S-clause Segmentation for Efficient Syntactic Analysis Using Decision Trees

Mi-Young Kim **Jong-Hyeok Lee**
Div. of Electrical and Computer Engineering
Pohang University of Science and Technology (POSTECH) and Advanced Information Technology Research Center(AlTrc)
San 31 Hyoja-dong, Nam-gu, Pohang 790-784, R. of Korea
colorful@postech.ac.kr jhlee@postech.ac.kr
fax: +82-54-279-5699 **fax: +82-54-279-5699**

Keywords : S-clause segmentation, Decision trees, Syntactic analysis, Long sentence analysis

Abstract

In dependency parsing of long sentences with fewer subjects than predicates, it is difficult to recognize which predicate governs which subject. To handle such syntactic ambiguity between subjects and predicates, this paper proposes an "S-clause" segmentation method, where an S(ubject)-clause is defined as a group of words containing several predicates and their common subject. We propose an automatic S-clause segmentation method using decision trees. The S-clause information was shown to be very effective in analyzing long sentences, with an improved performance of 5 percent.

1 Introduction

The longer the input sentences, the worse the parsing results are, since problems with syntactic ambiguity increase drastically. In our parser, subject errors form the second largest error portion, as 24.15% of syntactic parsing errors (see Table 1). Although the dependency errors in NP form the largest error portion, these errors are not significant since many applications (e.g. MT systems) using parsers deal with the NP structure as a one unit and do not analyze the syntactic relations within NP. So, this paper proposes a method to resolve subject dependency error problems. To improve the dependency parsing performance, we need to determine the correct dependency relations of subjects.

In most cases, a long sentence has fewer subjects than predicates. The reason is that several predicates can share one subject if they require the same word as their subject, or that the subject of a predicate is often omitted in a Korean sentence. So, in a long sentence, it is difficult to recognize the correct subject of some subjectless VPs. This paper proposes an S(ubject)-clause segmentation method to reduce ambiguity in determining the governor of a subject in dependency parsing. An S(ubject)-clause is defined as a group of words containing several predicates and their common subject. An S-clause includes one subject and several predicates which share the subject. The S-clause segmentation algorithm detects the boundary of predicates which share a common subjective word. We employ the C4.5 decision tree learning algorithm for this task.

The next section presents the background of previous work on sentence segmentation and clause detection. Next, dependency analysis procedure using S-clauses in Korean will be described. Afterwards, the features for decision tree learning to detect S-clauses will be explained, and some experimental results will show that the proposed S-clause segmentation method is effective in dependency parsing. Finally, a conclusion will be given.

dependency tree errors	subject-predicate dependency errors	predicate-predicate dependency errors	adjunct-predicate dependency errors	complement-predicate dependency errors	dependency errors within NP	dependency errors resulting from POS-tag errors
error %	24.15%	14.29%	17.35%	8.84%	27.55%	7.48%

Table 1. Dependency Tree Errors for 10,000 Test Sentences (avg 19.27 Words/sentence)

2 Previous Work

A considerable number of studies have been conducted on the syntactic analysis of long sentences. First, conjunctive structure identification methods have been proposed (Agarwal 1992;Jang 2002;Kurohashi 1994;Yoon 1997). These methods are based on structural parallelism and the lexical similarity of coordinate structures. While they perform well in detecting the boundary of a coordinate structure, they cannot determine the boundary of predicates that share a common subject. In addition, some papers insist that coordinate structure identification is impossible since Korean coordinate sentences do not maintain structural parallelism (Ko 1999).

Second, several studies have been made on clause segmentation (identification, splitting) (Sang and Dejean 2001). The clause seems to be a natural structure above the chunk (Ejerhed 1998). Clause identification splits sentences that center around a verb. The major problem with clause identification concerns the sharing of the same subject by different clauses (Vilson 1998). When a subject is omitted in a clause, Vilson(1998) attached the features of the previous subject to the conjunctions. However, the subject of a clause is not always the nearest subject. Therefore, a new method is required to detect the correct subject of a clause.

