Proc. of the 1st US-Japan Workshop on Integrated
Comrehension and Generation Systems in Multi-Media Environments, 1991

A Generalized Discrimination Network and its Applications to Incremental Disambiguation and Ellipsis Resolution

Hozumi Tanaka, Manabu Okumura
Department of Computer Science,
Tokyo Institute of Technology
2-12-1, O-okayama, Meguro-ku, Tokyo, 152, Japan
Tel: (+81-3)3726-1111 ext. 4188
email: dora@cs.titech.ac.jp

A discrimination network has been used for various problem solving systems, especially in natural language processing to represent multiple word meanings compactly. The problem solving process corresponds to a step by step downward traversal of the network from the root node to a leaf node guided by branches which satisfy the obtained constraints. Although a discrimination network has the excellent characteristics, it has critical problems that it cannot be traversed unless constraints are entered in an a priori-fixed order. To solve the above problems, we proposed a 'generalized discrimination network(GDN)' which can be traversed in any order of constraints. The problem solving process can traverse the GDN immediately whenever any constraints are obtained. In this paper, we show how GDN can neatly deal with various problems in natural language processing, such as word sense disambiguation, ellipsis resolution. Merits of using GDN for them are described.

1 Introduction

A discrimination network is a generalization of a decision tree[3] and has been used for various problem solving systems[5], especially in natural language processing to represent multiple word meanings compactly[10, 15, 13, 1]. A discrimination network is considered as a directed acyclic graph with one root node and many leaf nodes. The solutions are located at each leaf node of the network. The problem solving process corresponds to a step by step downward traversal of the network from the root node to a leaf node guided by branches which satisfy the obtained constraints. A discrimination network has the following advantages:

- in the discrimination network, a unique node can represent multiple solutions which correspond to the leaf nodes below it. Therefore, the downward traversal of the network corresponds to the continuous refinement of a solution into a more specific one. This is what the constraint programming paradigm[7] will achieve;

- compared with a linear search, the discrimination network's search algorithm is more efficient because the downward traversal is guided by constraints which are labels of-branches, and the search space can be gradually narrowed down. This search takes time $O(l)$ in the worst case, where $l(= \log n)$ is the height of the network, while the linear search takes time $O(n)$ in the worst case, where n is the number of all possible solutions[2];

- we can compact a set of rules since some of the same preconditions(constraints) can be merged into one branch in the network. Therefore, whether a constraint is satisfied is checked only once for one constraint and wasteful repeated computations can be avoided.

Although a discrimination network has the excellent characteristics mentioned above, it has two critical problems. The first one is that it cannot be traversed unless constraints are entered in an a priori-fixed order. Since the order in which constraints are obtained cannot be a priori fixed in general, the process to traverse the network downward may often have to suspend until the constraint in the right order is obtained. Because the network is traversed downward from the root node, constraints must be entered one

by one from constraints which are labels of branches connected to the root node. This order depends on the original structure of the network. The second problem is more serious. If some constraints, which are necessary for traversing the network downward, are not obtained, the problem solving process will not be able to traverse the network and a deadlock will happen. In this situation, the constraints that have been already obtained will cease to contribute to the problem solving process.

To solve the above problems, we proposed a 'generalized discrimination network(GDN)' which is a variant of a discrimination network and can be traversed in any order of constraints[17]. The problem solving process can traverse the GDN immediately whenever any constraints are obtained.

In this paper, we show how GDN can neatly deal with various problems in natural language processing, such as word sense disambiguation, ellipsis resolution. Merits of using GDN for them are described.

In section two, a discrimination network and GDN are outlined. In section three, various usages of GDN for natural language processing are described. Finally, in section four the possibility of other applications is explored.

2 Ordinary and generalized discrimination networks

2.1 Characteristics of discrimination networks

Figure 1 is a sample discrimination network[1]. Each branch of the network has a constraint as its label. Each leaf node of the network points to a solution. Other nodes represent a set of possible solutions which correspond to leaf nodes below them because further traversal along branches to multiple nodes are possible from them. The root node corresponds to the set of all solutions.

The problem solving process using discrimination networks is a step by step downward traversal of the network from the root node to a leaf node guided by branches which satisfy the obtained constraints. In this process, inappropriate alternatives are rejected and appropriate solutions are selected. Reaching a leaf node means that a solution is found.

