On Representation of Preference Scores

Kiyoshi YAMABANA Shin'ichiro KAMEI
Kazunori MURAKI

C&C Information Technology Research Laboratories
NEC Corporation
4-1-1 Miyazaki, Miyamae-ku, Kawasaki 216, JAPAN
E-mail: {yamabana,kamei,k-muraki}@mtl.cl.nec.co.jp

Abstract

We focus on ambiguity resolution in breadth-first parsing, and propose a new method for preference score representation, in which preference scores for a parse tree are represented and propagated as a vector.

We first show problems of conventional method, that appeared because the quantity assigned and propagated as a score was a single numerical value. In our method, preference scores are represented and propagated as a vector. Each vector element corresponds to a single linguistic phenomenon, and is calculated independently of other vector elements. The vector scores in child nodes and the rule are added as vectors to produce the vector score of the parent tree. The numerical preference score is calculated from this vector score by a separate totalizing function.

This method allows rule writers to represent disambiguation knowledge more naturally than previous methods. It also allows us to easily introduce new scoring factors and change existing scoring knowledge. We realized this method in an English analyzer, and confirmed these advantages.

1 Introduction

Disambiguation is a central problem in natural language processing. Many MT translation errors are caused by problems in disambiguation, and improving the disambiguation ability of the system is essential for improving translation quality.

Knowledge for resolving ambiguity in MT comes from many sources: it exists as lexical, structural, semantic or contextual knowledge. But methods to combine this knowledge and obtain a reasonable preference are just now being investigated. In this paper we propose a new model for preference score representation. This model allows us to naturally represent and utilize diverse kinds of disambiguation knowledge, and also enables us to easily add new knowledge to the system.

A conventional method for preference score representation assigned to each partial solution a numerical value that represented preference score of that solution. When a new solution is constructed from partial solutions by applying structure rules, the scores of child solutions and the score of the rule being applied are summed up to produce the score of the new solution. We show this simple method is not sufficient to treat some linguistic phenomena, such as relative clause construction. The problem is that the preference score is represented and propagated as a single numerical value. We also show that this method has difficulties in maintaining consistency in evaluation knowledge and in introducing new evaluation factors because the knowledge for evaluation of individual factors and the knowledge for combining different factors are not clearly separated.

To cope with these problems our method represents and propagates preference scores as a vector. Each vector element corresponds to a single linguistic phenomenon, that is calculated independently of other vector elements.

We realized an English analyzer using this method, that is a part of an English-Japanese machine translation system. In this implementation, a preference score was represented as a six-dimensional

vector. We were able to concisely write evaluations for various linguistic phenomena. We also confirmed that we could easily introduce new evaluation knowledge, and incorporate disambiguation knowledge from databases outside of the analyzer.

In section 2 we examine problems with the conventional method that motivated ours. In section 5 we propose a vector model of preference score representation and show how this method solves previous problems. In section 4 we show a realization of our method in an English analyzer. In section 5 we briefly compare our approach to other approaches. The last section is a conclusion.

2 Problems of Conventional Method

In a conventional method, a root node of a syntax tree is assigned a single numerical value, that represents the preference score of that tree. When a grammar rule is applied to construct a new tree, scores of children and the score of the grammar rule being applied are summed up and assigned a preference score to the root node of the new syntax tree (Figure 1). Although this method is quite general and reported to work well[l, 2], we think that it has the following problems:

1. A preference score has a double meaning.
   On one hand, a preference score of an interpretation (or a structure) represents the plausibility that interpretation is preferred as the correct one by human beings. The system chooses an interpretation with highest preference score among possible ones. On the other hand, a preference score has a meaning as a part of the parent structure score, since the preference score of a parent structure is usually a summation of scores of child structures and a score of a structure rule used in the construction. But, these two meanings sometimes contradict. In the next subsection we explain this problem in detail.

