Learning from Problem Solving and Communication: A Computational Model for Distributed Knowledge Systems

Takao TERANO
Graduate School of Systems Management, The University of Tsukuba, Tokyo
email: terano@gssm.otsuka.tsukuba.ac.jp

Abstract [1]

This paper examines the issues of machine learning in distributed knowledge systems, which will consist of distributed software agents with problem solving, communication and learning functions. To develop such systems, we must analyze the roles of problem solving and communication capabilities among individual agents or knowledge systems. To facilitate the analyses, we propose a computational model: LPC. The model consists of a set of agents with (a) a knowledge base for learned concepts, (b) a knowledge base for the problem solving, (c) a prolog-based inference mechanisms, and (d) a set of beliefs on the reliability of the other agents. Each agent can improve its own problem solving capabilities by inductive and/or deductive learning on the given problems and by reinforcement learning on the reliability of communications among the other agents. An experimental system of the model has been partially implemented in Prolog language. Experiments were carried out to examine the feasibility of the machine learning mechanisms of agents for problem solving and communication capabilities. The experimental results suggest that the proposed model is executable for analyzing the learning mechanisms applicable to distributed knowledge systems.

1 Introduction

Recently, a great deal of arguments have been devoted to the study of Very Large-scale Knowledge Based Systems (VLKBs). There are two approaches in developing VLKBs. One approach is to develop a centralized system which can be used for various objectives as is stated in the Cyc project [Lenat *et al.*, 1990]. In this approach, it is vital to facilitate the reuse of large amounts of knowledge according to the changes of the context in use.

The other approach is to develop many cooperative knowledge systems in distributed environments. The concepts proposed in Knowledgeable Community [Nishida *et al.*, 1993] and Knowledge Sharing Effort by DARPA [Neches 1991]) are categorized into this approach. Knowledge sharing, problem solving, and communication are key issues for such systems. When one knowledge system cannot solve a given problem by itself, it must ask other knowledge systems for intermediate results through queries on the problem it is solving, and it is desirable to learn the final results for future use. This paper focuses on such situations.

In this paper, we propose a computational model: LPC. In the model, we assume that (1) a set of problems is given to one of the agents, (2) any single agent cannot solve the problems, thus, the agents

[1] Presented *Int. Symp. on Fifth Generation Computer Systems 1994 (FGCS'94)* Workshop on Heterogeneous Cooporative Knowledge-Bases (W3), Dec., 15-16, 1994, Tokyo, Japan.

must communicate each other, (3) the agents have abilities to learn from both problem solving results and communications, and (4) the model has sufficient knowledge in total to solve the given problems. The model consists of a set of agents with (a) a knowledge base for learned concepts, (b) a knowledge base for the problem solving, (c) prolog-based inference mechanisms, and (d) a set of beliefs on the reliability of the other agents. Each agent can improve its own problem solving capabilities by inductive and/or deductive learning on the given problems and by reinforcement learning on the reliability of communications among the other agents.

This paper is organized as follows: In section 2, the background and motivation of the research is briefly described. In section 3, we explain the features of the proposed learning model. In section 4, intermediate experimental results are described. In section 5, we discuss related work, then in section 6, concluding remarks follows.

2 Background and Motivation for the Learning Model

Very large knowledge system development efforts (e.g., Cyc project [Lenat *et al.*, 1990]) have not usually discussed the problems of learning in distributed environments. But such discussions will become one of the main themes of distributed knowledge systems. In this section, we briefly survey the state-of-the-art researches on distributed knowledge systems in order to clarify our research motivation.

There are several researches on knowledge sharing and distributed knowledge systems. The first example is Knowledge Sharing Effort (KSE) [Neches 1991]. KSE presents a vision of the future in which knowledge-based system development and operation is facilitated by the infrastructure and technology of knowledge sharing. It describes an initiative currently under way to develop these ideas and suggests steps that must be taken in the future to try to realize this vision.

The second example is Knowledgeable Community. Nishida [Nishida *et al.*, 1993] describes the scope and goals of the Knowledgeable Community project and discuss how sharing of knowledge is achieved in the knowledgeable community with a framework of knowledge sharing and reuse based on a multi-agent architecture. In particular, they focus on the organizational structure that facilitates mediation between those agents requesting a service and those providing the service.