[1]For expository clarity, we use only tree-form examples in this paper.

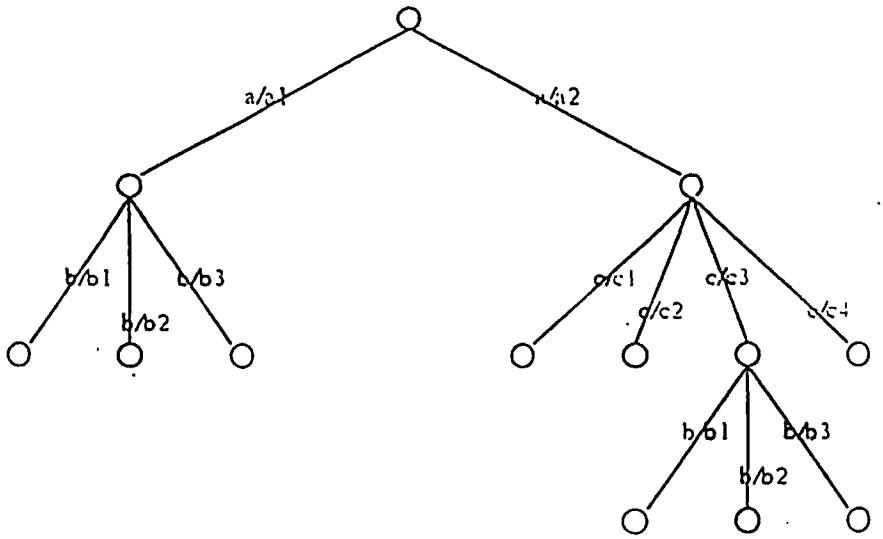

Figure 1: A sample discrimination network

As mentioned in section one, a discrimination network has three advantages. However, it also has two critical problems. Some solutions to the problems have been suggested[1, 23]. But as described in [17], we thought all of them are not satisfactory because they contain inner problems; therefore, we proposed a 'generalized discrimination network(GDN).'

2.2 Principles of generalized discrimination networks

In this section, principles of GDN are outlined. Consider the discrimination network shown in Figure 1. To represent the network as a table, we make stages of preparation. First, a numerical string is assigned to each node as a unique identifier. '1' is assigned to the root node. To each child node of the root node, an identifier of two digits $1i$(where i is an integer between 1 and n which represents the number of child nodes) is assigned. Similarly, to each child node of the node $1i_1i_2...i_m$, an identifier $1i_1i_2...i_mi$(where i is an integer between 1 and n which represents the number of child nodes) is assigned. To the nodes in Figure 1, identifiers are assigned as shown in Figure 2.

Second, constraint-identifier pairs are extracted from the network in the following form: a branch and a subordinate node which is directly connected by the branch. This correspondence between constraint and identifier means that if a constraint is satisfied, the nodes of corresponding identifiers can be reached in the network. For example, if constraint c/c_2 is satisfied, the

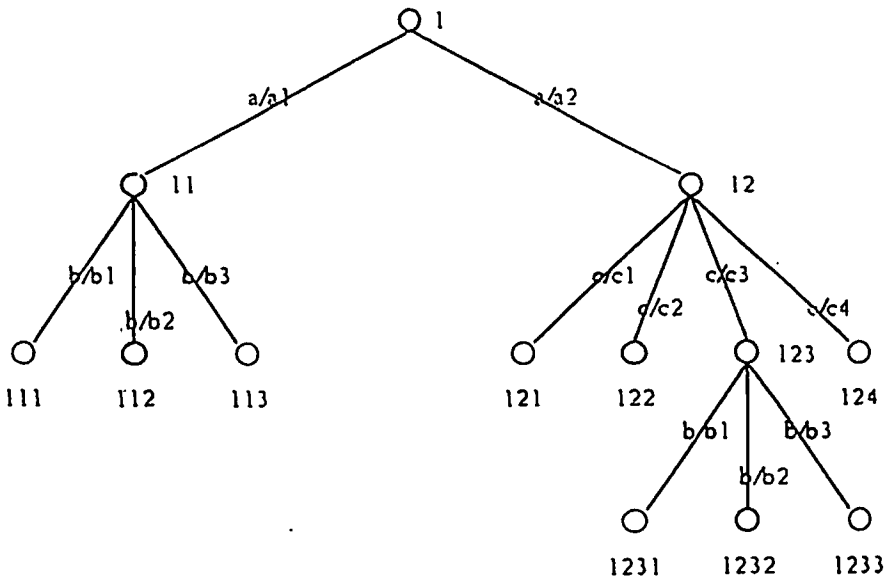

Figure 2: Discrimination network with identifier-assigned nodes

network can be traversed downward to the node of corresponding identifier 122.

