The Difficulty of Coping With Tradeoffs through Local Decision Making

Munehiro Nakazato*
Kei Matsubayashi†
Mario Tokoro‡
Department of Computer Science, Keio University.
3-14-1 Hiyoshi, Yokohama 223 Japan
E-mail: {nakazato,kei,mario}@mt.cs.keio.ac.jp§

Abstract

In Distributed Problem Solving (DPS), *Dynamic Reorganization* is said to be effective in order to cope with tradeoffs between the global accuracy and the efficiency of the problem solving. In order to realize an architecture for the Dynamic Reorganization, it is important to consider a question "*When should agents reorganize ?*". Such architectures, where each agent has a Meta-level component and makes decisions locally to improve the efficiency, seem to be robust systems. There is, however, the possibility that such an architecture decreases its global accuracy because each agent pursues its own efficiency to the extreme.

In this paper, we present several experiments on the pursuit game to discuss the difficulty of implementing an architecture which aims to improve the efficiency of the problem solving through the local decision making. We show the difficulty of controlling "When to reorganize" in such an system is a rather subtle problem and the "making plans of agents consistent" is difficult. Moreover, we show unsuitable "translation" of the global accuracy and efficiency into the individual agents' utility could arrow selfish agents to *exploit* cooperative agents and *co-evolve* with other selfish agents.

1 Introduction

In Distributed Problem Solving (DPS) domain, tradeoffs between the global accuracy and the efficiency of problem solving is an inevitable problem. **Dynamic Reorganization**, *i.e.* changing the organizations dynamically according to changes in the problem space,

*nakazato@mt.cs.keio.ac.jp
†kei@mt.cs.keio.ac.jp
‡mario@mt.cs.keio.ac.jp Also affiliated with Sony Computer Science Laboratory Inc.
§The registered staff of the original and Advanced Information Agency (IPA), Japan.

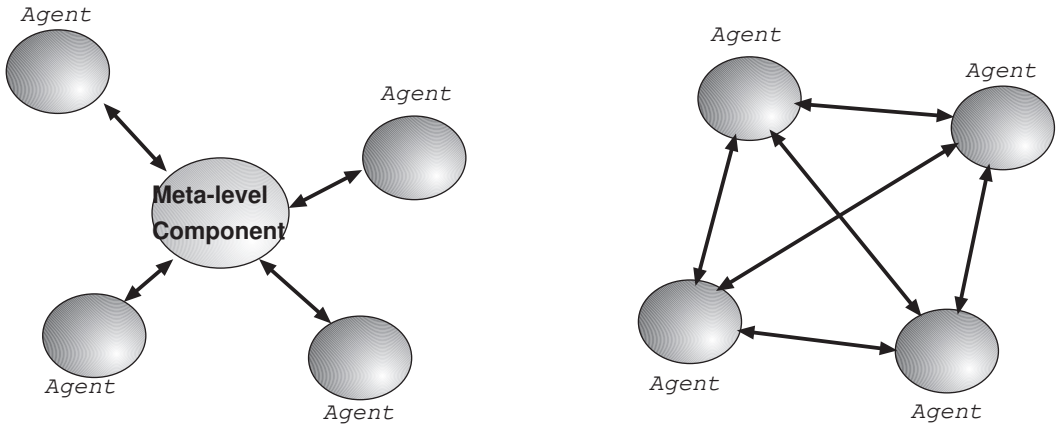
is said to be hopeful solution to cope with these tradeoffs, so that it is discussed by many researchers [2, 6, 7, 14].

Durfee *et al.*[1, 2] proposed a framework called *Partial Global Plan* (PGP) and illustrated some empirical results in the Distributed Vehicle Monitoring Testbed (DVMT). PGP reallocates the tasks to the agents dynamically. In their work, their attention was paid on balancing predictability and responsiveness. Ishida *et al.*[6, 7] proposed an another framework of reorganization. They adopt two reorganization primitives called *Decomposition* and *Composition* for the load balancing of *distributed production-system-based* agent production system. Although each approach of Dynamic Reorganization work well, an unexpected inefficiency could occur unless the reorganization works timely. In this paper, we show an empirical result that untimely reorganization leads to less efficient problem solving in the Pursuit Game domain.