The visions proposed in KSE and Knowledgeable Community are quite plausible for future distributed knowledge systems; however, we believe that a simple and executable model for such excellent future visions must be developed.

The third example is PACT (Palo Alto Collaborative Testbed) [Cutkovsky *et al.*, 1993], in which the researchers develop a concurrent engineering infrastructure that encompasses multiple sites, subsystems, and disciplines. Currently, PACT is one of few examples of distributed knowledge systems; however, the framework proposed in PACT is too specific for general models of knowledge sharing. Again, we believe that a model which is able to generalize the characteristics of specific distributed knowledge systems must be developed.

Compared with the researches for learning agent systems, (for example, by [Weiss 1993]), our motivation is to give some insights for developing a general framework of distributed knowledge systems. As is discussed in the other literature, in [Terano 1993], we also think the models should be used as a common framework to analyze the problems not only of the future distributed knowledge systems, but also of computer supported cooperative works (CSCWs), Computer Integrated Manufacturing (CIM) [Cutkovsky *et al.*, 1993], distributed information systems [Papazoglou *et al.*, 1992], and organizational behaviors of social and software agents [Carley *et al.*, 1993], [CACM 1994].

In these applications, adding to planning and problem solving problems among agents in conventional distributed artificial intelligence researches [Bond and Gasser, 1988], [Fox, 1981], the problems of learning capabilities of both each agent and the whole system seem to affect the knowledge sharing, problem solving, and communication facilities. If a user requires one of the knowledge systems to solve some problems, that is, as we use the WWW in Internet, the system will generate *one* feasible solution within the context the user has specified. The results should be learned for the future reuse. Such learned results are fully context dependent and may differ from each other, when another knowledge system solved the similar problems in the different context, where we might abandon the concepts of completeness of the knowledge or common knowledge.

This paper examines a multi-agent learning model for such distributed systems and describes some intermediate experimental results. The main objective of the research is to verify and validate the feasibility of learning mechanisms to improve problem solving and communication capabilities of multi-agents in distributed environments.

3 A Computational Model for Distributed Knowledge Systems

The problems given to LPC consist of a set of tuples: $\{(G, E)\}$, where G is the goal of the problem, and E is a set of examples common to all the agents. E is assumed to be static during the problem solving. The information of Es are used to improve the problem solving performance in similar manners found in *Multistrategy learning systems*[Michalski 1993], [Michalski 1994]. If E is empty, the problems become the same as the ones in distributed problem solving. If E contains plural examples, the agents can utilize them for inductive learning, and E contains only one example, the agents can use it for deductive learning, when the agents have sufficient domain knowledge. The goal G is divided into some sub-goals by each agent in the following way, if necessary.

The agents of LPC solve a given problem in part by using the learned results and their own communication and problem solving knowledge. If they cannot solve the goal, they decompose it into sub-goals, and then request the other agents to solve them. The requests are done based on the memory of the other agents' information. In order to improve the problem solving performance for a set of given similar problems, (1) the agents of LPC compile the problem solving knowledge both within and among agents via deductive and/or inductive learning techniques, and (2) the agents also organize themselves by changing the information on which agents they should request to solve the (sub-)problems.

The agents in the proposed model are considered to be the model on heterogeneous knowledge bases, because the domain knowledge, problem solvers, and communication facilities can be differ from each other, when we implement concrete systems.

3.1 The Agents of LPC

The architecture and problem solving of the LPC agents is summarized in Figure 1.

The agents of LPC are defined by the following five tuples:

$$A_i = (\ LC_i,\ KA_i,\ D_i,\ PS_i,\ COM_i\ )$$

1. **Learned Concepts** LC_i: The agents memorize the operational knowledge [Keller 1988] for learned concepts of problem solving knowledge. The agents first try to use them for new problem solving. If they are not applicable, use their communication knowledge and communication and/or problem solving functions.
2. **Knowledge on the other agents** KA_i: The KA_i consists of a set of tuples: $(\ \tilde{T}_k,\ \tilde{A}_j\ ,\ R_j\ )$, where, $\tilde{A}_j$ is the name of other agent, $\tilde{T}_k$ is the name of (sub-)problem sent to it, and R_j is the