In addition, many studies have focused on segmentation in long sentences. Some try to segment a long sentence using patterns and rules and to analyze each segment independently (Doi 1993; Kim 1995; Kim 2002 ;Li 1990). Similarly, an intrasentence segmentation method using machine learning is proposed (Kim 2001). Although this method reduces the complexity of syntactic analysis by segmenting a long sentence, the ambiguity problem with the dependency of subjects remains unsolved. Further, Lyon and Dickerson take advantage of the fact that declarative sentences can almost always be segmented into three concatenated sections (pre-subject, subject, predicate) which can reduce the complexity of parsing English sentences (Lyon and Dickerson 1995; Lyon and Dickerson 1997). This approach is useful for a simple sentence that contains a subject and a predicate. A long sentence generally contains more than a subject and a predicate. Therefore, the segmentation methods proposed by Lyon and Dickerson are inefficient for parsing long sentences. In studies on segmenting long sentences, little attention has been paid to detecting the boundaries of predicates which share a common subject.

To determine the correct subject of some subjectless VPs, we define the 'S-clause' and propose an S-clause segmentation method. In previous work, a clause is defined as a group of words containing a verb, and previous researchers split sentences centering around a verb to detect clauses. By contrast, we split sentences centering around a subject. So we call the proposed segment 'S(ubject)-clause' to distinguish it from a clause.

3 Dependency Analysis in Korean Language

3.1 Dependency Analysis Procedure

This section overviews our dependency analysis procedure for the Korean language.

1. Chunking
2. Detect clauses
   (Using clauses, determine the heads of arguments(except subjects)

3. Detect S-clauses
   (Using S-clauses, determine the heads of subjects and the heads of non-arguments(adjuncts))

First, we determine NP- and VP-chunks following the method of Kim (Kim et al, 2000). Next, we bind a predicate and its arguments to determine the heads of arguments using subcategorization and the selectional restriction information of predicates. This procedure is similar to the clause detection procedure. In this procedure, we also determine the grammatical function of unknown case words according to Lee's method (Lee et al, 2003).

It is important to identify the subject grammatical function of unknown case words correctly, since one S-clause is constructed per subject.

In Korean, arguments of predicates, especially subjects, are often omitted in a sentence. We leave the dependency relations of subjects unconnected, since ambiguity occurs when detecting the heads of subjects.

Third, using predicate information and grammatical function detection results after clause detection, we detect S-clauses. And then, using S-clauses, we determine the dependency relations between subjects and predicates. It can be also helpful in determining the heads of adjuncts, since their heads can be found within an S-clause boundary.

3.2 Dependency Analysis based on S-clauses

Before S-clause segmentation, we have determined the dependency relations between arguments (except subjects) and predicates. Next, we determine the heads of subjects after S-clause segmentation. Although we assume that all the predicates in an S-clause require the subject within the S-clause, some S-clause segmentation errors may exist. To recover the S-clause segmentation errors, we use selectional restriction information to find the relevant head of a subject. We regard the head of the subject within an S-clause as the farthest predicate in the S-clause which requires the concept of the subject.

Still, the dependency relations of adjuncts and those of predicates are not determined. The heads of adjuncts and those of predicates are dependent on the nearest head candidate not giving rise to crossing links. Using S-clauses, we can accomplish dependency parsing simply and effectively.

4 S-clause Segmentation based on Decision Tree Learning

4.1 The C4.5 Learning Algorithm

Decision tree induction algorithms have been successfully applied to NLP problems such as parsing (Magerman 1995;Haruno et al 1998), discourse analysis (Nomoto and Matsumoto 1998), sentence boundary disambiguation (Palmer 1997), phrase break prediction (Kim 2000) and word segmentation (Sornertlamvanich et al 2000). We employed a C4.5 (Quinlan 1993) decision tree induction program as the learning algorithm for S-clause segmentation.

The induction algorithm proceeds by evaluating the information content of a series of attributes and iteratively building a tree from the attribute values, with the leaves of the decision tree representing the values of the goal attributes. At each step of the learning procedure, the evolving tree branches from the attribute that divides the data items with the highest gain in information.

Branches will be added to the tree until the decision tree can classify all items in the training set. To reduce the effects of overfitting, C4.5 prunes the entire decision tree after construction. It recursively examines each subtree to determine whether replacing it with a leaf or branch will reduce the expected error rate. Pruning improves the ability of the decision tree to handle data which is different from the training data.