When dealing with Dynamic Reorganizations, "*How to construct Meta-level Architectures*" becomes a significant problem. Meta-level components undertake to decide when and how to reorganize the systems. We can classify Meta-level Architectures into two types as follows (Figure1).

Centralized Meta-level Reasoning System has only one meta-level component and it makes all the decision about reorganization, that is, the meta-reasoning is done in the *Global Decision Making*.

Distributed Meta-level Reasoning Each agent has its own meta-level component, and executes meta-level control (action sequences) individually. The meta-reasonings are thus done in the *Local Decision Making*.



(a) Centralized meta-level reasoning (b) Distributed meta-level reasoning

Figure 1: The two types of Meta-level architecture

In [2], similar concepts were introduced as *centralized meta-level organizations* and *broadcast meta-level organizations* which correspond to Centralized meta-level reasoning and Distributed meta-level reasonings respectively. In the case of *Centralized Meta-level Reasoning*, the meta-level control component may be embedded in one of the group of agents. In centralized meta-level organizations of PGP, the Meta-level control was done by a single agent with the least load. In the Pursuit Game domain, Stephens *et al.*[16]

also introduced *Controlling-Agent* organization allowing one supervisory agent to control all other agents.

Distributed meta-level reasoning seems to act more robustly than Centralized meta-level reasoning because it seems that if one of agents were in trouble, other agents would compensate for it. However, trading off the accuracy of problem solving against the efficiency with Distributed meta control architecture has several difficulties. First, because of the partiality of information which each agent has, a subtle difference of the viewpoints makes it difficult to make the plans of all agents consistent, so that inconsistent plans could be the cause of the decrease in the global accuracy. In this paper, we also present an experimental result that inconsistent plans sharply decrease the success ratio in capturing the prey. Second, in Distributed meta-level architectures, each individual meta-level component aims at improving the efficiency from its own viewpoint. For each agent, improving the efficiency means to reduce its own costs. Therefore, agents may tend to avoid cooperation with other agents. Such selfish agents then come to exploit cooperative agents. Therefore the number of cooperative agents decreases. However, cooperation between the agents is necessary to maintaining the global accuracy. Therefore, the global accuracy badly falls.

We present a simulation of a population of individuals trying to tradeoff the efficiency and the global accuracy through the local decision making in the Pursuit Game and compare them with those of through the centralized decision making. Thorough these simulations, we show how difficult it is to maintain the global accuracy with a Distributed Meta-reasoning architecture.

The outline of the rest of the paper is as follows. In the next section, we show previous work on pursuit game and our experimental frameworks. In Section 3, three different experiments about the difficulty of Distributed meta-reasoning are presented. The last section 4, we conclude this paper with our future work.

2 Background — Pursuit Game

We present some experiments about Distributed meta-reasonings on the pursuit game. In this section, we briefly review some previous works and show our experimental framework.

2.1 Previous Work

The original version of Pursuit Game was introduced by Benda *et al.* The Game consists of one escaping agent (Red agent, prey) and four pursuing agents (Blue agents, predators) in a two-dimensional infinite grid. The purpose of the Blue agents is to capture the Red agent by occupying the four *capture positions* surrounding Prey's current location (Figure 2).

Gasser *et al.*[3] proposed a coordination framework for representing and revising organizational knowledge. They regarded an organization as a set of settled and unsettled questions about belief and action. The reorganization is made by the agents settling some questions about the belief and actions of other agents. They also mentioned that the agents can capture the prey if they approach it keeping all four quadrant around the Prey occupied (*Lieb configuration*).

