a number of rules which have double quote (`` or ´´). It may be because, in editorials, citation are used relatively frequently and also quotes are used to place an emphasis on some words. It is interesting to see that there are many rules which have `ADJP`. Also, we find a very domain specific rule `S -> PP :`.

C. Press: Reviews

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
26.27   8 /  30624    9 / 905142   WHADVP  ->   NNPX
25.33  12 /  30624   14 / 905142   NP  ->   NP POS `` NNPX ´´
20.69   7 /  30624   10 / 905142   NP  ->   DT ADJP `` NNPX ´´
18.06  11 /  30624   18 / 905142   S  ->   NP VP @DLQ ADJP
17.24   7 /  30624   12 / 905142   NP  ->   PRP$ ADJP NNPX NNX
15.92   7 /  30624   13 / 905142   S  ->   NP VP @DLQ NP
15.45  46 /  30624   88 / 905142   PP  ->   NNPX NP
15.42  12 /  30624   23 / 905142   VP  ->   NNPX
14.78  16 /  30624   32 / 905142   NP  ->   `` S ´´
13.91   8 /  30624   17 / 905142   NP  ->   NNPX CD PP
11.05  74 /  30624  198 / 905142   NP  ->   `` NNPX ´´
10.89   7 /  30624   19 / 905142   NP  ->   NNPX , PP ,
 9.46   8 /  30624   25 / 905142   ADVP  ->   NNPX
 7.71   6 /  30624   23 / 905142   S  ->   PP , S , CC S
 7.17   8 /  30624   33 / 905142   NP  ->   `` NNPX PP ´´
 6.33   6 /  30624   28 / 905142   NP  ->   NP -LRB- ADJP -RRB-
 6.12   6 /  30624   29 / 905142   NP  ->   NP , NP , SBAR
 6.02  11 /  30624   54 / 905142   NP  ->   NP CC NP PP
 5.44   7 /  30624   38 / 905142   NP  ->   DT NNX ADJP
----- Example ------------------------------------
15.45  PP  ->   NNPX NP
                   Tag Error for Capitalized prepositions
14.78  NP  ->   `` S ´´
                   "``East Meets West´´"
                   "``Get Happy´´"
11.05  NP  ->   `` NNPX ´´
                   "``Panama´´"
                   "``Don Quixote´´"
                   "``Lucia Di Lammermoor"
                   Some tag error for Capitalized words
```

As is written previously, it is obvious that NNPX appears many times in the list.

D. Religion

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
26.83  26 /  31323   28 / 905142   S  ->   NP -RRB- S
25.28  14 /  31323   16 / 905142   NP  ->   NNPX CD : CD
19.26   6 /  31323    9 / 905142   X  ->   X
 8.15  11 /  31323   39 / 905142   NP  ->   DT CD NNPX
 7.88   6 /  31323   22 / 905142   NP  ->   `` NP '' CC `` NP ''
 5.62   7 /  31323   36 / 905142   SBAR  ->   @SNC `` S ''
 5.48  11 /  31323   58 / 905142   X  ->   NP , NP
 5.25   6 /  31323   33 / 905142   S  ->   S , CC ADVP S
----- Example ------------------------------------
26.83  S  ->   NP -RRB- S
                   Listing things (NP is a number)
25.28  NP  -> NNPX CD : CD
                   "St. Peter 1:4"
                   "St. John 3:8"
 8.15  NP  ->   DT CD NNPX
                   "the 70,000,000 Americans"
                   "the Four Gospels"
```

The second partial tree in the list `NP -> NNPX CD : CD` is obviously a domain specific one. It is a citation indicator in the Bible.

E. Skills and Hobbies

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
10.58  22 /  85521   22 / 905142   NP  ->   CD NNX ''
10.58   6 /  85521    6 / 905142   NP  ->   CD TO CD
10.58   6 /  85521    6 / 905142   ADJP  ->   @DLQ RBR PP
10.21  27 /  85521   28 / 905142   S  ->   SBAR :
 9.26   7 /  85521    8 / 905142   NP  ->   DT RB JJ NNX NNX
 7.70   8 /  85521   11 / 905142   VP  ->   VBX ADVP VP
 7.41  14 /  85521   20 / 905142   NP  ->   DT RBS JJ NNX PP
 7.41   7 /  85521   10 / 905142   NP  ->   CD TO CD NNX
 7.33   9 /  85521   13 / 905142   NP  ->   DT NNX CD NNX
 7.06  18 /  85521   27 / 905142   ADJP  ->   DT PP
 7.06  14 /  85521   21 / 905142   X  ->   NNX
 7.06  12 /  85521   18 / 905142   NP  ->   DT SBAR
 7.06  10 /  85521   15 / 905142   NP  ->   NNPX POS
 6.48  30 /  85521   49 / 905142   VP  ->   VB ADJP SBAR
 6.17  14 /  85521   24 / 905142   NP  ->   @DLQ $ CD
 6.17   7 /  85521   12 / 905142   VP  ->   VB NP PP , S
 5.29   8 /  85521   16 / 905142   SQ  ->   SBAR , SQ
----- Example ------------------------------------
10.21  S  ->   SBAR :
                   "How to feed :"
                   "What it does :"
 6.48  VP  ->   VB ADJP SBAR
                   "Be sure that its ventilation is adequate"
                   "Make sure your equipment includes a tripod"
                   "Be sure it is working"
```

F. Popular Lore

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
10.58   8 /  85521    8 / 905142   NP  ->   DT NP POS NNPX
10.58   6 /  85521    6 / 905142   NP  ->   NNX DT NNX PP
 8.23   7 /  85521    9 / 905142   S  ->   NP MD TO VP
 7.94   6 /  85521    8 / 905142   VP  ->   VBG ´´
 7.06   6 /  85521    9 / 905142   X  ->   DT JJR
 7.06   6 /  85521    9 / 905142   SBAR  ->   WHNP , PP , S
 7.06   6 /  85521    9 / 905142   NP  ->   NP , ADVP VP ,
 5.77   6 /  85521   11 / 905142   VP  ->   VBX NP : `` S
 5.77   6 /  85521   11 / 905142   VP  ->   VBG ´´ PP
 5.77   6 /  85521   11 / 905142   S  ->   TO `` VP ´´
 5.64  16 /  85521   30 / 905142   VP  ->   VBG ´´ NP
 5.29   7 /  85521   14 / 905142   ADJP  ->   RB DT
----- Example ----------------------------------
 5.64  VP  ->   VBG ´´ NP
                  Tag error: givin´ -> givin ´
```

The frequency of the partial trees in the list are all very low, and there are little to investigate in this section.