Here, we must pay attention to the other constraints in the path from the root node to the reached nodes. In the case of the node of identifier 122 mentioned above, constraint a/a_2 exists and is unsatisfied. Therefore, the reachability of node 122 is 'conditional' in that node 122 can be reached if constraint a/a_2 is satisfied.

Hence from Figure 2, pairs of a constraint and a 'conditional identifier,' that is, an identifier of the conditionally reached node, are obtained in Table 1. A conditional identifier consists of an identifier of the reached node followed by an 'if-clause' which represents a list of unsatisfied constraints. An identifier with no if-clause means that a node of the identifier can be reached unconditionally. For the constraints of attribute name b, multiple pairs exist and so sets like $\{111\ if\ a/a_1, 1231\ if\ a/a_2\ and\ c/c_3\}$ correspond to them. The existence of multiple pairs means that multiple corresponding nodes can be reached if a constraint is satisfied.

The preparation is finished. The regular order of constraints is a/a_2, c/c_3, b/b_1 for traversal of the network in Figure 2 downward to node 1231. Here, in contrast, the case where constraints are obtained in the order of b/b_1, c/c_3 is considered. Notice that in this case, a necessary constraint a/a_2 is not obtained and the order of the obtained constraints is irregular.

a/a_1	11
a/a_2	12
b/b_1	$\{111\ if\ a/a_1, 1231\ if\ a/a_2\ and\ c/c_3\}$
b/b_2	$\{112\ if\ a/a_1, 1232\ if\ a/a_2\ and\ c/c_3\}$
b/b_3	$\{113\ if\ a/a_1, 1233\ if\ a/a_2\ and\ c/c_3\}$
c/c_1	$121\ if\ a/a_2$
c/c_2	$122\ if\ a/a_2$
c/c_3	$123\ if\ a/a_2$
c/c_4	$124\ if\ a/a_2$

Table 1: Correspondence between constraint and node identifier

The discrimination process in our approach for that case is briefly described below. Figure 3 shows a 'state' transition which represents the discrimination process. A state is in the form of a conditional identifier. The initial state(a state where no constraints are obtained) is 1 with no if-clause(the identifier of the root node). Informally[2], after constraint b/b_1 is obtained, the state is computed as follows, with the current state(the initial state) and a set of conditional identifiers $\{111\ if\ a/a_1, 1231\ if\ a/a_2\ and\ c/c_3\}$ corresponding to the obtained constraint by Table 1:

> Both 111 and 1231 include 1 as a prefix-numerical string, so the longer strings 111 and 1231 are returned. Because the current state has no if-clause, the if-clause of the next state becomes the same as the if-clause of the conditional identifiers corresponding to the obtained constraint. The if-clause of the obtained constraint represents a list of constraints between the current node(1) and the reached node($\{111, 1231\}$) except the obtained constraint, that is, $if\ a/a_1$,$if\ a/a_2\ and\ c/c_3$ respectively. Therefore, the next state becomes $\{111\ if\ a/a_1, 1231\ if\ a/a_2\ and\ c/c_3\}$.

As shown in Figure 2, identifiers of mutually reachable nodes in the network are in prefix-numerical string relation with each other. Therefore, the operation between identifiers can easily check whether one node can be

[2]For formal detail, please refer to [17].

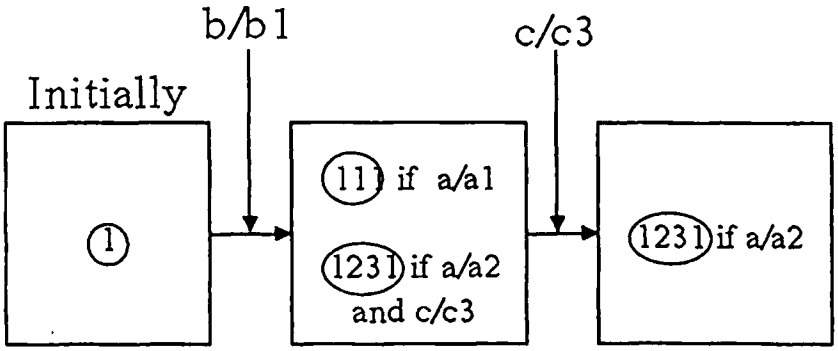

Figure 3: The discrimination process using GDN

reached from the other in the network. If one node is reachable from the other, the identifier of the subordinate one is returned. This operation corresponds to a downward traversal of the network by satisfying the obtained constraints. The analysis will fail if one identifier is not a prefix of the other.