2. Knowledge for scoring individual factors and knowledge for combining these individual scores are not clearly separated.
   Since scores evaluated for individual factors are directly added to the total preference score, there is no clear distinct place to hold the knowledge for combining these scores; it is implicit in individual scoring procedures. This causes practical problems. First, introducing a new evaluation factor becomes difficult, because rule writers must consider the amount of contribution from all other factors and try to keep consistency **before** they write the scoring procedure for the new factor. Second, maintaining the overall consistency in scoring knowledge becomes an extremely difficult task, because there is no easy way to separate and clarify contributions of individual factors to the total preference score.

2.1 Double Meaning of Score

We explain the first problem in the following. A typical contradiction occurs when a structure that is unacceptable as an independent structure appears as a part of a highly acceptable structure. This is a typical situation where we need "re-evaluation" of scores. Re-evaluation in this paper means by definition to evaluate again some factor that was already calculated and assigned in a child structure.

2.2 The Case of Gapped Structure

Consider a sentence with a relative clause:

I know the man she was talking to.

Since a gapped sentence in relative clauses behave the same way as an ordinary ungapped sentence except for gap existence (e.g. consider modification by an adverb), they should be treated by the same grammar rules[1] so that we can reduce the cost of writing and maintaining grammar rules. With this assumption, the gapped substructure

she was talking to

[1] This treatment agrees with many contemporary grammar theories.

has the same preference score as the score that is assigned when this structure appears as an independent sentence (not as a substructure of a relative clause). Since the preference score of a gapped interpretation should be lower than any ungapped interpretations, the preference score of this substructure should be very low. But, after combining with an antecedent, the gap disappears and the whole noun phrase

the man she was talking to

is acceptable as an independent noun phrase, and should receive at least an average score. There is a large jump in the preference score between the child structure and the parent structure.

In the conventional method, the only way to write this jump is to assign a high preference score to the grammar rule used. But, there is no reason that this rule is highly preferred to other grammar rules.

This treatment is not only ad-hoc, but also poses difficulty in management of the scoring rules, since the same scoring knowledge must be contained in several rules: the gap-filling rule must know precisely what score was assigned by gap-creating rules, in order to compensate the low score assigned by the latter. It is important to distinguish between a high score from high evaluation, and high score from re-evaluation. Without this distinction rule writers trying to incorporate new knowledge into the system may have difficulty in maintaining consistency.

2.3 Determining Part-of-Speech of " DE"

Another example of re-evaluation phenomena is part-of-speech determination of Japanese postposition/copula " DE", that has a slightly different reason for re-evaluation. " DE" has two possible pan of speech; one is postposition, another is an inflected form of a copula " DA".

Interpretation as a copula is always possible, while interpretation as a postposition is limited by the semantics of the head noun. In many cases, postpositional phrase *"NOUN* DE"(where *NOUN* is any noun) behaves as an optional case frame element of a verb, and in this case possible interpretation of " DE" is mainly determined by the semantics of head noun. For example, " DE" in "GAKKOU(school) DE" usually means general location since "GAKKOU" is an organization or a location, while " DE" in "NAIHU (knife) DE" means instrument. But, if the head noun is a person's name, for example "TANAKA-san" (Mr. TANAKA), interpretation as a postposition becomes less preferable (interpretation as a copula becomes more preferable), since appropriate interpretation cannot, be associated in this case.

But, when the postpositional phrase is attached to a verb that requires " DE" postpositional phrase as an obligatory case frame element, the preference only by the semantics of the head noun must be abandoned. Instead, the preference have to be re-evaluated using semantic restriction of the corresponding case frame of the verb.

For example, when " DE" is attached to a verb "OWARU" (end), *"NOUN* DE OWARU" means "end with *NOUN".* The semantic restriction on *NOUN is* weak in this case and postpositional interpretation of " DE", even if the head noun is a person's name, is highly acceptable. "TANAKA-san DE OWARU" is naturally interpreted as "end with Mr. TANAKA". It should obtain at least an average preference score through re-evaluation of the semantic restriction.

To treat this case in existing methods, a grammar rule that combines " DE" prepositional phrase with a verb has to know what score was assigned by previous rules. This causes the same problem mentioned in the former subsection.