G. Belles Letters, Bibliography, Memories, etc

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
 6.59   8 / 137293    8 / 905142   WHPP  ->   TO SBAR
 6.59   7 / 137293    7 / 905142   NP  ->   CD NNPX CD
 6.59   6 / 137293    6 / 905142   VP  ->   VBX RB NP
 6.04  22 / 137293   24 / 905142   WHPP  ->   @OF SBAR
 5.65   6 / 137293    7 / 905142   S  ->   PP S , SBAR
 5.65   6 / 137293    7 / 905142   ADJP  ->   JJ JJ CC JJ
 5.36  13 / 137293   16 / 905142   PP  ->   @SNC @OF NP
 5.36  13 / 137293   16 / 905142   NP  ->   `` DT NNX PP ''
 5.27   8 / 137293   10 / 905142   CONJP  ->   @DLQ RB
 5.13   7 / 137293    9 / 905142   PP  ->   @SNC NP
 5.04  13 / 137293   17 / 905142   NP  ->   PRP$ JJ CC JJ NNX
 5.02  16 / 137293   21 / 905142   WHPP  ->   IN SBAR
----- Example ------------------------------------
 6.59  WHPP  ->   TO SBAR
                  "to what they mean by concept like liberty"
                  "to what may happen next"
 6.04  WHPP  ->   @OF SBAR
                  "of what it is all about"
                  "of what he had to show his country"
 5.02  WHPP  ->   IN SBAR
                  "from what we encounter on earth"
                  "for what we propose"
                  "beyond what was internationally feasible"
```

Although it is interesting to see that the three frequent examples are in the same type (preposition followed by `SBAR`), these are bit pecurior in this domain. In other domain, the root node is not `WHPP` but `PP`. We will call this kind of error as the annotator's propensity.

H. Miscellaneous

ratio	freq./total (in domain)	freq./total (in corpus)	rule (Example)
16.82	70 / 53814	70 / 905142	S -> NP . S
16.82	17 / 53814	17 / 905142	S -> -LRB- VP . -RRB-
16.82	13 / 53814	13 / 905142	VP -> -LRB- VBG NP -RRB-
16.82	7 / 53814	7 / 905142	NP -> DT CD CC ADJP
16.82	7 / 53814	7 / 905142	NP -> DT ADJP PP NNX PP
16.82	7 / 53814	7 / 905142	NP -> -LRB- NNPX CD -RRB-
15.95	55 / 53814	58 / 905142	NP -> -LRB- NNX -RRB-
15.34	31 / 53814	34 / 905142	ADVP -> NP
14.72	7 / 53814	8 / 905142	NP -> ADJP ADJP NNPX
14.42	6 / 53814	7 / 905142	NP -> DT ADJP ADJP NNPX
14.42	6 / 53814	7 / 905142	NP -> ADJP CC ADJP NNX
14.23	11 / 53814	13 / 905142	PP -> -LRB- IN NP -RRB-
14.11	73 / 53814	87 / 905142	NP -> -LRB- LS -RRB-
14.02	15 / 53814	18 / 905142	NP -> ADJP ADJP NNX NNX
14.02	10 / 53814	12 / 905142	NP -> ADJP NNX CC NNX PP
13.46	8 / 53814	10 / 905142	NP -> NP NP , PP
12.62	313 / 53814	417 / 905142	ADJP -> NNPX
12.61	27 / 53814	36 / 905142	NP -> ADJP ADJP NNX PP
12.61	9 / 53814	12 / 905142	NP -> -LRB- NNPX -RRB-
12.15	13 / 53814	18 / 905142	VP -> NP VP
11.94	22 / 53814	31 / 905142	NP -> ADJP PP
11.91	17 / 53814	24 / 905142	NP -> CD SYM
11.64	9 / 53814	13 / 905142	CONJP -> CC
11.56	11 / 53814	16 / 905142	NP -> ADJP NNX NNX NNX
11.21	52 / 53814	78 / 905142	PP -> @SNC S
11.21	6 / 53814	9 / 905142	S -> NP VBX ADVP VP NP
11.21	6 / 53814	9 / 905142	NP -> NNX CD , PP
11.21	6 / 53814	9 / 905142	NP -> ADJP NNX CC NNX NNX
10.28	11 / 53814	18 / 905142	NP -> NP . NP
10.09	15 / 53814	25 / 905142	SBAR -> WDT S
9.74	22 / 53814	38 / 905142	ADJP -> JJ NNX
9.28	16 / 53814	29 / 905142	NP -> ADJP NNX NNX PP
9.17	6 / 53814	11 / 905142	NP -> -LRB- CD -RRB-
8.78	24 / 53814	46 / 905142	NP -> ADJP NNX CC NNX
8.78	12 / 53814	23 / 905142	NP -> NP NP PP
8.78	12 / 53814	23 / 905142	NP -> @DLQ ADJP NNX PP
8.41	7 / 53814	14 / 905142	S -> NP NP
7.85	84 / 53814	180 / 905142	NP -> ADJP NNX NNX
7.85	56 / 53814	120 / 905142	NP -> ADJP

```
 7.85   7 /  53814   15 / 905142   X  ->   PP CC ADVP
 7.82  79 /  53814  170 / 905142   NP  ->   ADJP ADJP NNX
 7.60  14 /  53814   31 / 905142   NP  ->   NNPX NNX NNPX
 7.48  36 /  53814   81 / 905142   NP  ->   @DLQ ADJP NNX
 7.42  15 /  53814   34 / 905142   NP  ->   NNX CD PP
 7.21   6 /  53814   14 / 905142   NP  ->   DT NNX CC NNX PP PP
 7.08  16 /  53814   38 / 905142   NP  ->   DT ADJP NNPX PP
 6.61  11 /  53814   28 / 905142   S  ->   NP , MD VP
 6.56  23 /  53814   59 / 905142   NP  ->   DT ADJP ADJP NNX PP
 6.54   7 /  53814   18 / 905142   NP  ->   ADJP NNX S
 6.49  98 /  53814  254 / 905142   ADJP  ->   NNX
 6.45 202 /  53814  527 / 905142   NP  ->   ADJP NNX PP
 6.41  16 /  53814   42 / 905142   NP  ->   DT ADJP NNX NNX PP
 6.27  19 /  53814   51 / 905142   NP  ->   NP ADVP VP
 6.08  17 /  53814   47 / 905142   NP  ->   CD ADJP NNX
 6.07  13 /  53814   36 / 905142   NP  ->   PRP$ ADJP ADJP NNX
 5.98  27 /  53814   76 / 905142   NP  ->   DT ADJP NNX PP PP
 5.89  14 /  53814   40 / 905142   NP  ->   PRP$ ADJP NNX PP
 5.89   7 /  53814   20 / 905142   NP  ->   DT ADJP NNX NNX NNX
 5.85   8 /  53814   23 / 905142   NP  ->   DT NNPX PP PP
 5.82   9 /  53814   26 / 905142   SBAR  ->   WHNP
 5.68 689 /  53814 2041 / 905142   NP  ->   ADJP NNX
 5.61   7 /  53814   21 / 905142   PP  ->   RB ADVP
 5.43  60 /  53814  186 / 905142   NP  ->   DT ADJP NNX NNX
 5.41  28 /  53814   87 / 905142   NP  ->   NP , ADVP
 5.31   6 /  53814   19 / 905142   ADJP  ->   ADVP JJ SBAR
 5.25  49 /  53814  157 / 905142   NP  ->   DT ADJP ADJP NNX
 5.22   9 /  53814   29 / 905142   NP  ->   NP , NP , NP , NP ,
                                            NP , CC NP
 5.10  10 /  53814   33 / 905142   NP  ->   CD CC CD
 5.05   6 /  53814   20 / 905142   NP  ->   DT ADJP NNX SBAR
----- Example ------------------------------------
16.82  S  ->   NP . S
                   title of section
16.82  S  ->   -LRB- VP . -RRB-
                   "(See Source of Data, below...)"
                   "(Multiply the result obtained in item 3...)"
16.82  VP  ->   -LRB- VBG NP -RRB-
                   "(excluding coal and lignite)"
                   "(excluding export)"
16.82  NP  ->   DT CD CC ADJP
                   "the one hundred and eighty-fifth"
```