Next, constraint c/c_3 corresponds to conditional identifier 123 *if* a/a_2. When the constraint is obtained, the state has multiple conditional identifiers $\{111\ if\ a/a_1, 1231\ if\ a/a_2\ and\ c/c_3\}$, and so the operations are performed on each conditional identifier with the constraint:

> As for 111 *if* a/a_1, the identifiers are not in a prefix relation, so the analysis fails. Therefore, the result is necessary only for conditional identifier 1231 *if* a/a_2 *and* c/c_3. The resultant identifier is 1231 from identifiers 123 and 1231. The resultant if-clause is *if* a/a_2 because constraint c/c_3 in the if-clause of the current state is obtained and removed from it. Hence the final result of traversal for the above case becomes 1231 *if* a/a_2, which means that node 1231 is reachable if constraint a/a_2 is obtained.

The if-clause allows us to cope with the irregular order of the obtained constraints and the lack of necessary constraints. The if-clause is the storage of constraints which must be obtained to reach the destination node but have not been obtained yet. When constraints are obtained in irregular order, the constraints in the wrong order can be found in the if-clause and are removed from it.

Thus constraints in the if-clause are expected to be obtained, and so they can be used as predictions for constraints which will be obtained later. In another viewpoint, constraints in the if-clause can be regarded as assumptions in the abductive reasoning[9, 16] because the if-clause means that the

conclusion of a rule(corresponding to the leaf node) holds and so preconditions(constraints) in the if-clause must hold in the future.

3 Various usages of GDN for natural language processing

In this section, we show how various problems in natural language processing can be dealt with by GDN. We treat word sense disambiguation and ellipsis resolution.

Figure 4 is a portion of the discrimination network which represents the word senses(and case frames) of the Japanese verb 'naosu.' Each branch of the network has as its label a selectional restriction on surface cases such as postpositions 'ga,' 'wo,' and so on. Each leaf node of the network points to a unique word sense, which is represented by the underlined label . Other nodes represent ambiguous word meanings which include all word senses corresponding to the leaf nodes below them because from these, the further traversal along branches to multiple nodes is possible. The root node corresponds to the most ambiguous meaning: it is a representation which includes all leaves, namely, all word senses. A set of constraints in the path from the root node to a leaf node represents the case frame for the word sense corresponding to the leaf node.

The word sense disambiguation process using a discrimination network is a step by step downward traversal of the network from the root node to a leaf node guided by branches which satisfy the obtained constraints. In this process, semantically inappropriate alternatives are rejected and appropriate word senses are selected by virtue of information about other words in the sentence. The reaching of a leaf node means that the ambiguity has been fully resolved.

Works such as [10, 15, 13, 1] realize the word sense disambiguation process as a downward traversal of the discrimination network. In these works, the merits of such a network are described, as mentioned in section one.

Using GDN instead of a discrimination network adds another advantage in that the word sense disambiguation process proceeds incrementally. We think incremental disambiguation[14] is a better strategy for word sense disambiguation because a combinatorial explosion of the number of total am-