We remark on a possible alternative. In the discussion above, re-evaluation became necessary because semantic restriction was different depending if the postpositional phrase was considered obligatory or optional. Then, even in the conventional method, by treating them as different lexical items, there is no need to re-evaluate preference scores. But, this approach increases the number of possible syntax trees greatly, and not desirable.

Generally speaking, re-evaluation of preference scores becomes necessary when a linguistic structure is constructed through more than one rule. It is not an extra phenomenon that occurs only in some special linguistic phenomena.

3 Vector Representation of Preference Scores

Above phenomena occurred because a preference score for final selection and a preference information that propagates over the tree are identified. We propose a new framework of preference score representation, in which the score for final selection and the score for holding/propagating preference information are distinguished. The latter score is represented and propagated as a vector.

3.1 The Method

First we give a description of the proposed method.

1. To each node of a syntax tree, a vector, that is called a vector score, is assigned. It represents general preference information of the syntax tree underlying that node.
2. When a new syntax tree is constructed, the vector score of the new tree is calculated as a sum of the vector scores of child nodes and a vector score given by the grammar rule being applied.
3. To each node of a syntax tree is assigned a scalar, that is called a total score. It represents the preference for final selection of the syntax tree underlying that node.
4. The total score of a node is calculated when that node is constructed, by a totalizing function that is given in the grammar rule being applied, from the vector score of that node.

Figure 2 shows the way scores are calculated in our method. The newly constructed syntax tree has three children. Each children has a vector score $\vec{S}_i (1 \leq i \leq 3)$. The grammar rule has a vector score $\vec{G}$. Then the vector score $\vec{S}$ of the root node is calculated by $\vec{S} = \vec{S}_1 + \vec{S}_2 + \vec{S}_3 + \vec{G}$. The total score T_{score} of the root node is calculated by $T_{score} = F_{total}(\vec{S})$ where F_{total} is a totalizing function, and is not affected directly by the total scores of child nodes.

3.2 Treatment of "Double Meaning"

In this method, the numerical score for final selection is distinct from the vector score that is propagated as preference information. Thus, there is no problem of "doubleness" of meaning. We have only to replace corresponding vector element by the newly evaluated value.

This is shown in Figure 3(a). Factor "existence of gaps" is an element of the vector score and contributes the total score through a large weight (in this example 10^5). At point A, a gapped structure is introduced and -1 is given to this element. The total score is calculated from this vector score, and has a large negative value because of the large weight associated to this element.

During parsing, this element propagates untouched to parent nodes and continues to contribute large negative value to the total score. At point B, the gapped structure is combined with an antecedant, and the gap disappears. The vector element "existence of gaps" is cleared to neutral (in this example 0) by the grammar rule. After this, the negative contribution from this element disappears.

The following is an example of related grammar rules. The rule that creates a gapped verb phrase is:

```
verb_phrase{slash Np; score Score}
 -> verb-{subcat Np; score Score}, <add(Score, gap_existence, -1)>;
```

This rule means that a gapped verb phrase is constructed from a transitive verb without an object and the element "gap_existence" of preference score is added -1.

The rule that combines a gapped sentence with an antecedent is:

```
noun_phrase {score Score}
 -> noun_phrase Np, sentence{slash Np}, <clear(Score, gap_existence)>;
```

We have only to specify that the evaluation "gap_existence" is cleared, then it is reflected in total score automatically.

In Figure 3(b), we show a treatment by a conventional method for comparison. The negative contribution (-10^5) by an existence of a gap is directly added to the total score by the grammar

rule at point A. When the gapped structure and the antecedent is combined at point B, this negative contribution is removed by the grammar rule by adding exactly the same value which was subtracted earlier. The grammar rule has to know what value was added to the total score in the tree below. If the value is changed by rule writers, both must be changed simultaneously.
The case of Japanese postposition "DE" can be treated in a similar way.