```
16.82  NP  ->   DT ADJP PP NNX PP
                   "the average per capita income of the U.S."
16.82  NP  ->   -LRB- NNPX CD -RRB-
                   "(August 31)"
                   Tagging mistakes capitalize month names
15.95  NP  ->   -LRB- NNX -RRB-
                   "subsection (A) above"
15.34  ADVP  ->   NP
                   "today"
                   "each September"
                   Tagging propensity(?)
14.72  NP  ->   ADJP ADJP NNPX
                   Tagging mistakes capitalize words
14.42  NP  ->   DT ADJP ADJP NNPX
                   Tagging mistakes capitalize words
14.23  PP  ->   -LRB- IN NP -RRB-
                   "(in U.S.)"
                   "(in 1938-1939)"
14.11  NP  ->   -LRB- LS -RRB-
                   enumeration
12.62  ADJP  ->   NNPX
                   Tagging mistakes capitalized word
12.61  NP  ->   -LRB- NNPX -RRB-
                   "the North Atlantic Treaty Organization (NATO)"
11.21  PP  ->   @SNC S
                   tagging error
```

We can find more number of `ADJP` appears than that in other sections. By observing the instances, we can conclude that this is caused by the annotator's propensity. Even a simple `JJ NN` like *warm heart*, annotators annotate like `(NP (ADJP (JJ warm)) (NN heart))` which is usually `(NP (JJ warm) (NN heart))`.

J. Learned

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
 6.51  28 / 139137   28 / 905142   NP  ->   CD : CD
 6.51  20 / 139137   20 / 905142   NP  ->   NNX :
 6.51  11 / 139137   11 / 905142   NP  ->   NNX -LRB- CD -RRB-
 6.51  10 / 139137   10 / 905142   NP  ->   NNX PP :
 6.51   8 / 139137    8 / 905142   NP  ->   NP -LRB- NP -RRB-
                                           -LRB- NP -RRB-
 6.51   8 / 139137    8 / 905142   NP  ->   NP -LRB- -RRB-
 6.51   8 / 139137    8 / 905142   NP  ->   NNX CD , CC CD
 6.51   8 / 139137    8 / 905142   NP  ->   NNX -LRB- NNX -RRB-
 6.51   7 / 139137    7 / 905142   NP  ->   IN
 6.51   6 / 139137    6 / 905142   NP  ->   NP NP :
 6.51   6 / 139137    6 / 905142   NP  ->   NNX NNX CD .
 6.51   6 / 139137    6 / 905142   NP  ->   CD : NNX
 6.51   6 / 139137    6 / 905142   ADJP  ->   JJ : CC JJ
 6.22  44 / 139137   46 / 905142   S  ->   S -LRB- NP -RRB-
 5.85   9 / 139137   10 / 905142   NP  ->   SBARQ
 5.78   8 / 139137    9 / 905142   NP  ->   NNX CD : CD
 5.69   7 / 139137    8 / 905142   SBARQ  ->   WHNP SQ . .
 5.58   6 / 139137    7 / 905142   NP  ->   CD : CD NNX
 5.49  27 / 139137   32 / 905142   NP  ->   -LRB- NP -RRB-
 5.34  32 / 139137   39 / 905142   NP  ->   CC
 5.30  22 / 139137   27 / 905142   NP  ->   NP CC
 5.20   8 / 139137   10 / 905142   NP  ->   DT NNX NNX CC JJ NNX
----- Example ------------------------------------
 6.51  NP  ->   CD : CD
                  "titer of 1 : 512 in saline"
                  "averaging around 60 - 80"
 6.22  S  ->   S -LRB- NP -RRB-
                  Sentence followed by name and year in bracket
                  Sentence followed by figure name in bracket
 5.49  NP  ->   -LRB- NP -RRB-
                  Name in bracket
                  Reference in bracket
 5.34  NP  ->   CC
                  Special string (**f) is deleted in sentences
 5.30  NP  ->   NP CC
                  Special string (**f) is deleted in sentences
```

There are many numbers `CD` in the list. Also there are several kinds of domain specific partial trees, like `S -> S -LRB- NP -RRB-` or NP -LRB- NP -LRB- as shown in the example.