4.2 Features

This section explains the concrete feature setting we used for learning. The S-clause is a broader concept than the clause. In order to determine the S-clauses, we must choose the clauses that are suitable for addition to the S-clause. Since the head word of a clause is the predicate in the clause, we merely use predicate information. The feature set focuses on the predicates.

An S-clause can be embedded in another S-clause. Therefore, we should learn two methods to detect the left boundary and right boundary of an S-clause independently.

We should include one subject between the left boundary and the right boundary of an S-clause. We call the subject to include in an S-clause the 'target subject'.

First, when we detect the left boundary of an S-clause, we consider the predicates between the

1st Feature	Type of a predicate
2nd Feature	Surface form of the last ending of a predicate
3rd Feature	Comma

Table 2: Linguistic Feature Types Used for Learning

Feature Type	**Values**
1st	adnominal, conjunctive, quotative, nominal, final, null
2nd	ㄴ, ㅁ, 기, 음, ㄴ데, ㄴ즉, ㄴ지, ㄹ지, ㄹ지니, 거나, 게, 고, 나, 는데, 는지, 니, 다가, 도록, 든지, 듯이, 라, 려고, 며, 면, 면서, 므로, 아, 아서, 어, 어도, 어서, 어야, 으나, 으니, 으려고, 으며, 으면, 으면서, 으므로, 자, 지, 지마는, null….
3rd	1, 0, null

Table 3: Values for Each Feature Type

'target subject' and the nearest subject which precedes the 'target subject'.

Each predicate has 3 features, as shown in Table 2. The 1st feature concerns the type of a predicate. Next, the 2nd feature takes the value of the surface form of the last ending of a predicate. Korean is an agglutinative language and the ending of a predicate indicates the connective function with the next VP (e.g. '으므로(because)' indicates it functions as a reason for the next VP).

The 3rd feature deals with the information whether a predicate is followed by a comma or not. The use of a comma to insert a pause in a sentence is an important key to detect an S-clause boundary.

We use 12 features for left boundary detection — 4 predicates, and 3 features for each predicate as summarized in Table 2. The class set consists of 5 values (0~4) to indicate the position of the predicate that becomes a left boundary. If the class value is 0, it means the S-clause includes no predicates preceding the 'target subject'. Other wise, if the class value is 1, it means that that the S-clause includes one nearest predicate which appears preceding the 'target subject'.

The window size of predicates for the left boundary is 4. If there are less than 4 predicates, then we fill the empty features with 'null'.

For right boundary detection, we consider the predicates between the 'target subject' and the next subject following the 'target subject'.

We use 15 features for right boundary detection — 5 predicates, and the same 3 features for each predicate as in Table 2. Among the predicates between 'target subject' and the next subject following the 'target subject', we consider 4 predicates which appear near the 'target subject' and 1 predicate which locates last. The reason that 1 predicate which locates last is considered is as follows: If all the predicates between 'target subject' and the next subject following the 'target subject' require the 'target subject' as their common subject, the right boundary becomes the last predicate among them, since Korean is a head-final language.

Although the feature set is the same as that for right boundary detection, the window size for the right boundary is 5, which is larger than that for the left boundary. The reason is that Korean is a head-final language and the predicates of a subject appear after the subject.

The detailed values of each feature type are summarized in Table 3.

We first detect the S-clause which includes the last subject of an input word set. If an S-clause is detected, we exclude the words which are included in the S-clause from the input word set. Then, we recursively detect the S-clause including the last subject in the remaining word set until there are no subjects in the modified word set.

5 Experimental Results

We evaluated the proposed S-clause segmentation method using the Matec99'[1] test set. We evaluated the following 2 properties of the S-clause segmentation program.

1. The amount of training data vs. S-clause segmentation accuracy vs. parsing accuracy

[1] Morphological Analyzer and Tagger Evaluation Contest in 1999

Number of training sentences	5000	10000	20000	30000	40000	50000
S-clause Precision	82.14%	83.68%	83.70%	84.10%	84.06%	84.40%
S-clause Recall	81.98%	83.54%	83.61%	84.02%	83.98%	84.30%
Parsing Accuracy	86.28%	87.53%	87.86%	88.79%	89.09%	89.12%

Table 4: The Amount of Training Sentences vs. S-clause Accuracy vs. Parsing Accuracy for the 10000 test sentences