3.3 Separation of Evaluation Knowledge

Another advantage of our method is that knowledge for evaluating individual factors are stored separately from knowledge for gathering individual evaluations to obtain the final selection score. The former is used only in calculation of vector scores, and the latter only for calculation of total scores.

3.3.1 Introduction of New Evaluation Factors

In our method, new evaluation factors can be introduced more easily than previous methods.

Suppose we introduce a factor "part of speech of sentence head", so that verb-headed sentential interpretations are preferred than noun-headed interpretations. In older methods, evaluation of individual factor was directly reflected by the total preference score, so rule writers had to consider two things at a time: one is that verb-headed sentences should be preferred than noun-headed sentences, and another is the interference with other evaluation factors. As a consequence, introducing new evaluation factor was a difficult task.

In our method, evaluation for each factor is stored separately in an vector element. The step to evaluate a factor to calculate a vector element and the step to calculate total score from the vector score is completely separated. This separation makes it possible to introduce a new evaluation factor in two steps.

In the first step, we introduce a new element to the vector score and write a scoring procedure for the new factor. The result is stored in the newly introduced vector element. For example, we give 1 point to verb-headed structures, -1 point to noun-headed structures. In this step, we don't need to consider the interference with other evaluation factors. There is no change in the totalizing function, and the total score is not changed.

In the second step, we modify the totalizing function so that it uses that newly incorporated vector element for calculation of the total score. We examine the resulting ambiguities and associated vector scores, then determine how this new vector element contributes to the total score. In this step we need to consider interference effects with other factors.

3.3.2 Developing Grammars by Many Writers

When grammar rules are developed by more than one person, it often becomes difficult to keep consistency in evaluation knowledge. Our method has an advantage in this respect.

In the conventional method, rule writers had to consider in advance the interference between evaluation factors that are being written by other writers. In our method, evaluating knowledge is separated. As explained above, writing rule can be divided into two steps: first we write individual evaluation procedures, then we write totalizing function. The latter is the only step that requires overall consistency, and the first step can be performed without considering other evaluation factors, that is, without consulting other rule writers.

4 Realization in an English Analyzer

We realized an English analyzer using this method, that is the analyzer part of an experimental English to Japanese machine translation system. The number of grammar rules is approximately 300, and the number of lexical entries in the English dictionary is about 500. The main body of the analyzer is a unification-based chart parser, and the parsing strategy is bottom-up breadth-first. There is no pruning during parsing, and after all possible solutions are obtained, the solution with the highest preference score is selected. The grammar accepts standard English constructs, including questions, relatives and coordinations. In addition, it uses simple contextual information to determine the translation of some words.

4.1 Elements of Vector Score

The system mainly uses syntactic information for disambiguation. The weights reflect a disambiguation strategy, in that unpreferred interpretations are filtered out first, then preferred ones are selected (Filtering-Selecting Strategy, FSS hereafter). For the same kind of evaluation knowledge, a negative factor is given a heavier weight than a positive one. This is because if there is one unpreferred substructure, the whole structure should have low evaluation even if there are many other preferred substructures.

The vector score is represented as a six dimensional vector. Their meanings are as follows:

1. Strong Inhibitory Syntactic Factors (Grammaticality) *(v_1)*
2. Strong Preferable Syntactic Factors (Collocation) *(v_2)*
3. Contextual Factors (v_3)
4. Mixed Syntactic/Semantic Factors (v_4)
5. Weak Inhibitory Syntactic Factors *(v_5)*
6. Weak Preferable Syntactic Factors *(v_6)*

The first element marks strongly unpreferable syntactic constructs. In this realization it consists of "Existence of Gap" factor. It is negative when the structure contains a gap, and neutral if not. The value is assigned by gap creating grammar rules, and cleared by gap filling rules. This factor is separated because it is most dominant contributor to the total score when it contains non-zero value. Another reason is that it has to be re-evaluated (cleared) in gap filling rules.

The second element marks strongly preferred syntactic constructs. It is given a positive value if the structure contains a highly collocated combination of words.