K. General Fiction

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
11.58   7 /  54721   10 / 905142   NP  ->   PRP S
11.03   6 /  54721    9 / 905142   S  ->   ADVP S : : S
10.75  13 /  54721   20 / 905142   S  ->   PP S , CC S
 8.82   8 /  54721   15 / 905142   S  ->   S , CC PP S
 8.10  24 /  54721   49 / 905142   PP  ->   RP
 7.72   7 /  54721   15 / 905142   VP  ->   VP CC RB VP
 7.63   6 /  54721   13 / 905142   S  ->   NP VP @DLQ NP
 7.44   9 /  54721   20 / 905142   S  ->   INTJ
 7.14  19 /  54721   44 / 905142   PP  ->   RP NP
 6.74  11 /  54721   27 / 905142   PP  ->   RB
 6.36  10 /  54721   26 / 905142   S  ->   ADVP S , CC S
 5.67  12 /  54721   35 / 905142   PP  ->   @DLQ IN NP
 5.51   6 /  54721   18 / 905142   S  ->   NP VP @DLQ ADJP
 5.34  10 /  54721   31 / 905142   S  ->   NP VP VP
----- Example ------------------------------------
10.75  S  ->   PP S , CC S
                   "In the half darkness the banisters gleamed,
                    and the hall seemed enormous"
                   "Of course I had to give her Eileen´s address,
                    but she never came near us"
 8.82  S  ->   S , CC PP S
                   "There was a glass pane in the front door,
                    and through this he could see into..."
 8.10  PP  ->   RP
                   "extend out"
                   "go off"
 7.72  VP  ->   VP CC RB VP
                   "looked around the room and then called out..."
                   "snorted and then laughed aloud"
 5.67  PP  ->   @DLQ IN NP
                   "just for his private satisfaction"
                   "only in the next day or two"
 5.34  S  ->   NP VP VP
                   Tagging mistake for passive, past perfect
```

At a glance, we can tell that there are many partial trees whose heads are S's, compare to many NP's in the non-fiction section. This is the case throughout the fiction sections. By the examples, it is intuitively true to say these are domain

specific (fiction style sentences). For example, the cases in which the first example partial tree appears in news paper article, may be limited to such narrative sentences.

L. Mystery and Detective Fiction

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
14.28   8 /  46089   11 / 905142   SQ  ->   S , SQ
12.00  11 /  46089   18 / 905142   VP  ->   VBX RB (=NEG)
11.78   6 /  46089   10 / 905142   S  ->   NP VBX RB VP , NP
10.71   6 /  46089   11 / 905142   ADVP  ->   RB , RB
 9.82   8 /  46089   16 / 905142   S  ->   SQ ´´
 9.82   7 /  46089   14 / 905142   S  ->   S ´´ :
 9.82   6 /  46089   12 / 905142   VP  ->   VBX RB PP
 9.16   7 /  46089   15 / 905142   S  ->   `` ADJP ´´ , S
 8.21  23 /  46089   55 / 905142   VP  ->   VBX , `` S
 8.09  14 /  46089   34 / 905142   VP  ->   VP , ADVP VP
 8.09   7 /  46089   17 / 905142   S  ->   `` X ´´ , S
 6.93   6 /  46089   17 / 905142   NP  ->   NNX JJ
 6.31   9 /  46089   28 / 905142   X  ->   VBX RB VP
 6.04   8 /  46089   26 / 905142   S  ->   X ´´
 5.95  10 /  46089   33 / 905142   S  ->   NP VP , ADJP
 5.61   8 /  46089   28 / 905142   S  ->   `` NP ´´ , S
 5.61   6 /  46089   21 / 905142   ADJP  ->   JJ RB
 5.38  17 /  46089   62 / 905142   PP  ->   @DLQ PP
 5.20  62 /  46089  234 / 905142   PP  ->   RB PP
----- Example ------------------------------------
14.28  SQ  ->   S , SQ
                  tag questions
12.00  VP  ->   VBX RB (=NEG)
                  negation
 9.82  S  ->   SQ ´´
                  the last sentence in quote
 9.16  S  ->   `` ADJP ´´ , S
                  "``All right´´, Calenda said"
                  "``Cool´´, I told him"
 8.21  VP  ->   VBX , `` S
                  upto the first sentence in quote
 8.09  VP  ->   VP , ADVP VP
                  "said, then changed the subject"
                  "frowned slightly, then became sad and moody"
 5.95  S  ->   NP VP , ADJP
                  "The kick came, sudden and vicious but short"
 5.61  S  ->   `` NP ´´ , S
                  "``My God´´, he muttered"
                  "``Nice place´´ I told him"
```

```
5.20  PP  ->   RB PP
                  "back to New York"
                  "aside from everything else"
```

There are a lot of partial trees with double quotes. As we can see from the examples, there are conversations or a part of a conversation. It is interesting to see that there is no rule with double quote in the previous section (general fiction), when it is compared with this section.

M. Science Fiction

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
17.89   7 /  11068   32 / 905142   S  ->   S , SINV
10.22   8 /  11068   64 / 905142   S  ->   SBARQ ´´ .
 7.84   7 /  11068   73 / 905142   S  ->   SQ ´´ .
 6.96   8 /  11068   94 / 905142   SQ  ->   VP
 6.73   7 /  11068   85 / 905142   SBAR  ->   ADJP S
 6.29  20 /  11068  260 / 905142   SINV  ->   VP NP
----- Example ------------------------------------
17.89  S  ->   S , SINV
                  "``Forgive me, Sandalphon´´, said Hal"
                  "``sentence´´, remarked Helva"
 6.73  SBAR  ->   ADJP S
                  "How long he had been there"
 6.29  SINV  ->   VP NP
                  "Said the whining voice"
                  "Asked Mercer"
```

There is little to see in this section. All but the last one have very low frequencies. The last one is introducer of a conversation, which is very common in fiction sections.

N. Adventure and Western Fiction

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
14.59  45 /  55831   50 / 905142   VP  ->   VBX RB
12.97   8 /  55831   10 / 905142   VP  ->   VBX RB PP
12.61   7 /  55831    9 / 905142   S  ->   `` INTJ , INTJ '' , S
11.35  21 /  55831   30 / 905142   VP  ->   VP , RB VP
10.46  20 /  55831   31 / 905142   X  ->   @DLQ
 9.98   8 /  55831   13 / 905142   S  ->   S , RB S
 8.11   7 /  55831   14 / 905142   S  ->   S '' :
 8.11   6 /  55831   12 / 905142   VP  ->   NP
 7.48   6 /  55831   13 / 905142   S  ->   S , CC RB S
 7.48   6 /  55831   13 / 905142   S  ->   INTJ .
 6.95  12 /  55831   28 / 905142   S  ->   `` NP '' , S
 6.68   7 /  55831   17 / 905142   VP  ->   VBX NP PP , S
 6.48   6 /  55831   15 / 905142   S  ->   `` VP
 6.34   9 /  55831   23 / 905142   VP  ->   VBX NP PP ADVP
 6.30  14 /  55831   36 / 905142   NP  ->   @DLQ SBAR
 6.05  22 /  55831   59 / 905142   S  ->   RB S
 5.99  17 /  55831   46 / 905142   S  ->   VP , NP
 5.67 149 /  55831  426 / 905142   S  ->   `` S '' , S
 5.40   6 /  55831   18 / 905142   VP  ->   VBX ADVP , S
 5.19   8 /  55831   25 / 905142   VP  ->   VBG NP PRT
 5.16   7 /  55831   22 / 905142   X  ->   DT
 5.12   6 /  55831   19 / 905142   VP  ->   RB VBN PP
 5.10  11 /  55831   35 / 905142   ADVP  ->   ADVP RB
 5.04  14 /  55831   45 / 905142   NP  ->   DT RB JJ NNX
----- Example ------------------------------------
14.59  VP  ->   VBX RB
                  "said quickly"
                  "whistled tunelessly"
11.35  VP  ->   VP , RB VP
                  "swirled low to the ground, then rose with..."
                  "waited for a long moment, then asked the..."
                  Similar structure in L (RB<->ADJP)
 6.95  S  ->   `` NP '' , S
                  "``Your choice;;,he said briefly"
                  "``Temper, temper'', Mrs.Forsythe said"
                  Same structure in L
 6.30  NP  ->   @DLQ SBAR
                  "all they wanted"
                  "all that had happened"
```

```
6.05  S  ->   RB S
                  `then´ followed by a sentence
5.67  S  ->   `` S ´´ , S
                  "``I will see´´, Morgan said"
                  "``I loved this valley´´ he whispered huskily"
5.04  NP  ->   DT RB JJ NNX
                  "a very strong woman"
                  "that bitterly cold day"
```

Again, there are a lot of conversations. A relatively frequent appearance of `RB` may indicate that there are a lot of adverb modifiers in this section.