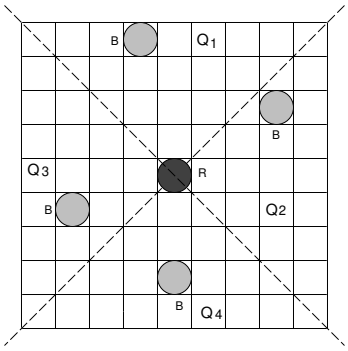


Figure 2: Pursuit game and Lieb configuration [3]

Stephens and Merx [16, 15] proposed and compared four different schemes for controlling Blue agents. They also introduced six different initial grid configurations (Scenario 1–6) which was characteristic each(Figure 3, 4). In their investigation, they concluded a centralized strategy in which one supervisory agent controls the other 3 agents (Controlling-Agent) is the most successful in capturing Red, but in terms of heuristic efficiency [16], a strategy in which agents share common knowledge and plans (Negotiating-Agent) is better than the centralized organization.

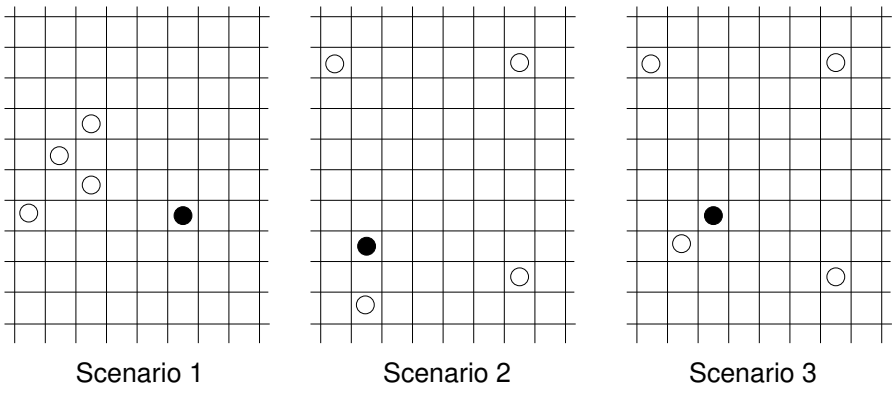


Figure 3: The scenarios of the Pursuit Game (1)

Based on the Stephens' conclusion, Osawa [14] gave an experiment about adaptive strategies. The result shows that an adaptive organizations, where each agent searches for the Prey autonomously with a small amount of communication (Autonomous-agent) at an earlier phase, then adopts a centralized organization (Controlling-agent), could capture the Prey more efficiently in terms of the heuristic costs and communication costs. He also proposed a meta-level coordination strategy to implement this kind of dynamic reorganization and calculated the maximum time cost incurred by the static strategies and the dynamic strategy in Pursuit Game domain.

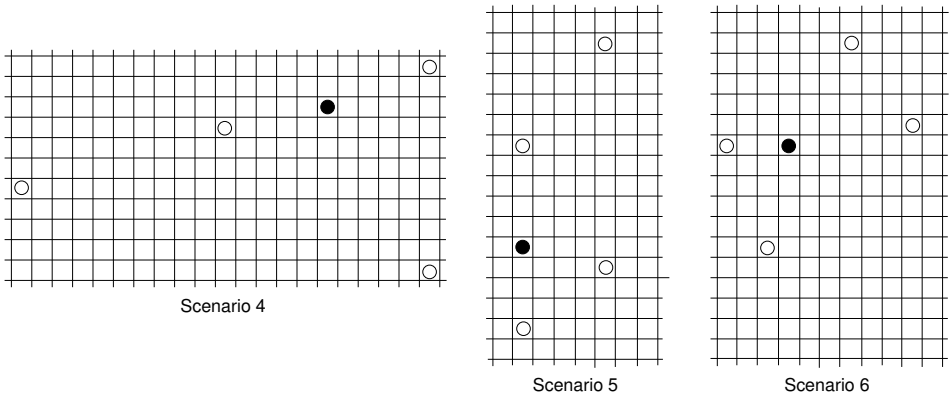


Figure 4: The scenarios of the Pursuit Game (2)

2.2 An Experimental Framework

Most of previous works mentioned above considered the situation where the Red agent moves randomly. This means that the Red agent stays almost at the same position. Manela [11] illustrated the pursuit game can be solved easily by introducing the concept of *boredom*. Furthermore, Korf [8] proposed a simple solution to capturing the Prey. Therefore, in order to make the pursuit game an appropriate testbed, we allow the Red agent to move strategically by moving to a neighboring cell that is the furthest away from the nearest Blue agent and limit the speed of the Red to 80 percent of that of the Blue agents[1].