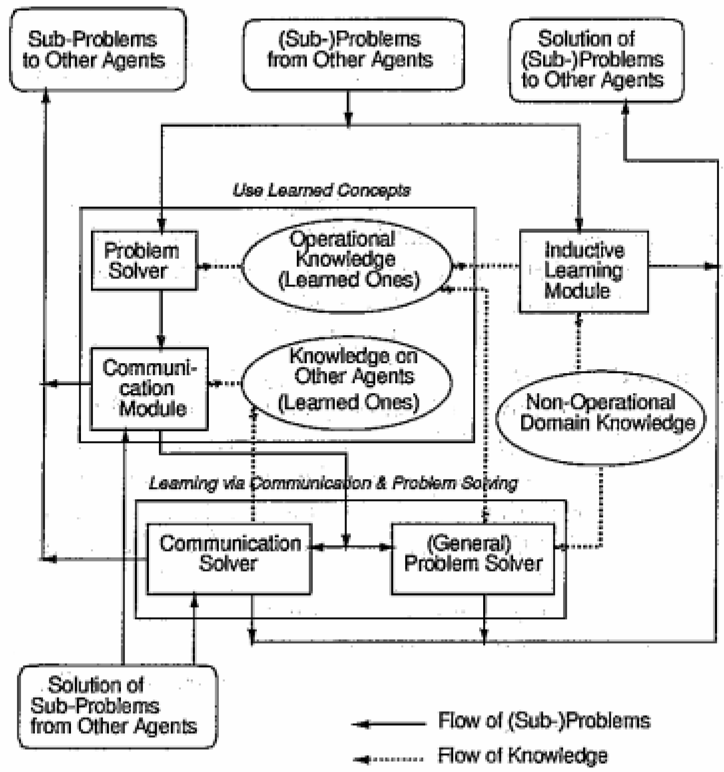

Figure 1: Architecture and Problem Solving of LPC Agents

credit value computed from $\tilde{A}_j$ and $\tilde{T}_k$. That is

$$KA_i = \{ (\tilde{T}_k , \tilde{A}_j , R_j)\}.$$

Each agent can memorize a given number of tuples at a time.

3. **Domain Knowledge D_i:** The knowledge D_is of the agents are consistent with, but different from each other. The contents of the D_is cannot be transferred among the agents; that is, the agents do not know who has the correct problem solving knowledge. When the inductive learning function is executed, D_i is used as the bias for the learning. When general problem solving or deductive learning functions are executed, D_i is used as non-operational domain knowledge.

4. **Problem Solver PS_i:** Using PS_is, the agents solve the given (sub-)problems. The agent generally executes PS_i twice: first using only LC_i, second using both LC_i and D_i. PS_i is further equipped with some inductive learning functions. The power of PS_is can be different in each agent.

5. **Communication Value COM_i:** Each agent A_i tentatively records communication values COM_i for other agents A_k. The value of COM_i has one of the following forms:

$$(\textit{ send}, \tilde{T}_k , \tilde{A}_i , \tilde{A}_k),$$

$$(\textit{ succeeded}, \tilde{S}_k , \tilde{A}_i , \tilde{A}_k),$$

$$(\textit{ failed}, \tilde{T}_k , \tilde{A}_i , \tilde{A}_k),$$

where S_k means the (sub-)solutions of $\tilde{T}_k$. The value of COM_i is used to update KA_i.

3.2 Problem Solving in LPC

The agents try to solve given (sub-)problems) by using learned concepts LC_i, knowledge on other agents KA_i, or inductive learning via D_i, then try to learn via communication and/or general problem solving. The problem solving processes of the agents are summarized in the following. Steps 2, 3 and 4, and steps 5 and 6 can be executed different orders depending upon the features of the given problems. This may cause some implementation issues.

1. **Problem Accepting:** When A_i gets a message from another agent A_k, A_i tries to accept a (sub-)problem $\tilde{T}_k$.

2. **Problem Solving via Inductive Learning** Using common examples E and D_i, PS_i tries to execute inductive learning. When the results are successful, A_i returns the results and update LC_i.

3. **Problem Solving via Learned Concepts** Agent A_i solves $\tilde{T}_k$ via PS_i, if it can solve or decompose $\tilde{T}_k$ using LC_i. When the results are successful, A_i returns the results.

4. **Problem Solving via Knowledge on Other Agents** When A_i fails in the above step, it will inform the name of another agent A_j to A_k by using KA_i, if R_j of KA_i has a large enough reliability value, which means A_j has a higher possibility of solving $\tilde{T}_k$. When the results are successful, A_i returns the results.