P. Romance and Love Story

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
15.99   7 /  56599    7 / 905142   S  ->   CC SBARQ
15.99   6 /  56599    6 / 905142   S  ->   `` NP VP , NP ´´
12.23  13 /  56599   17 / 905142   S  ->   SQ S
11.99   6 /  56599    8 / 905142   S  ->   `` VP , NP ´´
11.42  15 /  56599   21 / 905142   S  ->   SBARQ S
11.42  10 /  56599   14 / 905142   S  ->   `` NP VBX RB VP ´´
10.66   6 /  56599    9 / 905142   WHNP  ->   WP DT NNX
 8.72  18 /  56599   33 / 905142   INTJ  ->   `` UH ´´
 8.72   6 /  56599   11 / 905142   SBARQ  ->   `` WHNP SQ ´´ . .
 8.47   9 /  56599   17 / 905142   S  ->   `` INTJ , S ´´
 8.00  16 /  56599   32 / 905142   S  ->   `` NP VBX VP ´´
 8.00   6 /  56599   12 / 905142   CONJP  ->   CC @DLQ
 7.53   8 /  56599   17 / 905142   ADJP  ->   ADVP RB
 7.38   6 /  56599   13 / 905142   S  ->   `` S , S ´´
 7.37  65 /  56599  141 / 905142   S  ->   `` NP VP ´´
 7.02  18 /  56599   41 / 905142   S  ->   S , CC S , CC S
 6.95  10 /  56599   23 / 905142   ADVP  ->   VB
 6.33  38 /  56599   96 / 905142   VP  ->   VBX , S
 6.09   8 /  56599   21 / 905142   NP  ->   PRP$ JJ , JJ NNX
 6.00   6 /  56599   16 / 905142   SQ  ->   MD RB NP VP
 6.00   6 /  56599   16 / 905142   S  ->   `` PP S ´´
 5.95 180 /  56599  484 / 905142   NP  ->   NP PP
 5.64  12 /  56599   34 / 905142   SQ  ->   VBX RB NP VP
 5.64   6 /  56599   17 / 905142   VP  ->   VBX , NP
 5.56   8 /  56599   23 / 905142   S  ->   CC S CC S
 5.51  20 /  56599   58 / 905142   VP  ->   VBX RB NP
 5.33  15 /  56599   45 / 905142   S  ->   `` VP ´´
 5.33   8 /  56599   24 / 905142   S  ->   S CC S CC S
 5.33   7 /  56599   21 / 905142   SBAR  ->   RB @SNC S
 5.05   6 /  56599   19 / 905142   ADJP  ->   JJ CC JJ CC JJ
 5.03  11 /  56599   35 / 905142   S  ->   `` NP MD VP ´´
----- Example ------------------------------------
15.99  S  ->   `` NP VP , NP ´´
                   "``That is n´t like you, Janice´´"
12.23  S  ->   SQ S
                   "``Does this bother you´´?? I said"
                   "``Is it a woman´´?? She asked gently"
11.99  S  ->   `` VP , NP ´´
                   "``Wait and see, Dad´´"
```

```
11.42  S  ->   SBARQ S
                   "``What´s the matter´´?? She asked suddenly"
 8.72  INTJ  ->   `` UH ´´
                   "``Yes´´"
                   "``Uhhu´´"
 7.37  S  ->   `` NP VP ´´
                   "``You said a mouthful´´"
 7.02  S  ->   S , CC S , CC S
                   sequence of sentences by `and´ and `but´
 6.33  VP  ->   VBX , S
                   "said, ``Answer me properly, Spencer´´"
 5.95  NP  ->   NP PP
                   "a signal from Spencer"
                   "the Christmas card with no name on it"
 5.64  SQ  ->   VBX (=AUX) RB (=NEG) NP VP
                   Negative questions
 5.33  S  ->   `` VP ´´
                   "``Pack your clothes´´"
                   "``Be my guest´´"
```

Obviously there are a lot of rules with open and close double quotation marks. It indicates that an utterance consists of only a single sentence. So we can say that, also from the evidence of the examples, in this section there are many utterances, and that these are a simple sentence.

R. Humor