The third element "Contextual Factors" contains a positive value if the structure is contextually preferred. Since it is difficult to reject an interpretation only by context information, negative values are rarely assigned. We did not provide a distinct negative element. This is mainly used for word sense disambiguation, and interpretation of questions if it is a simple inquiry or asking something.

The fourth element "Mixed Syntactic/Semantic Factors" contains a value that specifies plausibility of constructions such as adjunct attachment or prepositional phrase attachment that requires evaluating mixed syntactic/semantic information.

The fifth element represents weak inhibitory preferences such as dislike for noun-head interpretation of a sentence.

The sixth element contains other syntactic preferences.

4.2 Totalizing Function

The totalizing function is a simple weighted sum:

$$total\_score = \sum_{i=1}^{6} v_i \cdot 10^{6-i}$$

The weight reflects the importance of each factor, with FSS considered. If the final structure has a gap, it is highly unlikely to be a correct interpretation, so this factor has the highest weight. Other weights are ordered by similar considerations.

4.3 An Example

We show an example scoring. The input sentence is:

There is a Shinkansen that leaves about 6 o'clock.

The parser produces 5 syntactic ambiguities for this sentence. They arise from:

- part of speech ambiguity (preposition or adjective) of "about"

• attachment ambiguity of preposition/adjective "about"

Interpreting "about" as a preposition, we obtain three attachment ambiguities shown in Figure 4 (a), (b) and (c). Interpreting "about" as an adjective, "about 6 o'clock" has to be an adverb, and there is two attachment ambiguities shown in Figure 4 (d) and (e).

Interpretation of "about" as an adjective is preferred because it is directly followed by a quantity, and 1 is added to "Weak Preferable Factors" element when the construction rule is applied. On the other hand, prepositional interpretation of "about" leaves this element unchanged.

Interpretation that adverbial phrase "about 6 o'clock" depend on "is" is unpreferred because the distance is far, so -1 is added to "Other Preferences" element[2] (Figure 4 (e)).

For simplicity of explanation , here we suppose that only distance information is used for attachment disambiguation. To the "Attachment Ambiguities" element, negated distance between a preposition and its depending word is added in the attachment rule.

Combining these evaluations, the system can obtain the correct answer (e). It is easy to understand why this result is obtained, since each subtree itself contains a detailed scoring information.

4.4 Introducing New Evaluation Factors

At First, we had only the 1st, 2nd, 5th and 6th elements as the vector score elements. We introduced the 3rd and 4th elements later in the course of development.

Addition of these factors was simple. The only thing that need decision is how to include these factors into the totalizing function. They were given above weights because they should be overridden by strong preference such as collocations, and should override other weak negative preferences. After the totalizing function is settled, evaluation functions were written independently of other factors.

5 Related Work

In this section we briefly mention related works on disambiguation and their relation to this work.

One of central target of disambiguation has been the prepositional phrase attachment ambiguity resolution. For example, Kimball[3] proposed Right Association, in which postmodifiers tend to attach as low as possible. Frazier and Fodor[4] proposed Minimal Attachment, in which preferred a syntax tree with fewer nodes. In order to overcome some major problems Lexical Preference was proposed by Ford et al.[5].

Shieber[6] proposed a control strategy of depth-first shift-reduce parsing in which a syntactic tree satisfying RA, MA and LP is obtained first. His idea was to translate these syntactic preferences to parsing operation preferences. In this, RA is expressed as preferring shift to reduction in shift-reduce conflict. Although this idea is ingenious, and might shed some light on the human recognition process, this kind of approach has a serious problem: it is very difficult to introduce new disambiguation information to the system. Since the preference information is deeply embedded into the parsing algorithm itself, we need to devise a new parsing algorithm every time we introduce new information for disambiguation. In real applications it is often necessary to combine various lexical/syntactic/semantic information, in addition to proposed principles mentioned above.

This discussion explains the reason we focused solely on a framework, in which breadth-first parsing and scoring are combined[l, 2, 7, 8, 9, 10].