5. **General Problem Solving:** PS_i tries to solve $\tilde{T}_k$ using both LC_i and D_i. When the results are successful, A_i returns the results and update LC_i. If it fails, the results are moved to COM_i.

6. **Solving New Communication Paths:** When the results of the problem solving are successful, A_i returns the results. When the results of the problem solving are not successful, COM_i opens the communication to another agent A_j. The communication trials continue until another agent accepts it. Upon acceptance, COM_i updates KA_i.
7. **Credit Value Computing:** After the execution of Steps 4 or 6, The credit value R_j increases when the agent $\tilde{A}_j$ succeeds to solve $\tilde{T}_k$, and decreases when $\tilde{A}_j$ fails to solve $\tilde{T}_k$. The values are computed based on the techniques of profit sharing pattern type credit assignment algorithms in genetic algorithms and reinforcement learning.

4 Experimental Results

4.1 List Replacement Problem

In the experiments, we use the *list replacement problem* as a testbed. Simple illustrations of the problem are shown in Figure 2. A list of items with constants or variables accompanied with or without examples is given to the model. The goal of the problem solving is to make the given list empty, by decomposing, deleting or replacing the elements via knowledge among the agents. The problem is quite simple but scalable in the size. The problem aims to simulate cooperative works of multi-agents in manufacturing processes, office works, query processing in distributed systems, and so on. Therefore, we think the problem is just complex enough to assess the initial feasibility of the proposed model.

```
(1) List Replacement Problem without Examples
  Initial Problem: ([a, cd, bcd], ())
Gaol:             ([], ())
-- Problem solving by Agent-1:
   Knowledge:   a --> cd  Result: ([cd, cd, bcd],())
                          %%% Divided into a Subproblem
-- Problem solving by Agent-2:
   Knowledge:   cd  --> e  Result: ([e, e, be], ())
                          %%% Aggregate Subproblems
-- Problem solving by Agent-3:
   Knowledge:   b  --> nil  Result: ([e, e, e], ())
                          %%% Partial Problem Solved
-- Problem solving by Agent-4:
   Knowledge:   e  --> nil  Result: ([], ())
                          %%% All the Problem Solved
```

```
(2) List Replacement Problem with Plural Examples
  Initial Problem: ([a, cd, bcd],
                   (([a, cd, bcd] -> [a, d, bcd])
                    ([a, cd, bcd] -> [a, cd, bd])
                    ([a, cd, bcd] -> [a, d, bd])))
 -- Problem solving by Agent-1:
    Knowledge:   a --> cd  Result: ([cd, cd, bcd],())
                     ([cd cd, bcd],
                      (([a, cd, bcd] -> [a, d, bcd])
                       ([a, cd, bcd] -> [a, cd, bd])
                       ([a, cd, bcd] -> [a, d, bd])))
                        %%% Divided into a Subproblem
 -- Problem solving by Agent-5:
   Knowledge:      Result:  c --> nil
                        %%% Inductive Learning
                     ([d, d, bd],
                      (([cd, cd, bcd] -> [a, d, bcd])
                       ([a, cd, bcd] -> [a, cd, bd])
                       ([a, cd, bcd] -> [a, d, bd])))
                        %%% Partial Problem Solved
 -- Problem solving by Agent-3:
   Knowledge:  b --> nil
                     ([d, d, d],
                      (([a, cd, bcd] -> [a, d, bcd])
                       ([a, cd, bcd] -> [a, cd, bd])
                       ([a, cd, bcd] -> [a, d, bd])))
 ...
```