```
ratio freq./total   freq./total      rule (Example)
      (in domain)   (in corpus)
 6.92   6 /  16687   47 / 905142   NP  ->   DT ADJP NP
 6.78   7 /  16687   56 / 905142   NP  ->   PRP @DLQ
 5.67   7 /  16687   67 / 905142   PP  ->   IN `` NP ´´
----- Example ------------------------------------
 5.67  PP  ->   IN `` NP ´´
                  "on ``The Civic Sprit of the Southland´´"
                  "as ``off-Broadway´´"
```

There is nothing to say. The size of this section is small.

Bibliography

[Biber 93] Douglas Biber: "Using Register-Diversified Corpora for General Language Studies" *Journal of Computer Linguistics Vol.19, Num 2*, pp219-241. (1993)

[Black et al. 91] E.Black, S.Abney, D Flickenger, C.Gdaniec, R.Grishman, P.Harrison, D.Hindle, R.Ingria, F.Jelinek, J.Klavans, M.Liberman, M.Marcus, S.Roukos, B.Santorini, T.Strzalkowski: "A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars" *Proceedings of the 4th DARPA Speech and Natural Language Workshop* (1991)

[Black et al. 93] Ezra Black, Fred Jelinek, John Lafferty, David Magerman, Robert Mercer and Salim Roukos: "Towards History-Based Grammars: Using Richer Models for Probabilistic Parsing" *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics (ACL)* (1993)

[Black 93] Ezra Black "Parsing English By Computer: The State Of the Art" *Proceedings of the ATR International Workshop on Speech Translation* (1993)

[Bod 92] Rens Bod "Data Oriented Parsing (DOP)" *Proceedings of the 14th Conference on Computational Linguistics (COLING)* (1992)

[Bod 93] Rens Bod "Using an Annotated Corpus as a Stochastic Grammar" *Proceedings of the 6th Conference of the European Chapter of the Association for Computational Linguistics (EACL)* (1993)

[Bod 95] Rens Bod "Enriching Linguistics with Statistics: Performance Models of Natural Language" *Doctoral thesis, Universiteit van Amsterdam* (1995)

[Bod and Scha 96] Rens Bod and Remko Scha "Data-Oriented Language Processing: An Overview" *ftp://ftp.fwi.uva.nl/pub/theory/illc/researchreports/LP-96-13.text.ps.gz* (1996)

[Brill et al. 90] Eric Brill, David Magerman, Mitchell Marcus and Beatrice Santorini "Deducting Linguistic Structure from the Statistics of Large Corpora" *Proceeding of the June DARPA Speech and Natural Language workshop. Hidden Valley, Pennsylvania.* (1990)

[Brill 93] Eric Brill "Automatic Grammar Induction and Parsing Free Text: A Transformation-Based Approach" *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics (ACL)* (1993)

[Briscoe and Carroll 93] Ted Briscoe and John Carroll "Generalized Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars" *Journal of Computational Linguistics Vol.19,No.1* (1993)

[Bross et al. 72] I D J Bross and P A Shapiro and B B Anderson: "How information is carried in scientific sub-languages" *Science", Vol.176, 1303–1307* (1972)

[Carroll and Briscoe 96] John Carroll and Ted Briscoe: "Apportioning development effort in a probabilistic LR parsing system through evaluation" *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* (1996)

[Charniak 93] Eugene Charniak: *Statistical Language Learning* The MIT Press (1993)

[Charniak 97] Eugene Charniak: "Statistical Parsing with a Context-free Grammar and Word Statistics" *Proceedings of the 14th National Conference on Artificial Inteligence* (1997)

[Chitrao and Grishman 90] Mahesh Chitrao, Ralph Grishman: "Statistical Parsing of Message" *Proceeding of the June DARPA Speech and Natural Language workshop. Hidden Valley, Pennsylvania.* (1990)

[Chitrao 90] Mahesh Chitrao "Statistical Techniques for Parsing Messages" *Doctoral thesis, New York University* (1990)

[CL Special Issue I 93] "Special Issue on Using Large Corpora: I" *Journal of Computational Linguistics Vol.19,Num.1* (1993)

[CL Special Issue II 93] "Special Issue on Using Large Corpora: II" *Journal of Computational Linguistics Vol.19,Num.2* (1993)

[Collins 96] Michael Collins "A New Statistical Parser Based on Bigram Lexical Dependencies" *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)* (1996)

[Collins 97] Michael Collins "Three Generative, Lexicalised Models for Statistical Parsing" *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL)* (1997)

[Cormen et.al 90] Thomas Cormen, Charles Leiserson and Ronald Rivest: *Introduction to Algorithms* The MIT Press (1990)

[Francis and Kucera 64/79] W. Nelson Francis and Henry Kucera: "Manual of information to accompany A Standard Corpus of Present-Day Edited American English for use with Digital Computers" *Brown University, Department of Linguistics* (1964/1979)

[Fujisaki 84] Tetsunosuke Fujisaki: "A Stochastic Approach to Sentence Parsing" *Proceedings of the 10th Conference on Computational Linguistics (COLING)* (1984)

[Fujisaki et al. 89] Tetsunosuke Fujisaki, Fred Jelinek, J.Cocke, Ezra Black and T.Nishino: "Probabilistic Parsing Method for Sentence Disambiguation" *Proceedings of 1st International Workshop on Parsing Technologies* (1989)

[Fujio and Matsumoto 97] Masakazu Fujio and Yuji Matsumoto: "Statistic Based Dependency Analysis" *Natural Language group - Information Processing Society of Japan (In Japanese)* (1997)

[Garside and Leech 85] Roger Garside, Fanny Leech: "A Probabilistic Parser" *Proceedings of the 2nd Conference of the European Chapter of the Association for Computational Linguistics (EACL)* (1985)

[Gnanadesikan 77] R.Gnanadesikan: *Methods for Statistical Data Analysis of Multivariate Observations* John Wiley & Sons, Inc (1977)

[Grishman et al. 84a] Ralph Grishman, Ngo Than Nhan, Elaine Marsh and Lynette Hirschmann: "Automated determination of sublanguage syntactic usage" *Proceedings of the 10th Conference on Computational Linguistics (COLING)* (1984)

[Grishman et al. 84b] Ralph Grishman, Ngo Than Nhan and Elaine Marsh: "Tuning Natural Language Grammers for New Domains" *Conference on Intelligent Systems and Machines, Rochester, Minnesota, pp342-346* (1984)

[Grishman et al. 86] Ralph Grishman, Lynette Hirschman and Ngo Than Nhan: "Discovery Procedures for Sublanguage Selectional Patterns: Initial Experiments" *Journal of Computational Linguistics Vol.12 No.3* (1986)

[Grishman 86] Ralph Grishman: *Computational Linguistics: An introduction* Cambridge University Press (1986)

[Grishman and Kittredge 86] Ralph Grishman and Richard Kittredge (ed.): *Analyzing Language in Restricted Domains: Sublanguage Description and Processing* Lawrence Erlbaum Associates, Publishers (1986)

[Grishman and Sterling 92] Ralph Grishman and John Sterling: "Acquisition of Selectional Patterns" *Proceedings of the 14th Conference on Computational Linguistics (COLING)* (1992)