For the actions of Blue agents, we allowed two types of strategy.

Self-centered strategy: An agent just heads for the nearest capture position. This corresponds to Communicating-Agent in [16].

Cooperative strategy: An agent tries to cooperate with other cooperative agents. First, the agent strives to form the Lieb configuration with other Cooperative agent[2]. Then if all other cooperative agents occupy their quadrant each, it approaches the Prey keeping itself in its quadrant. If the capture position which corresponds to its quadrant is already occupied by other agent, it selects other quadrant and takes a roundabout way. This action is quite similar to Controlling-Agent in [16], though there is no controlling agent. If there is no cooperative agent except himself, this agent just approaches the Prey.

These strategies contrast with each other. Self-centered strategy can reduce moving costs to approach the prey, however, it cannot guarantee capturing the prey. Cooperative strategy needs more moving costs because it forms Lieb configuration first, however, it can always capture the prey.

[1] If the prey moves as fast as the predators, it is impossible for the predator to capture the prey. Therefore, we need to restrict the speed of the prey. From our preparatory experiment, we concluded 80 percent of the predators' speed is proper for our experiment

[2] By broadcasting messages, the Cooperative agents can recognize the existence of each other.

Each agent can select either one of those strategies and change from one to another at any time. After occupying a capture position, the Blue just follows the Prey without any strategies. The point of our implementation is that Blue agents can employ action sequences different from one another in one problem solving process. Some agent may act always cooperatively, while others may change their action from the Self-centered strategy to the Cooperative one.

3 Experimental Evaluation

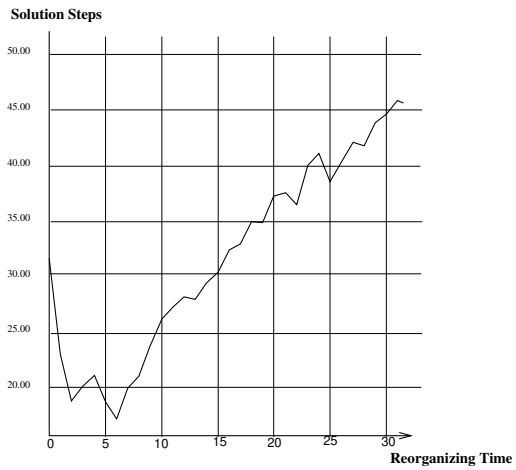
In this section, we presents three experiments concerning the difficulty of Distributed meta-reasonings. First, we show a little difference of time to reorganize leads a significant fall in efficiency. In addition, we show inconsistency of plans of agents make it difficult for the predators to capture the prey. Finally, in order to compare the behavior of distributed meta-reasoning agents with that of centralized meta-reasoning agents, we evolve population of Blue agents.

3.1 When should Agents Reorganize ?

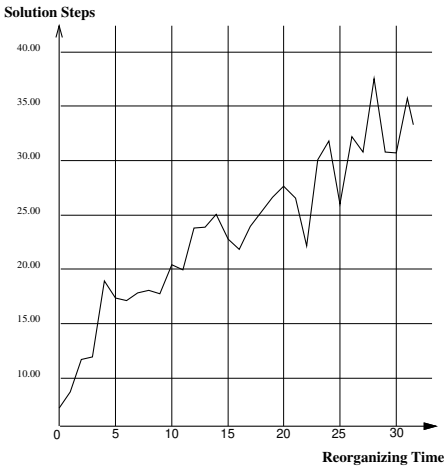
On the one hand, the Self-centered agents can smoothly get close to the Prey in an earlier phase of pursuing. However, they easily fail to capture the Prey in the final phase. On the other hand, the Cooperative agents take time to get close to the prey in earlier phase since they try to make the *Lieb configuration*. But they can finally surround the four direction of the Prey. That is, in Pursuit Game, there is a tradeoff between the accuracy of the global solutions and the efficiency of problem solving. Osawa [14] demonstrated that the adaptive strategy that adopts Communicating Organization (Self-centered action) in earlier phase and then changes its organization to Centralized Organization (Cooperative action) can capture the Prey more efficiently than the static strategies, *i.e.* with fewer moves.