Figure 2: Illustrative Examples of *List Replacement Problem*

4.2 Experiments on Problem Solving and Communication

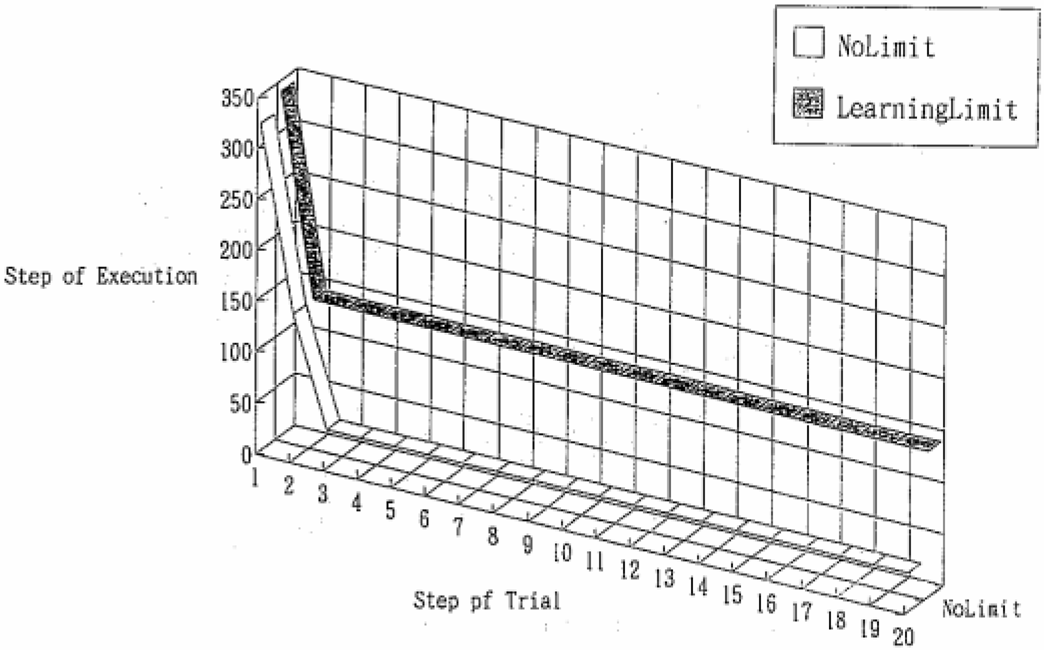

Figure 3: Learning from Problem Solving in LPC Model

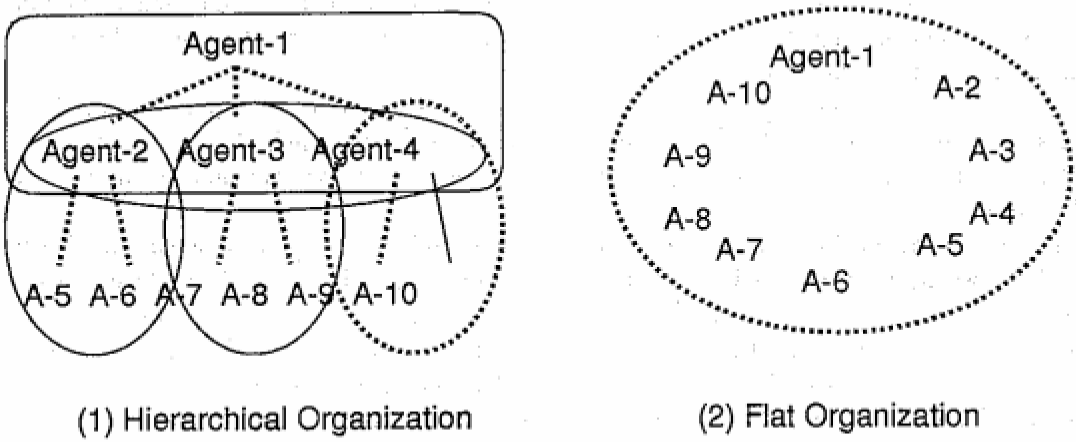

Figure 4: Organizational Structures in the Experiment

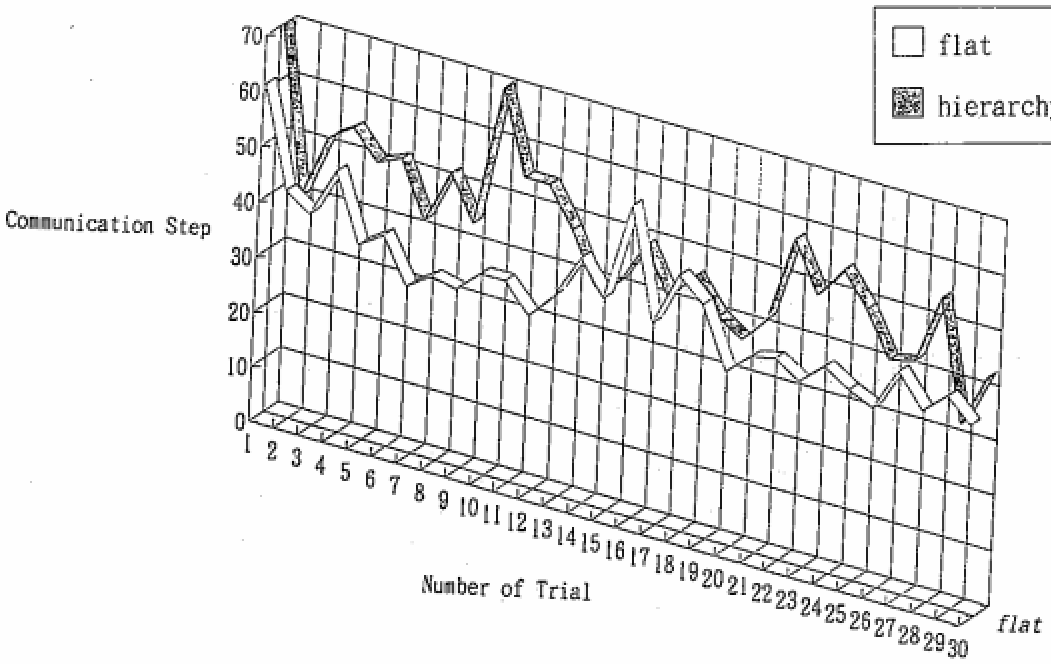

Figure 5: Learning from Communication in LPC Model

The experimental system is partially implemented. The main differences of the architecture in Figure 1 are that (1) the inductive learning component is omitted, and that (2) the reinforcement learning for communication knowledge only gives on-off values for the other agents. We have used 10 agents in the experiments.

When agents are continuously given similar problems, they begin to learn rapidly. Figure 3 shows the results of giving agents a knowledge intensive problem, in which each agent has sufficient number of memories for storing learned concepts. In such a case, the learned concepts have been gathered into specific agents. As a result, shown in the white line in the figure, the total corresponding problem solving steps are reduced remarkably. However, if we limit the number of memories for each agent equal to only 2, then the problem solving steps become larger than the one with sufficient number of memories as is shown in the gray line in the figure.

Next, in order to assess the effects of the communication functions, we have prepared two different organizational structures of agents: the one is hierarchical structure, in which the communication is restricted to the structure shown in Figure 4 (1), the other is flat one, where every agent can communicate with each other in Figure 4 (2).

In this experiment, the agents in the flat structure begin to form self-organized optimal communication paths. Figure 5 shows the number of trials and the number of communications. The white and gray lines in the figure respectively show the results of learning via communication for flat and hierarchical organizations. This result comes from the fact that the flat organization is more flexible fro the given problems than the hierarchical one. This result indicates that the agents' communication knowledge was restructured as learning proceeded in the distributed environment.