[Grishman and Sterling 94] Ralph Grishman, John Sterling: "Generalizing Automatically Generated Selectional Patterns" *Proceedings of the 15th Conference on Computational Linguistics (COLING)* (1994)

[Grishman et al. 94] Ralph Grishman, Catherine Macleod and Adam Meyers: "Comlex Syntax: Building a Computational Lexicon" *Proceedings of the 15th Conference on Computational Linguistics (COLING)* (1994)

[Harris 68] Zellig Harris: *Mathematical Structures of Language* John Wiley and Sons, New York (1968)

[Hirshman et al. 75] Lynette Hirschman, Ralph Grishman and Naomi Sager: "Grammatically based automatic word class formation" *Information Processing and Management vol.11, pp.39-57.* (1975)

[Inui et al. 97] Kentaro Inui, Virach Sornlertlamvanich, Hozumi Tanaka and Takenobu Tokunaga: "A new Formalization of Probabilistic GLR Parsing" *Proceedings of 5th International Workshop on Parsing Technologies* (1997)

[Isabelle 84] P Isabelle: "Machine Translation at the TAUM group" *in King,M. (ed.) Machine Translation Today: The State of the Art, Edinburgh University Press* (1984)

[Jelinek 91] Fred Jelinek, B Merialdo, S Roukos, and M Strauss: "A Dynamic Language Model for Speech Recognition" *Proceedings of DARPA Speech and Natural Language Workshop* (1991)

[Jones and Somers 97] Daniel Jones and Harold Somers (ed.): *New Methods in Language Processing* UCL Press (1982)

[Joshi et al. 75] Aravind Joshi, Leon Levy and M Takahashi: "Tree Adjunct grammars" *Journal of the Computer and System Sciences, 10* (1975)

[Joshi 87] Aravind Joshi: "An introduction to Tree Adjoining Grammars" *In A.Manaster-Ramer (ed.), Mathematics of Language, John Benjamins, Amsterdam* (1987)

[Joshi 94] Aravind Joshi and B Srinivas: "Disambiguation of Super Parts of Speech (or Supertags): Almost Parsing" *Proceedings of the 15th Conference on Computational Linguistics (COLING)* (1994)

[Joshi 96] Aravind Joshi: "NLP Research at UPenn" *JSPS workshop; Tokyo* (1996)

[Karlgren and Cutting 94] Jussi Karlgren and Douglass Cutting: "Recognizing Text Genres with Simple Metrics Using Discriminant Analysis" *Proceedings of the 15th Conference on Computational Linguistics (COLING)* (1994)

[Katz 87] Slava 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* (1987)

[Kittredge 82] Richard Kittredge, John Lehrberger (ed.): *Sublanguage: Studies of Language in Restricted Semantic domains* Walter de Gruyter, Berlin (1982)

[Krenn and Samuelsson 97] Brigitte Krenn and Christer Samuelsson: "The Linguist's Guide to Statistics" *http://www.coli.uni-sb.de/~krenn/stat_nlp.ps* (1997)

[Kurohashi and Nagao 97] Sadao Kurohashi and Makoto Nagao: "Building a Japanese Parsed Corpus while Improving the Parsing System" *Proceedings of the Natural Language Pacific Rim Symposium* (1997)

[Kurohashi 96] Sadao Kurohashi: "Japanese syntactic analyser: KNP" *Kyoto University* (1996)

[Leech et al. 83] Geoffrey Leech, Roger Garside, E. Atwell: "The Automatic Grammatical Tagging of the LOB Corpus" *ICAME News 7, pp13-33* (1983)

[Leech et al. 87] Geoffrey Leech, Roger Garside, G. Sampson: *The computational analysis of English; A corpus based approach* London, Longman (1987)

[Lehrberger 82] John Lehrberger: "Automatic Translation and the Concept of Sublanguage" *in Sublanguage: Study of Language in Restricted Semantic Domains* (1982)

[Magerman and Marcus 91a] David Margerman and Mitchell Marcus: "Pearl: A Probabilistic Chart Parser" *Proceedings of the 5th Conference of the European Chapter of the Association for Computational Linguistics (EACL)* (1991)

[Magerman and Marcus 91b] David Margerman and Mitchell Marcus: "Parsing the Voyager domain using Pearl" *Proceedings of Speech and Natural Language Workshop, pp 231-236* (1991)

[Magerman and Weir 92] David Magerman and C.Weir: "Efficiency, Robustness and Accuracy in Picky Chart Parsing" *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics (ACL)* (1992)

[Magerman 95] David Magerman: "Statistical Decision-Tree Models for Parsing" *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL)* (1995)

[Marcus 90] Mitchell Marcus: "Tutorial on Tagging and Processing Large Textual Corpora" *Presented at the 28th Annual Meeting of the Association for Computational Linguistics (ACL)* (1990)

[Marcus 93] Mitchell Marcus, Beatrice Santorini and Mary Marcinkiewicz: "Building a Large Annotated Corpus of English: The Penn TreeBank" *Journal of Computational Linguistics, Vol.19 No.1, pp313-330.* (1993)

[Marcus 96] Mitchell Marcus, Beatrice Santorini and Mary Marcinkiewicz: "Building a large annotated corpus of English: the Penn Treebank" *in the distributed Penn Tree Bank Project II CD-ROM*, Linguistic Data Consortium, University of Pennsylvania. (1996)

[Matsumoto et al. 97] Yuuji Matsumoto, Sadao Kurohashi, Osamu Yamaji, Yuu Taeki and Makoto Nagao: "Japanese morphological analyzing System: JUMAN" *Kyoto University and Nara Institute of Science and Technology* (1997)

[McEnery and Wilson 96] Tony McEnery and Andrew Wilson: *Corpus Linguistics* Edinburgh University Press (1996)

[Ostendorf et al. 95] Mari Ostendorf, Fred Richardson, Rukemini Iyer, Ashvin Kannan, Orith Ronen and Rebecca Bates: "The 1994 BU NAB News Benchmark System" *Proceedings of the ARPA Spoken Language Systems Technology Workshop* (1995)

[Pereira and Schabes 92] Fernando Pereira and Yves Schabes: "Inside-Outside Reestimation from Partially Bracketed Corpora" *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics (ACL)* (1992)

[Ratnaparkhi 97] Adwait Ratnaparkhi: "A Linear Observed Time Statistical Parser Based on Maximum Entropy Models" *Proceedings of Empirical Method for Natural Language Processings* (1997)

[Rayner 88] Manny Rayner: "Applying Explanation-Based Generalization to Natural-Language Processing" *Proceedings of the International Conference on Fifth Generation Computer Systems* (1988)

[Rayner 96] Manny Rayner and David Cater: "Fast Parsing Using Pruning and Grammar Specialization" *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)* (1996)

[Resnik 92] Phillip Resnik: "Probablistic Tree-Adjoining Grammar as a Framework for Statistical Natural Language Processing" *Proceedings of the 14th Conference on Computational Linguistics (COLING)* (1992)

[Rosenfeld 94] Ronald Rosenfeld: "A Hybrid Approach to Adaptive Statistical Language Modeling" *Proceedings of Human Language Technology Workshop* (1994)

[Russell and Norvig 95] Stuart Russell and Peter Norvig: "Artificial Intelligence: A Modern Approach" *Prentice Hall Series in Artificial Intelligence* (1995)

[Samuelsson 94a] Christer Samuelsson: "Fast Natural-Language Parsing Using Explanation-Based Learning" *doctoral thesis, The Royal Institute of Technology and Stockholm University* (1994)