Here a question, "*When should the agents change their organization?*" arises. In order to investigate the question, we do some experiments using the 6 scenarios introduced by Stephens et al[16] as the initial configurations of the grid. Figure 5 shows the cases of scenario 1 and 6. *Reorganizing time* is time (steps) when agents change their action from Self-centered to Cooperative (SC agent). "$Reorganizing\ time = 0$" means that the agents take only the Cooperative action (All-C agent). Solution Steps means the number of moves required by agents to capture the Prey.

In our experiment, it turned out that the most efficient time to reorganize depends on the scenario, and the adaptive strategies (SC agent) need more moves than All-C in Scenario 4, 6. This is because it becomes difficult to form the Lieb configuration after coming too close to the Prey. In addition, in the case of Scenario 6, since the Lieb configuration is already formed at the beginning of the game, if agents act in self-centered way, this configuration is broken due to the Prey's strategic action though All-C agents can converge to the Prey keeping this configuration. Therefore, the Blue agents have to decide the reorganizing time appropriately based on the configuration of the environment around them.



(a) Scenario 1



(b) Scenario 6

Figure 5: Reorganizing Time and Moves to capture the Prey

3.2 Consistency between Local Meta-reasonings

In the case of distributed meta-reasoning, as mentioned above, subtle difference of the environment may cause considerable inefficiency of the problem solving. Therefore, the little difference of the viewpoints can make the plans of agents inconsistent. Moreover, a failure of some agents may occur. What would happen if the meta plans of agents become inconsistent? We thus investigated how the success ratio in capturing the Prey decreases when some self-centered agents are mixed in a group of Blue agent. Table 1 shows the capturing ratio of the group of SC agents and All-C agents against the number of agents, which always takes Self-centered action (All-S agents), mixed in the group. SC agents change their action in the most efficient timing for each Scenario as proved in the previous section. In the case of Scenario 4 and 6, only All-C agents are shown because they are the most efficient organization.

Though the SC agent is a very efficient strategy, it not so robust. In each Scenario, as the number of All-S agent increases, the SC agents become inaccurate for solving the Game, *i.e.* they cannot capture the prey, rather than the All-C agents. This is because the SC agents may occupy the capture position faster than the All-S agents: If a SC agent occupies one of capture position before changing its action to the Cooperative, All-S might not be able to capture another position since the All-S agents cannot go roundabout behind the Prey. Especially, in the case of Scenario 1 and 5, SC agents cannot capture the Prey even in the half of the trial against one All-S agent, because in these scenarios, it is difficult for Blue agents to make the Lieb configuration since agents' center of gravity is in one side of the Prey at the initial configuration.

Scenario		1		2		3		4	5		6
Agent Type		*All-C*	*SC*	*All-C*	*SC*	*All-C*	*SC*	*All-C*	*All-C*	*SC*	*All-C*
Number of All-S	0	100	100	100	100	100	100	100	100	100	100
	1	100	36.4	100	83.4	100	97.3	94.5	100	42.3	100
	2	69.3	3.3	74.3	43.5	58.1	64.7	36.1	45.4	16.5	73.2

Table 1: Success ratio against the number of Always Self-centered agent (%)

3.3 A Genetic Search for Accurate and Efficient Meta Strategies

In the previous section, it turned out that the reorganizing strategies are not always robust, *i.e.* they could easily lose the accuracy against the inconsistency of plans, although they are able to reduce the costs for solving the problem. Therefore it is important to balance the efficiency of the problem solving and the robustness against the inconsistency of plans.