5 Related Work on the Learning Model

There are several researches in the literature on multi-agent learning models. The architecture of the proposed model is some extensions of the ideas in practical systems e.g., presented in [Sugawara, 1993], [Vittal 1992]. The learning mechanisms equipped in such systems are considered to be limited ones from the problem solving perspectives. The proposed model has provided a uniform framework for such learning mechanisms. Shaw [Shaw 1989], Weiss [Weiss 1993], and Tan [?] deal with reinforcement learning in multi-agent systems. The problem is how several agents can collectively learn to coordinate their actions such that they solve a given environmental task together. We think that these ideas are effective in learning by communication in multi-agent learning. Although the examples presented here is a very simplified one, we think that it has a potential described in [Bradzil 1991] or [Sian 1991].

The self organizational properties found in [Ishida, *et al.*, 1992] can also be simulated in the proposed model. Furthermore, the problem solving schemas and communication protocols found in [Smith, 1980], [Collin *et al.*, 1991], [Bridgeland *et al.*, 1990], [Mason *et al.*, 1989], [Georgeff, 1983], and [Yokoo *et al.*, 1992] may be introduced in the proposed model.

There are some other researches about the extension of EBL. DeJong [DeJong *et al.*, 1986] argues that the inadequacies of the framework by Mitchell [Mitchell *et al.*, 1986] arise in the treatment of concept operationality, organization of knowledge into schemata, and learning from observation. Yamamura [Yamamura, Kobayashi, 1991] have augmented the framework of EBL on plural examples, and considering relations between generalizations and operationality, and proposed a learning method to generate operational generalizations incrementally. These researches concentrate on the extension of the EBL framework within a *single* knowledge system. On the other hand, the concepts of learning by problem solving is another extension of EBL researches into *multiple* distributed exvironments [Cho 1994].