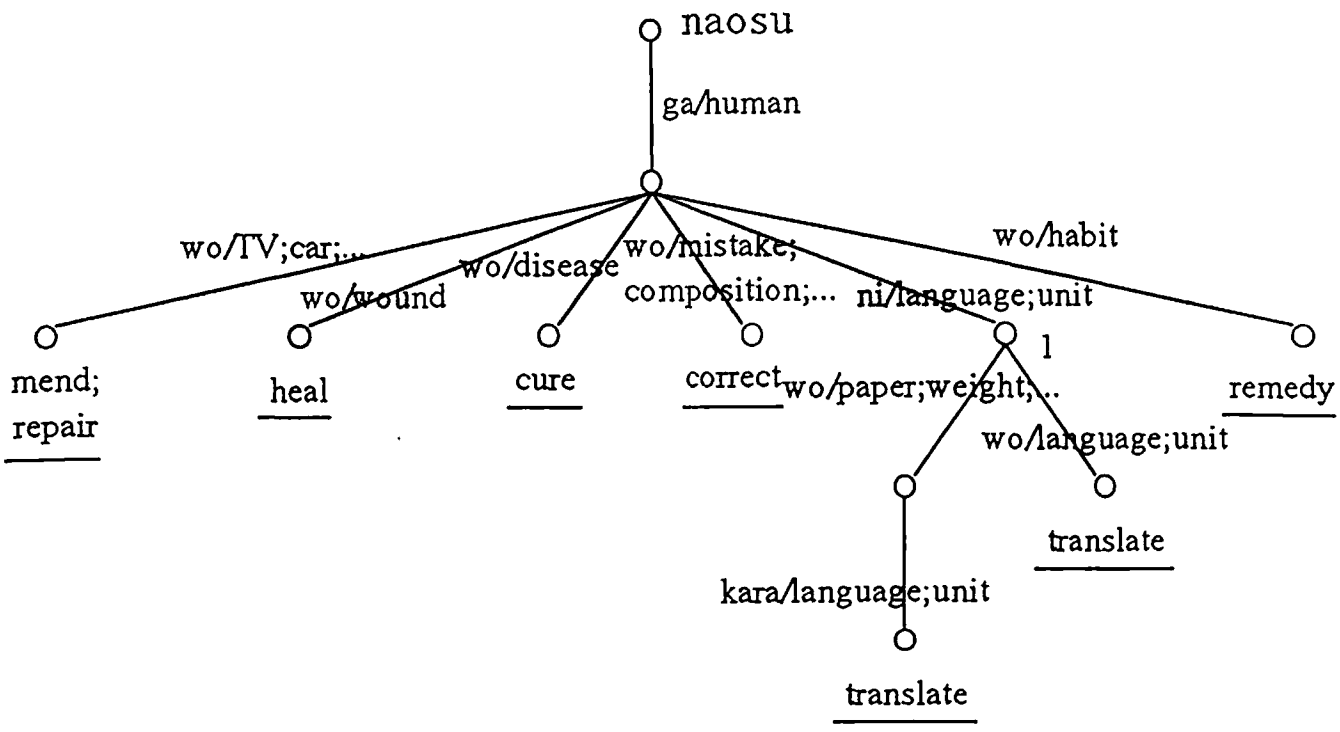

Figure 4: A portion of the discrimination network of the word senses of the verb 'naosu'

biguities might occur unless word sense ambiguity is incrementally resolved as early as possible whenever constraints are obtained during the analytical process of a sentence.

The incremental disambiguation approach allows us to deal with cases as well where it is impossible to disambiguate unless information about succeeding sentences is taken into account. This is because the incremental disambiguation process is considered to be the refinement of ambiguous(undetermined) results of semantic processing by newly obtained constraints and makes it easy to represent a partial semantic interpretation of a sentence and disambiguate it by additional constraints from subsequent inputs.

Notice that the partial traversal of the discrimination network is considered to be the representation of the ambiguous result of partial semantic processing; the ambiguous result is easily represented by nodes except leaf nodes in the network. Making a new decision by additional constraints from subsequent inputs naturally corresponds to traversing from the current node further downward using newly obtained constraints. Discrimination networks are therefore well suited for the incremental disambiguation approach.

However, because the order in which constraints are obtained incrementally might be deviated from an a priori-fixed one and some constraints which are necessary for traversing the network downward might not be obtained, it is not always possible to traverse the network downward during the analytical process. GDN solves these problems of the discrimination network that prevent the incremental disambiguation approach and enables the downward traversal of the network, which uses the incrementally obtained constraints during the analytical process. As surface variations such as relative clauses and passive forms in English show, problematic situations to the discrimination network often happen. In Japanese, the situation is worse. Because Japanese has greater word order freedom, the degree of deviation from the a priori-fixed order is considerable. In addition, in Japanese, phrases are often omitted that can be easily guessed from the contextual information. Therefore, we think GDN is essential to the analysis of such a language as Japanese.