McRoy[7] reports word sense disambiguation method in which multiple disambiguation knowledge is combined. She uses word frequency, morphology, collocation, clustering, syntax, and various affinities between two words.

Our method mostly differs from hers in that the quantity propagated over the syntax tree is not a scalar but a vector. This property itself made it easier to treat re-evaluation phenomena and to introduce new evaluation factors. Her approach is also different from ours in that she claims there is no need for special knowledge to combine different evaluation factors, provided "specificity" is appropriately taken into account. But we believe this knowledge is essential for most natural language processing systems. Our claim is a bit different: Clear separation of evaluation knowledge

[2] In real system this information comes from a database outside and the value and the reason are a bit different form this explanation

for individual factors from knowledge for combining these factors is most important. This separation gave us good property when introducing new evaluation factors or writing grammar rules by more than one person.

Since the only thing our method requires is an evaluation score on some linguistic phenomena, it can be easily combined with preference information obtained by approaches such as statistics-based or example-based[14]. Our simple approach might offer a base for fusion of various old and new approaches to disambiguation.

6 Conclusion

In this paper we proposed a new method of preference score representation. The main characteristics of this method are the separation of the numerical score for final selection and the preference information that propagates over a tree. The latter was represented and propagated as a vector score. A numerical score is calculated from the vector score when a syntax tree is constructed.

By this method we can naturally write disambiguation process that require re-evaluation of some linguistic factor, like relative clause construction in English or part-of-speech disambiguation for a Japanese function word. The separation of knowledge allows us to easily introduce new evaluation factors, and to have multiple writers of rules.

References

[1] Maruyama, Naoko et al. (1988). "A Japanese sentence analyzer." IBM Journal of Research and Development 32(2), 238-250.

[2] Hobbs, Jerry. R et al. (1991). "Robust Processing of Real-World Natural-Language Texts." Proceedings of the 3rd Conference on Applied Natural Language Processing, 186-192.

[3] Kimball, John P. (1973). "Seven principles of surface structure parsing in natural language." Cognition 2, 15-47.

[4] Frazier, Lyn, and Fodor, Janet Dean (1979). "The Sausage Machine: a new two-stage parsing model." Cognition 6, 291-325.

[5] Ford, Marylyn; Bresnan, Joan; and Kaplan, Ronald (1982) "A Competence-Based Theory of Syntactic Closure." in "The mental Representation of Grammatical Relations" MIT Press.

[6] Shieber, Stuart M. (1983). "Sentence Disambiguation by a Shift-Reduce Parsing Technique." Proc. of the 21st Annual Meeting of the ACL, 113-118.

[7] McRoy, Susan W. (1992). "Using Multiple Knowledge Sources for Word Sense Discrimination." Computational Linguistics 18(1) pp.1-30.

[8] Wilks, Yorick; Huang, Xiuming, and Fass, Dan (1985). "Syntax, Preference and Right Attachment." Proc. IJCA1-85, 779-78-1.

[9] Tsujii, J. et al. (1988). "How to Get Preferred Readings in Natural Language Analysis." Proc. COLING-88.

[10] Nagao, K. (1990). "A Preferential Constraint Satisfaction Technique for Natural Language Analysis." Proc. of ECAI-92.

[11] Schubert, Lenhart (1986). "Are there preference trade-offs in attachment decisions?" Proc. AAAI-86, 601-605.

[12] Schubert, Lenhart (1984). "On Parsing Preferences." Proc. COLING-84, pp.247-250.

[13] Hobbs, Jerry R., and Bear, John (1990). "Two Principles of Prase Preference." Proceedings of COLING-90 vol.3 pp.162-167.

[14] Sumita, E. et al. (1991). "Experiments and Prospects of Example-Based machine Translation." Proc. 29th ACL.