6 Concluding Remarks

This paper has proposed a machine learning model from problem solving and communication for distributed knowledge system. Our framework is applicable to the next generation information systems, which will involve distributed information agents which work in a synergistic manner (cooperatively) by exchanging information and expertise, coordinating their activities, and negotiating how to solve parts of a common knowledge-intensive problem.

The multi-agent learning model proposed here is also considered to be some extensions of knowledge acquisition tools [Boose *et al.*, 1988] and machine learning techniques [Michalski 1993] for distributed environments. The model can be extended so that the domain knowledge which each agent has (i.e., the capabilities for problem solving) will increase as time passes. Further, it is natural from our prospect that the domain knowledge is partially exchangeable among the agents.

References

[Bond and Gasser, 1988] Bond, A. H., Gasser, L.: An Analysis of Problems and Research in DAI. *Readings in Distributed Artificial Intelligence*, Morgan Kaufmann, pp. 3-36, 1988.

[Boose *et al.*, 1988] Boose, J. H., Gaines, B. R., (eds.): *Knowledge Acquisition Tools for Expert Systems*, Academic Press, 1988.

[Bradzil 1991] Bradzil, P., Muggleton, S.: Learning to Relate Terms in a Multiple Agent Environment. *Proc. European Working Session on Learning '91*, pp. 424-439, 1991.

[Bridgeland *et al.*, 1990] Bridgeland, D. M., Huhns, M. N.: Distributed Truth Maintenance. *AAAI-90*, pp. 72-77, 1990.

[Carley *et al.*, 1993] Carley, K., Park, D., Prietula, M.: Agent Honesty, Cooperation and Benevolence in an Artificial Organization, AI and Theories of Groups and Organizations: Conceptual and Empirical Research. *Proc. AAAI-93 Workshop on AI and Theories of Groups and Organizations* pp. 1–7, 1993.

[CACM 1994] Special Issue: Intelligent Agents. *Communications of the ACM*, Vol. 37, No. 7, 1994.

[Cho 1994] Cho, D., Kusunoki, F., Ono, S., Terano, T.: Developing a Multi-Agent Model for Distributed Knowledge Systems. *Proc. 2nd Int. Conf. on Expert Systems for Development*, pp. 49-54, April, 1994.

[Cutkovsky *et al.*, 1993] PACT: An Experiment in Integrating Concurrent Software Engineering, *IEEE Computer*, Vol. 26, No. 1, pp. 28-37, 1993.

[Collin *et al.*, 1991] Collin, Z., Dechter, R., Kaiz, S.: On the Feasibility of Distributed Constraint Satisfaction. *IJCAI-91,* pp. 318-324, 1991.

[DeJong *et al.*, 1986] De Jong, G. F., Mooney, R. J.: Explanation-Based Generalization: An Alternative View. *Machine Learning* , Vol.1, No.2, pp.145-176, 1986.

[Fox, 1981] Fox, M.: An Organizational View of Distributed Systems. *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 11, No. 1, pp. 70-80, 1981.

[Georgeff, 1983] Georgeff, M. P.: Communication and Interaction in Multi-Agent Planning. *AAAI-83*, pp. 125-129, 1983.

[Ishida, *et al.*, 1992] Ishida, T., Gasser, L., Yokoo, M.: Organization Self-Design of Distributed Production Systems. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 4, No. 2, pp. 123-134, 1992.

[Keller 1988] Keller, R. M.: Defining Operationality for Explanation-Based Learning. *Artificial Intelligence*, Vol. 35, No.2, pp. 227-241, 1988.

[Lee 1993] Lee, K. C., Mansfield Jr. W. H., Sheth, A. P.: A Framework for Controlling Cooperative Agents. *IEEE Computer*, Vol. 26, No.7, pp. 8-16, 1993.

[Lenat *et al.*, 1990] Lenat, D.B., Guha,R.V.: *Building Large Knowledge-Based Systems.* Addison-Wesley, 1990.