	Our parser without S-clause segmentation procedure	Our parser with S-clause segmentation procedure	KN Parser	Korean Yon-sei parser
Accuracy in detecting the head of a subject	51.60 %	84.03 %	74.21 %	Unknown
Parsing Accuracy	84.29%	89.12%	89.93 %	87.30%

Table 5: Parsing Accuracy Comparison (avg 19.27word/sentence)

2. Significance of features

5.1 The Amount of Training Data vs. S-clause Segmentation Accuracy vs. Parsing Accuracy

The test set is different from the training set and the average length of the test sentence is 19.27 words/sentence while that of the training sentence is 14.63 words/sentence. We selected longer sentences as a test set since the S-clause segmentation method is proposed to improve the performance of syntactic analysis in long sentences.

The parsing accuracy is calculated as (correct dependency links)/(all the dependency links). The number of detected dependency links and that of the true dependency links are equal, so parsing accuracy is the same as parsing recall. For the reason, we do not measure the parsing recall separately. However, in the case of S-clauses, the S-clause precision is different from the S-clause recall, since the subject grammatical function detection results for unknown case words are not perfectly correct. We measured the S-clause precision as (correct S-clauses)/(all the detected S-clauses), and the S-clause recall as (correct S-clauses)/(all the true S-clauses).

To show the effectiveness of S-clauses, we compare the parsing result using S-clauses and without S-clauses, and also compare our parser performance with others which analyze similar languages with Korean.

In the experiments, we obtained the following two results.

1. The better the S-clause segmentation performance, the better the parsing accuracy that results.
2. The maximum S-clause accuracy is 84.40% and the maximum parsing accuracy is 89.12% with 50000 training sentences. The test set size is 10,000 sentences.

We will discuss the maximum accuracy of 89.12% compared with the Japanese KN parser, which shows the highest performance in Japanese dependency parsing.

The characteristics of Japanese are similar to the Korean language. So, the mechanism of syntactic analysis in Korean can also be applied to Japanese. In Japanese dependency parsers and Korean dependency parsers, the KN parser shows the highest performance. In addition, we can freely obtain the programs. So, we compare the performance of our parser with that of the KN parser. To do this, we need a bilingual test corpus. We obtain 10,000 Japanese test set by translating the 10,000 Korean test set using Korean-to-Japanese machine translation system of our own. Then, several researchers specializing in Japanese manually corrected the translation results. We experimented on the performance of the KN parser using these 10,000 Japanese sets.

To detect the head of a subject, the KN parser uses only some heuristics (Kurohashi 1994). As shown in Table 5, the performance of our parser

Feature	**Accuracy Change**	**Feature**	**Accuracy Change**
1st type	**-7.34%**	3rd type	-0.04%
1st surface form	-1.15%	3rd surface form	-0.02%
1st comma	-2.42%	3rd comma	-0.82%
2nd type	-0.32%	4th type	-0.0%
2nd surface form	-0.23%	4th surface form	-0.0%
2nd comma	-5.29%	4th comma	-0.01%

Table 6: S-clause Accuracy Change When Each Attribute for Left Boundary Removed

Feature	**Accuracy Change**	**Feature**	**Accuracy Change**
1st type	-3 %	**3rd comma**	**-3 %**
1st Surface form.	-0.8 %	4th type	-0.2 %
1st comma	**-2.7 %**	4th surface form	-0.3 %
2nd type	-0.3 %	4th comma	0.0 %
2nd surface form	-1.3 %	5th type	-0.8 %
2nd comma	**-1.9 %**	5th Surface form.	-0.1 %
3rd type	0.0 %	5th comma	0.0 %
3rd surface form	0.0 %		

Table 7: S-clause Accuracy Change When Each Attribute for Right Boundary Removed

S-clause errors	Subject detection errors	Pos-tag errors	Double subject errors	Left boundary errors	Right boundary errors	Predicate role of adverbials	Other-wise
Error %	25.15%	20.66%	11.38%	16.47%	20.96%	2.00%	3.38%

Table 8: The Type of S-clause Errors

without S-clause segmentation is worse than that of the KN parser. In our parser without S-clause segmentation, a word simply depends on the nearest head not giving rise to crossing links. However, after S-clause segmentation, the performance of our parser is similar to that of the KN parser. The accuracy of our parser in detecting the head of a subject is also better than that of the KN parser.