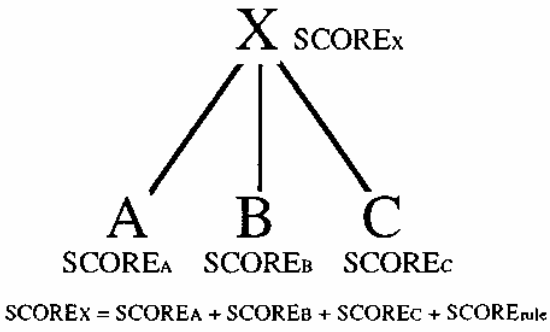


Figure1

Conventinal Scoring Method

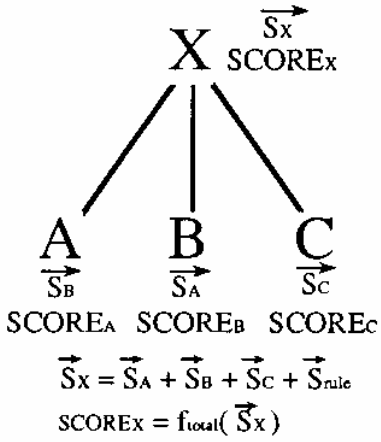
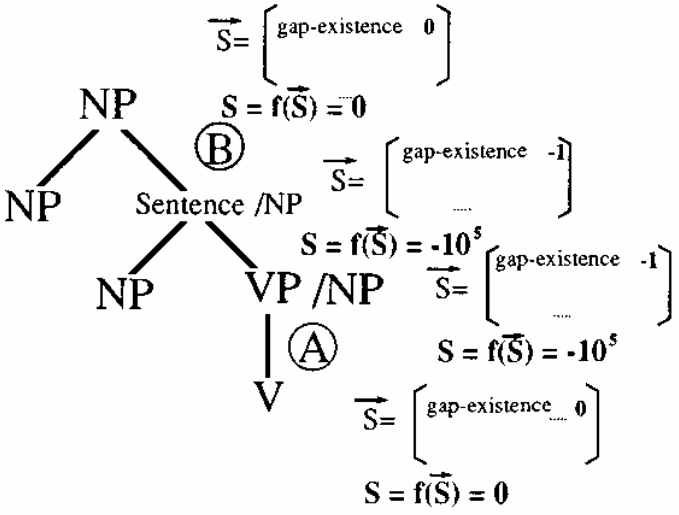


Figure2

Scoring by Proposed Method

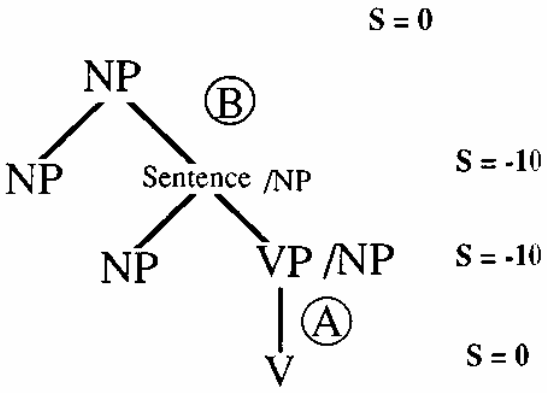
Figure 3

Treatment of a Gapped Structure



(a)

Proposed Method



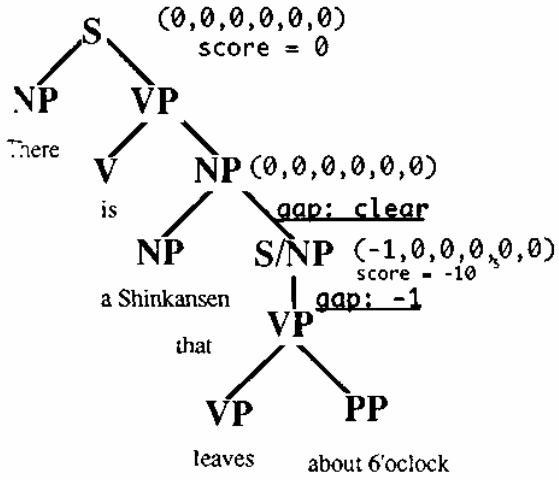
(b)