[Mason *et al.*, 1989] Mason, C. L., Johnson, R. R.: DATMS: A Framework for Distributed Assumption Based Reasoning. Gasser, L., Huhns, M. N. (eds.): *Distributed Artificial Intelligence*, Vol.II, pp.293 - 318, Morgan Kaufmann, 1989.

[Michalski 1993] Michalski, R.S.: Toward a Unified Theory of Learning: Multi Strategy Task-adaptive Learning. Buchanan, B. G., Wilkins, D.C. (eds.): *Readings in Knowledge Acquisition and Learning*, Morgan-Kaufmann, pp.7-38, 1993.

[Michalski 1994] Michalski, R., Tecuci, G. (eds.): *Machine Learning IV: A Multistrategy Approach.* Morgan-Kaufmann, 1994.

[Minton 1988] Minton, S. N.: *Learning Search Control Knowledge: An Explanation-Based Approach.* Kluwer, 1988.

[Mitchell *et al.*, 1986] Mitchell, T.M., Keller, R. M., Kedar-Cabelli, S.: Explanation-Based Generalization: A Unifying View. *Machine Learning* , Vol.1, No.1, pp.47-80, 1986.

[Neches 1991] Neches, R. et al., Enabling Technology for Knowledge Sharing. *AI Magazine*, Vol. 12, No. 3, pp.36-56, 1991.

[Nishida *et al.*, 1993] Nishida, T., Takeda, H.: Towards the Knowledgeable Community. *Proc. KB&KS'93*, pp.157-166, 1993.

[Papazoglou *et al.*, 1992] Papazoglou, M. K., Laufmann, S. C., Sellis, T. K.: An Organizational Framework for Cooperating Intelligent Information Systems. *International Journal of Intelligent and Cooperative Information Systems*, Vol. 1, No.1, pp.169-202, 1992.

[Shavlik 1990] Shavlik, J.W.: *Extending Explanation-Based Learning by Generalizing the Structure of Explanations.* Morgan-Kaufmann (Research Notes in Artificial Intelligence Series), 1990.

[Shaw 1989] Shaw, M. J., Whinston, A. B.: Learning and Adaptation in Distributed Artificial Intelligence Systems. Gasser, L., Huhns, M. L. (eds.): *Distributed Artificial Intelligence II*, Morgan-Kauffman, pp. 413-430, 1989.

[Sian 1991] Sian, S. S.: Adaptation Based on Cooperative Learing in Multi-Agent Systems. Demazeau, Y., Müller, J.-P. (eds.): *Decentralized Artificial Intelligence 2*, pp. 257-272, 1991.

[Smith, 1980] Smith, R. G.: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Trans. on Computers*, Vol. 29, No. 12, pp. 1104-1113, 1980.

[Sugawara, 1993] Sugawara, T., Lesser, V.: On-Line Learning of Coordination Plans. *Proc. 12th Int. Workshop on Distributed AI*, 1993.
[Tan 1993] Tan Tan, M.: Multi-Agent Reinforcement Learning: Independent vs. Cooporative Agents. *Proc. 10th Machine Learning Workshop*, pp. 330-337, 1993.

[Terano 1993] Terano, T. : Requirements of AI Models Applicable to Organizational Learning Theory and Two Related Examples. *Proc. AAAI-93 Workshop on AI and Theories of Groups and Organizations*, pp. 92-95, 1993.

[Yamamura, Kobayashi, 1991] Yamamura, M., Kobayashi, S.: An Augmented EBL and its Application to the Utility Problem. *12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, pp. 623-629, 1991.

[Yokoo *et al.*, 1992] Yokoo, M., Durfee, E. H., Ishida, T., Kuwabara, K.: Distributed Constraint Satisfaction for Formalizing Distributed Problem Solving. *12th International Conference on Distributed Computing Systems (ICDCS-92)*, pp. 614-621, 1992.

[Vittal 1992] Vittal, J., Silver, B., Frawley, W., Iba, G., Fawcett, T., Dusseault, S., Doleac, J.: Intelligent and Cooperative Information Systems Meet Machine Learning. *Int. J. of Intelligent and Cooperative Information Systems*, Vol. 1, No. 2, pp. 347-361, 1992.

[Weiss 1993] Weiss, G.: Learning to Coordinate Actions in Multi-Agent Systems, *13th International Joint Conference on Artificial Intelligence, (IJCAI-93)*,, pp. 311–316, 1993.