We also compare the performance of our parser with a Korean Yon-sei dependency parser, as shown in Table 5. The parser using S-clauses outperforms the Yon-sei parser by 1 percent. Since the Yon-sei dependency parser is not an open resource, we simply compare the performance of our parser with that of Yon-sei parser written in Kim (2002). Therefore, the comparison of the performance between our parser and the Korean Yon-sei dependency parser may not be so reasonable.

5. 2 Significance of Features

Next, we will summarize the significance of each feature introduced in Section 4.2. Table 6 and Table 7 illustrate how the S-clause accuracy is reduced when each feature is removed. Table 6 clearly demonstrates that the most significant feature for the left boundary is the type of the previous 1st predicate— we obtain the information from the decision rules that, especially, the 'adnominal' type of the previous 1st predicate is a significant feature. As shown Table 6, 4th predicate information has no effect on the left boundary.

Table 7 demonstrates that the most significant feature for the right boundary is comma information, since the S-clause accuracy without 1st, 2nd or 3rd comma information shows high accuracy decrease. The 5th predicate information is more useful than the 4th predicate. In other words, the last predicate can be the head of a subject than the intermediate predicate.

This result may partially support heuristics; the left boundary would be an adnominal predicate since only adnominal predicates are *followed* by their subjects (other predicates are *preceded* by their subjects). Next, after the comma, a boundary mostly occurs. In particular, we need to concentrate on the types of predicates to attain a higher level of accuracy. To some extent, most features contribute to the parsing performance.

In our experiment, only the surface form of the endings of conjunctive predicates, rather than other predicates, is effective on performance. The reason is that the surface form of the ending of the non-conjunctive predicates does not indicate the connective function with the next VPs.

5.3 Discussion about S-clause Errors

We classify the S-clause errors, as shown in Table 8. Table 8 shows that many S-clause errors are due to the Korean characteristics.

Among the S-clause errors, subject detection errors rank first, which occupy 25.15%. So, the S-clause accuracy result is different from the S-clause recall result.

Next, POS tagging errors result in the S-clause segmentation errors of 20.66 percent.

These two errors occur before S-clause segmentation. So, this is another issue that remains for future work.

Also, double subject errors are 11.38%. Some Korean predicates can require two subjects. This is contrary to our assumption of S-clauses. Since 11.38% is large portion of all the errors, we should consider double subject construction and identify the characteristics of the predicates in double subject constructions.

The right boundary errors are more than left boundary errors. It means that the right boundary detection is more difficult.

Finally, some adverbials, not predicates, can function as predicates of subjects. Since we only detect boundaries focusing on predicates, these adverbials information cannot be used. We should include these adverbials that function as predicates into the S-clause boundary candidates.

6 Conclusion

This paper proposes an S-clause segmentation method to reduce syntactic ambiguity in long sentences. An S(ubject)-clause is defined as a group of words containing several predicates and their common subject. An S-clause includes one subject and several predicates that share the subject.

We have described an S-clause segmentation method that uses decision trees. The experimental results show that the parser using S-clauses outperforms the parser without S-clauses by 5% and also outperforms conventional Korean dependency parsers by 1 percent. To improve the S-clause accuracy, we should detect double subject constructions and adverbials which function as predicates. We plan to continue our research in two directions. First, we will combine our S-clause segmentation method with a coordinate structure detection method and test the parsing results. Second, we will apply it to a machine translation system and translate each S-clause independently and test the translation performance.

Acknowledgements

This work was supported by the Korea Science and Engineering Foundation (KOSEF) through the Advanced Information Technology Research Center(AITrc) and by the Brain Korea 21 Project in 2003.

References

Rajeev Agarwal, Lois Boggess. A simple but useful approach to conjunct identification. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, p.15-21, 1992.

Orasan Constantin. *A hybrid method for clause splitting in unrestricted English texts*. In Proceedings of ACIDCA 2000

Shinchi Doi, Kazunori Muraki, Shinichiro Kamei and Kiyoshi Yamabana. Long sentence analysis by domain-specific pattern grammar, In P*roceedings of the 6th Conference on the European Chapter of the Association of Computational Linguistics*, p.466, 1993.