Next, we describe how GDN deals with ellipsis resolution which fills in the missing phrases. Because GDN is a variant representation form of a set of case frames, the ellipsis resolution approach with GDN is case frame-based[4, 21]. It can detect easily which phrases are omitted and can test the possible substitutions for the missing phrases by means of the selectional restriction, as described below. However, it requires another mechanism that finds candidates for the substitution and decides the order of priority among them from the contextual information[20, 11]. The mechanism is out of the scope of this paper and the naive heuristic 'recency' is assumed.

The positions of the missing phrases are detected in two ways with GDN. First, they are indicated by the if-clause mentioned in section two because constraints in the if-clause are not obtained but are expected to be. Secondly, in the case where the reached node is not a leaf node when the analysis of the whole sentence is finished, constraints between the current node and leaf nodes are considered to be the positions of the missing phrases because the verb should have a unique word sense[3]. So in such case, ellipsis resolution enables the further disambiguation of the word sense. By applying to the candidates for the substitution the selectional restriction which corresponds

[3]Of course, another mechanism for checking the syntactic grammaticality is necessary in the case of languages such as English because it is strange to say that 'I repair.' is grammatical and the object is omitted in the sentence. As for Japanese, such a mechanism seems unnecessary.

to the detected position of the missing phrase, the semantic appropriateness of the substitution is checked.

Finally, we illustrate how incremental word sense disambiguation and ellipsis resolution proceed jointly in Japanese. Consider the sentences:

> tarou ga nihongo de ronbun wo kaki,
> (Tarou Japanese a paper wrote)
> eigo ni hanako ga naosita.
> (English Hanako)
>
> Tarou wrote a paper in Japanese, and Hanako translated it into English.

In the example, the word sense of the verb 'naosu' is ambiguous as shown in the network of Figure 4[4] and the order of obtained constraints 'ni/eigo(English), ga/hanako(Hanako)' is irregular compared with the one of the network. Constraint 'ga/human' must be obtained earlier than 'ni/language;unit' in the network. Additionally, the phrase corresponding to 'it' is omitted.

At the end of the first sentence, the candidates for ellipsis resolution is ['ronbun'(a paper), 'nihongo'(Japanese), 'tarou'(Tarou)], where the first element is preferred. The second sentence is analyzed as follows:

> 'Eigo'(English) is a language and satisfies a selectional restriction of a postposition 'ni' and node 1 is reached. Next, 'hanako'(Hanako) satisfies a selectional restriction of 'ga' and the reachability of node 1 becomes unconditional. The word sense ambiguity of the verb 'naosu' has been reduced from seven to two, but the word sense is still ambiguous[5]. Because the reached node does not point to a unique word sense, the further traversal is tried from it. As the result, we get the candidate positions of the missing phrases, that is, {'wo/language;unit'}, {'wo/paper;weight','kara/language;unit'}. Then, ellipsis resolution is tried using the candidates mentioned above. Because 'ronbun'(a paper) is a paper and 'nihongo'(Japanese) is a language, they satisfy a selectional restriction of postpositions 'wo' and 'kara' respectively. Thus, the substitutions are decided

[4] However, only the correct word sense is written in the English translation.

[5] We think there is a subtle difference between the remaining two word senses because they have a different case frame.

and the word sense of the verb 'naosu' is uniquely determined. The final result for the second sentence is 'Hanako translated the paper(which Tarou wrote) from Japanese into English.'

4 Conclusion

We showed how GDN can neatly deal with various problems in natural language processing, such as word sense disambiguation, ellipsis resolution and parsing of word order-free languages. We also described merits of using GDN for them.

Lastly, other possible applications shall be mentioned. First, the parsing mechanism using GDN can neatly handle word order-free languages. ID/LP framework in Generalized Phrase Structure Grammar formalism[8] can write grammar rules of those languages in a general way. And our GDN-based parsing mechanism augmented with a Hasse diagram[6] can parse directly and efficiently with ID/LP rules. Please refer to [22] for the details of the parsing algorithm.

Second, we think GDN realizes a better way to organize(index) MOPs. MOP(Memory Organization Packages)[18] is considered to be a hierarchically-organized memory for scripts[19] and is often used to understand stories in specific domain, such as articles about terrorism. Usually MOPs are indexed in the form of multiple redundant discrimination networks[12]. 'Multiple redundant' networks, where like a lattice, multiple redundant paths are provided to reach the same script, are inevitable if order freedom of obtained constraints is considered because of the problems of discrimination networks mentioned in section one. Our GDN, however, solves these problems and can organize MOPs more concisely without redundant links.