Conventional Method

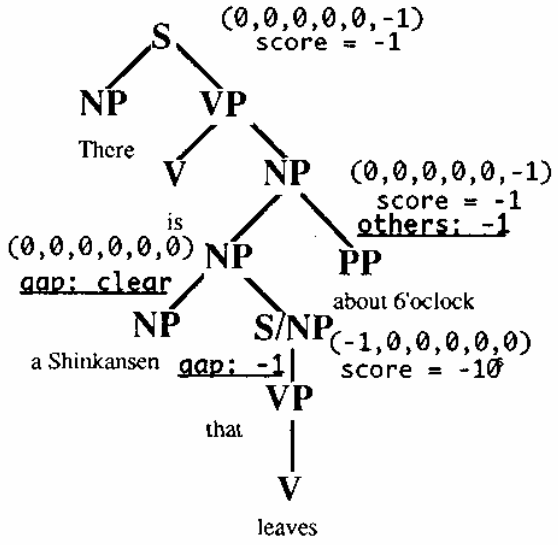
Fig. 4

An Example of Scores for Syntactic Ambiguities of

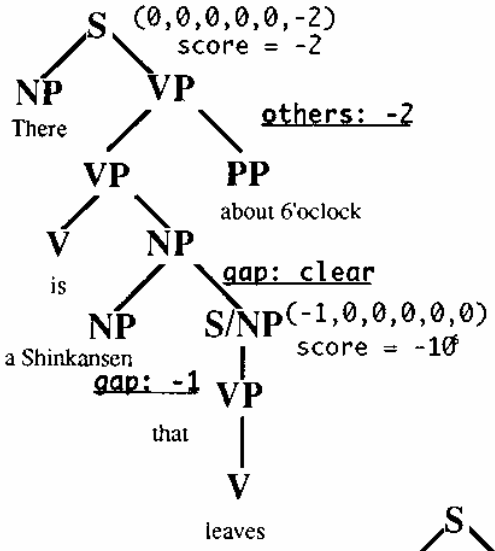
There is a Shinkansen that leaves about 6 'clock,



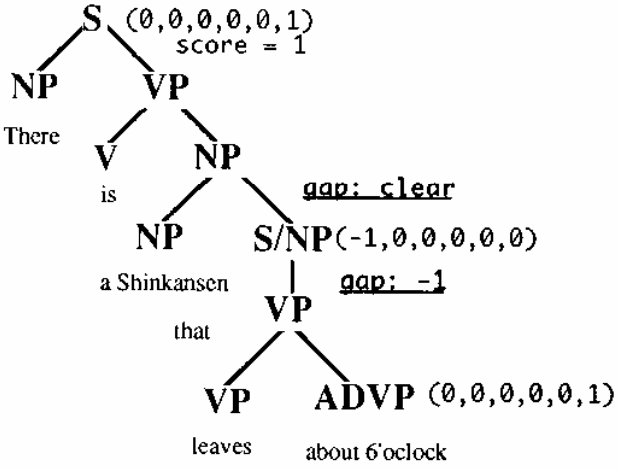
(a)



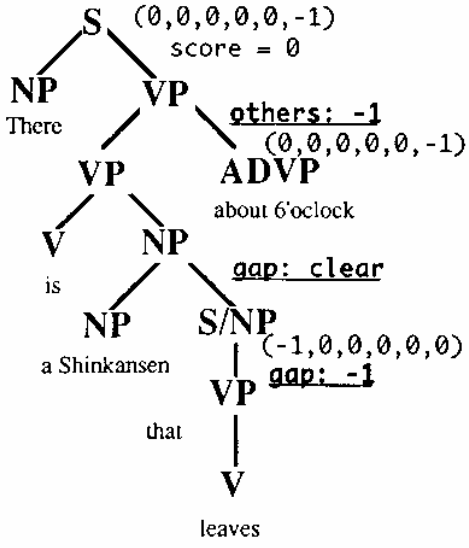
(b)



(c)



(d)



(e)