Eva Ejerhed. Finding clauses in unrestricted text by finitary and stochastic methods. In *Proceedings of the 2nd Conference on Applied Natural Language Processing*, p.219-227, 1998.

Masahiko Haruno, Satoshi Shirai, and Yoshifumi Ooyama. Using Decision Trees to Construct a Practical Parser, In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, p.505-511, 1998.

Jaechol Jang, Euikyu Park, and Dongryeol Ra. A Korean conjunctive structure analysis based on sentence segmentation *(In Korean)*. In *Proceedings of 14th Hangul and Korean Information Processing,* p.139-146, 2002.

Byeongchang Kim and Geunbae Lee. Decision-Tree based Error Correction for Statistical Phrase Break Prediction in Korean. In *Proceedings of the 18th International conference on Computational Linguistics*, p.1051-1055, 2000

Kwangbaek Kim, Euikyu Park, Dongryeol Ra and Juntae Yoon. A method of Korean parsing based on sentence segmentation. In *Proceedings of 14th Hangul and Korean Information Processing,* p.163-168, 2002

Miyoung Kim, Sin-Jae Kang and Jong-Hyeok Lee. *Text Chunking by Rule and Lexical Information.* In proceedings of the 12th Hangul and Korean Information Processing Conference, Chonju, Korea. pp 103~109. 2000

Sungdong Kim and Yungtaek Kim. Sentence analysis using pattern matching in English-Korean machine translation, In *Proceedings of the International Conference on Computer Processing of Oriental Languages*, p.199-206, 1995.

Sungdong Kim, Byungtak Zhang and Yungtaek Kim. Learning-based intrasentence segmentation for efficient translation of long sentences. *Machine Translation* 16:151-174, 2001.

Kwangju Ko. Review of coordinate sentences *(in Korean)*, *Journal of Korean*, 9:39~80, 1999.

Sadao Kurohashi and Makoto Nagao, A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures, *Computational Linguistics*, 20(4):507-534,1994

Vilson. J. Leffa. Clause processing in complex sentences. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*, 1998.

Wei-Chuan Li, Tzusheng Pei, Bing-Huang Lee and Chuei-Feng Chiou. Parsing long English sentences with pattern rules. In *Proceedings of the 13th International Conference on Computational Linguistics,* p.410-412, 1990.

YongHun Lee, Mi-Young Kim and Jong-Hyeok Lee. *Grammatical Role Determination of Unknown Cases in Korean Coordinate Structures.* In Proceedings of the 30th Korea Information Science Society Spring Conference, p.543-545, 2003.

Caroline Lyon, and Bob Dickerson. A fast partial parse of natural language sentences using a connectionist method, In *Proceedings of the 7th conference on the European Chapter of the Association of Computational Linguistics*, p.215-222, 1995

Caroline Lyon, and Bob Dickerson. Reducing the complexity of parsing by a method of decomposition. In *Proceedings of the 6th International Workshop on Parsing Technology*, 1997.

David M. Magerman. Statistical Decision-Tree Models for Parsing, In *Proceedings of the 33rd Annual Meeting of Association for Computational Linguistics*, p.276-283. 1995

Tadashi Nomoto and Yuji Matsumoto. Discourse Parsing: A Decision Tree Approach. In proceedings of the 6th Workshop on Very Large Corpora, p.216-224, 1998

David D. Palmer, Marti A. Hearst. Adaptive Multilingual Sentence Boundary Disambiguation, *Computational Linguistics*, 27:241-261, 1997

J. Ross Quinlan. C4.5 Programs for Machine Learning. *Morgan Kaufmann Publishers.* 1993

Erik F. Tjong Kim Sang and Herve Dejean. Introduction to the CoNLL-2001 Shared Task: Clause Identification. In *proceedings of CoNLL-2001*, p. 53-57, 2001.

Virach Sornertlamvanich, Tanapong Potipiti and Thatsanee Charoenporn. Automatic Corpus-Based Thai Word Extraction with the C4.5 Learning Algorithm, In *Proceedings of the 18th International Conference on Computational Linguistics,* p.802-807, 2000

Juntae Yoon, M. Song. Analysis of coordinate conjunctive phrase in Korean *(in Korean), Journal of Korea Information Science Society,* 24:326-336, 1997