[Samuelsson 94b] Christer Samuelsson: "Grammar Specialization through Entropy Thresholds" *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics (ACL)* (1994)

[Sankar et al. 96] A.Sankar, A.Stolcke, T.Chung, L.Neumeyer, M.Weintraub, H.Franco and F.Beaufays: "Noise-Resistant Feature Extraction and Model Training for Robust Speech Recognition" *DARPA Speech Recognition Workshop* (1996)

[Schabes et al. 88] Yves Schabes, Anne Abeille and Aravind Joshi: "Parsing strategies with lexicalized grammars: Application to tree adjoining grammars" *Proceedings of the 12th Conference on Computational Linguistics (COLING)* (1988)

[Schabes 90] Yves Schabes: "Mathematical and Computational Aspects of Lexicalized Grammars" *PhD thesis, University of Pennsylvania* (1990)

[Schabes 92] Yves Schabes: "Stochastic Lexicalized Tree-Adjoining Grammars" *Proceedings of the 14th Conference on Computational Linguistics (COLING)* (1992)

[Schabes and Waters 93] Yves Schabes, R.Waters: "Lexicalized Context-Free Grammars" *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics (ACL)* (1993)

[Sekine et al. 92a] Satoshi Sekine, Jeremy Carroll, Sofia Ananiadou, Jun-ichi Tsujii: "Automatic Learning for Semantic Collocation" *Proceedings of the 3rd Conference on Applied Natural Language Processing* (1992)

[Sekine et al. 92b] Satoshi Sekine, Sofia Ananiadou, Jeremy Carroll, Jun-ichi Tsujii: "Linguistic Knowledge Generator" *Proceedings of the 14th Conference on Computational Linguistics (COLING)* (1992)

[Sekine 92] Satoshi Sekine "Automatic Linguistic Knowledge Acquisition from Corpora" *MSc thesis; University of Manchester Institute of Science and Technology* (1992)

[Sekine 93] Satoshi Sekine: "Linguistic Knowledge Acquisition from Corpora using Dempster-Shafer Theory and Human Intervention" *Shizen-Gengo-Syori-Kenkyuukai (Research Group on Natural Language Processing) Information Processing Society of Japan (In Japanese)* (1993)

[Sekine 94a] Satoshi Sekine: "Automatic Sublanguage Identification for a New Text" *Proceedings of the 2nd Annual Workshop on Very Large Corpora* (1994)

[Sekine 94b] Satoshi Sekine: "A New Direction for Sublanguage NLP" *Proceedings of the International Conference on New Methods in Language Processing* (1994)

[Sekine 95] Satoshi Sekine: "A New Direction for Sublanguage NLP" *Journal of Gengo Syori Gakkai (Natural Language Processing)* (1995)

[Sekine et al. 95] Satoshi Sekine, John Sterling, Ralph Grishman: "NYU/BBN 1994 CSR Evaluation" *Proceedings of the Spoken Language Systems Technology Workshop* (1995)

[Sekine and Tsujii 95] Satoshi Sekine, Jun-ichi Tsujii: "Automatic Linguistic Knowledge Acquisition from Corpora" *Journal of Machine Translation* (1995)

[Sekine and Grishman 95] Satoshi Sekine, Ralph Grishman: "A Corpus-based Probabilistic Grammar with Only Two Non-terminals" *Proceedings of the 4th International Workshop on Parsing Technology* (1995)

[Sekine and Grishman 96] Satoshi Sekine, Ralph Grishman: "NYU Language Modeling Experiments for 1995 CSR Evaluation" *Proceedings of the Spoken Language Systems Technology Workshop* (1996)

[Sekine 96a] Satoshi Sekine: "Apple Pie Parser Homepage" *http://cs.nyu.edu/cs/projects/proteus/app*

[Sekine 96b] Satoshi Sekine: "Modeling Topic Coherence for Speech Recognition" *Proceedings of the 16th Conference on Computational Linguistics (COLING)* (1996)

[Sekine et al. 97a] Satoshi Sekine, Andrew Borthwick and Ralph Grishman: "NYU language Modeling Experiment for 1996 CSR Evaluation" *Proceedings of the DARPA Speech Recognition Workshop* (1997)

[Sekine and Grishman 97] Satoshi Sekine and Ralph Grishman: "Domain Project Report" *Final Report of Domain project* (1997)

[Sekine 97] Satoshi Sekine: "The Domain Dependence of Parsing" *Proceedings of the 5th Conference on Applied Natural Language Processing* (1997)

[Sekine and Collins 97] Satoshi Sekine and Micheal Collins: "Bracketing evaluation program" *http://cs.nyu.edu/cs/projects/proteus/evalb*

[Sekine et al. 97b] Satoshi Sekine, Kiyotaka Uchimoto and Hitoshi Isahara: "Corpus-based Japanese Parser Using Context Information" *Proceedings of the Natural Language Pacific Rim Symposium* (1997)

[Seymore 97] Kristie Seymore, Stanley Chen, Maxine Eskenazi and Roni Rosenfeld: "Language and Pronunciation Modeling in the CMU 1996 Hub 4 Evaluation" *Proceedings of the DARPA Speech Recognition Workshop* (1997)

[Shirai et al. 97] Kiyoaki Shirai, Kentaro Inui, Hozumi Tanaka and Takenobu Tokunaga: "An Empirical Study on Statistical Disambiguation of Japanese Dependency Structures Using a Lexically Sensitive Language Model" *Proceedings of the Natural Language Pacific Rim Symposium* (1997)

[Sima'an 97] Khalil Sima'an: "Explanation-Based Learning of Data Oriented Parsing" *Proceedings of the Workshop on Computational Natural Language Learning (CoNLL)* (1997)

[Simmons and Yu 91] Robert Simmons and Yeong-Ho Yu: "The Acquisition and Application of Context Sensitive Grammar for English" *Proceedings of the 29st Annual Meeting of the Association for Computational Linguistics (ACL)* (1991)

[Slocum 86] Jonathan Slocum: "How One Might Automatically Identify and Adapt to a Sublanguage: An Initial Exploration" *in Analyzing Language in Restricted Domains* (1986)

[Sparck-Jones 73] K.Sparck-Jones: "Index Term Weighting" *Information Storage and Retrieval, Vol.9, p619-633* (1973)

[Speech Workshop 97] *Proceedings of the Speech Recognition Workshop* DARPA (1997)

[Speech Workshop 96] *Proceedings of the Speech Recognition Workshop* DARPA (1996)

[Speech Workshop 95] *Proceedings of the Spoken Language Systems Technology Workshop* ARPA (1995)

[Tanaka 89] Hozumi Tanaka: *Shizen Gengo Kaiseki no Kiso - Introduction to Natural Language Analysis* - (in Japanese) Sangyo Tosyo (1989)

[Weng 97] Fuliang Weng, Andreas Stolcke and Ananth Sankar: "Hub-4 Language Modeling using Domain Interpolation and Data Clustering" *Proceedings of the DARPA Speech Recognition Workshop* (1997)

[Willett 88] Peter Willett: "Recent trends in hierarchic document clustering: a critical review" *Information Processing and Management, vol.24 num.5, pp577-597* (1988)

[Woodland et al. 97] P. Woodland, M. Gales, D. Pye and S. Young: "Broadcast News Transcription Using HTK" *Proceedings of the DARPA Speech Recognition Workshop* (1997)

[Young and Bloothooft 97] Steve Young and Gerrit Bloothooft (ed.): *Corpus-based Methods in Language and Speech Recognition* Kluwer Academic Publishers (1997)

[Zipf 49] G.Zipf: *Human behaviour and the principle of least effort* Addison-Wesley (1949)

[Zipf 65] G.Zipf: *The psycho-biology of language: An introduction to dynamic philology* MIT Press (1965)