References

[1] G. Adriaens and S.L. Small. Word expert parsing revisited in a cognitive science perspective. In S.L. Small, G.W. Cottrell, and M.K. Tanenhaus, editors, *Lexical Ambiguity Resolution : Perspectives from Psycholinguistics, Neuropsychology, and Artificial Intelligence*, pages 13–43. Morgan Kaufmann Publishers, 1988.

[2] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms.* Addison-Wesley, 1983.

[3] G. Brassard and P. Bratley. *Algorithmics.* Prentice Hall, 1988.

[4] J.G. Carbonell. Discourse pragmatics and ellipsis resolution in task-oriented natural language interfaces. In *Proc. of the 21st Annual Meeting of the Association for Computational Linguistics*, pages 164–168, 1983.

[5] E. Charniak, C.K. Riesbeck, and D.V. McDermott. *Artificial Intelligence Programming.* Lawrence Erlbaum Associates, 1980.

[6] D.J. Cooke and H.E. Bez. *Computer Mathematics.* Cambridge University Press, 1984.

[7] M. Dincbas. Constraints, logic programming and deductive databases. In *Proc. of the France-Japan Artificial Intelligence and Computer Science Symposium 86*, pages 1–27, 1986.

[8] D. Gazdar, E. Klein, G. Pullum, and I. Sag. *Generalized Phrase Structure Grammar.* Basil Blackwell, 1985.

[9] J.R. Hobbs, M. Stickel, P. Martin, and D. Edwards. Interpretation as abduction. In *Proc. of the 26th Annual Meeting of the Association for Computational Linguistics*, pages 95–103, 1988.

[10] P.S. Jacobs. Concretion: Assumption-based understanding. In *Proc. of the 12th International Conference on Computational Linguistics*, pages 270–274, 1988.

[11] M. Kameyama. *Zero Anaphora : The Case of Japanese.* PhD thesis, Stanford University, 1985.

[12] J.L. Kolodner. *Retrieval and Organizational Strategies in Conceptual Memory.* Lawrence Erlbaum Associates, 1984.

[13] S.L. Lytinen. Are vague words ambiguous? In S.L. Small, G.W. Cottrell, and M.K. Tanenhaus, editors, *Lexical Ambiguity Resolution : Perspectives from Psycholinguitics, Neuropsychology, and Artificial Intelligence*, pages 109–128. Morgan Kaufmann Publishers, 1988.

[14] C. S. Mellish. *Computer Interpretation of Natural Language Descriptions.* Ellis Horwood, 1985.

[15] G.D. Moerdler and K.R. McKeown. Beyond semantic ambiguity. In *Proc. of the 7th National Conference on Artificial Intelligence*, pages 751–755, 1988.

[16] P. Norvig and R. Wilensky. A critical evaluation of commensurable abduction models for semantic interpretation. In *Proc. of the 13th International Conference on Computational Linguistics*, volume 3, pages 225–230, 1990.

[17] M. Okumura and H. Tanaka. Towards incremental disambiguation with a generalized discrimination network. In *Proc. of the 8th National Conference on Artificial Intelligence*, pages 990–995, 1990.

[18] R. Schank. *Dynamic Memory: A Theory of Learning in Computers and People.* Cambridge University Press, 1982.

[19] R. C. Schank and R. P. Abelson. *Scripts, Plans, Goals and Understanding.* Lawrence Erlbaum Assoc., 1977.

[20] C. L. Sidner. Focusing in the comprehension of definite anaphora. In M. Brady and R. C. Berwick, editors, *Computational Models of Discourse*, pages 267–330. MIT Press, 1983.

[21] A. Stolcke. Gapping and frame semantics: A fresh look from a cognitive perspective. In *Proc. of the 13th International Conference on Computational Linguistics*, volume 2, pages 341–346, 1990.

[22] M. Surapant, M. Okumura, and H. Tanaka. Direct ID/LP parsing with a generalized discrimination network. In *Proc. of the 14th International Conference on Computational Linguistics*, 1992. to be submitted.

[23] W.A. Woods. Taxonomic lattice structures for situation recognition. In *Theoretical Issues in Natural Language Processing 2*, pages 33–41, 1978.