To see how well a group of the agents that make local or global decision can tradeoff the efficiency of problem solving against the global accuracy, we evolve the population of agents playing the Pursuit Game. We experiment this using the technics of *Genetic Programming* (GP): an advanced version of *Genetic Algorithms* (GAs) introduced by Koza [9, 10]. While each individual is described with bit sequences in GAs, the individuals of GP consist of S-expressions of LISP. We translate the strategies of Blue agents in chromosomes of GP. As mentioned above, the Blue agents can take arbitrary action sequence like SC, All-C, etc. In order to make this simulation feasible, we restrict the action sequences of the Blue agents up to 3 actions like "SCS" which means this type of agent takes the Self-centered action first, then changes to the Cooperative action, finally changes to the Cooperative action. As thresholds to change the actions, Blue agents can take three types of parameter: (a) *the Simulation steps*, (b) *the Number of Blue occupying the capture positions*, and (c) *Convergence ratio* estimated by the sum of distance between the Red and each Blue agent. Besides, the better genes are selected to the next generation using *tournament selection*[3].

Each game consists of 100 simulation steps. However the game ends when the Red is captured. A Blue agent (whose strategy is described in a gene of GP) plays one game with three other agents chosen randomly from the population and is paid utilities according to the pursuit outcome. This process is repeated N times for each Blue agent and the *fitness* of each is the sum of those utilities. The fitness of Blue agent i is thus defined as

$$f(i) = \sum_{j=1}^{N} u(i,j)$$

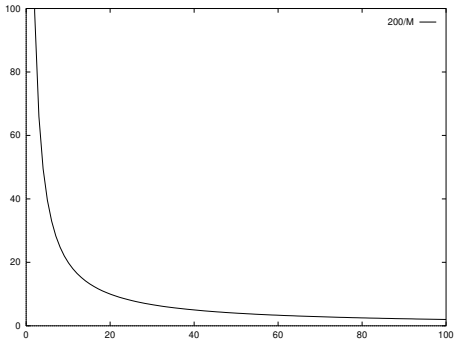
[3]Tournament selection randomly selects 5 genetic programs from the population. The fitness of each member of the group is compared and the actual best replaces the worst.

where $u(i,j)$ is the utility of Blue agent i in the j th game. As utility, we used two types of function below.

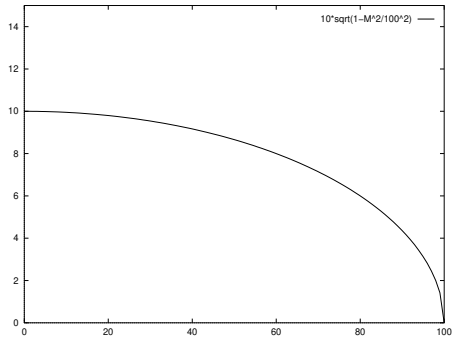
$$(1) \qquad u(i,j) = \frac{200}{M_i}$$

$$(2) \qquad u(i,j) = 10\sqrt{1 - \frac{M_i^2}{100^2}}$$

where M_i is the number of moves acquired by Blue agent i to occupy one of four capture positions (not to capture the Prey). In each functions, if the group of the agents cannot capture the Prey in 100 simulation steps, no one is paid the utility. This means that the faster an agent reaches a capture position, the higher utility it can get, but if it and its fellow fail to capture the Prey, no one get utility. Figure 6 shows both of utility functions. Utility function (1) draws hyperbolic line, and (2) draws oval.



(1) Utility function 1. (hyperbolic)



(2) Utility function 2. (oval)

Figure 6: Utility functions for Blue agents

Figure 7 (a) shows the average accuracy (percentage of success in capturing the Prey) agents making decisions locally (Distributed meta-reasoning). As a initial configuration of grid, we used Scenario 1 in Figure 3[4]. For comparison, we present the case of population which make global decision(Centralized meta-reasoning) in Figure 7 (b).

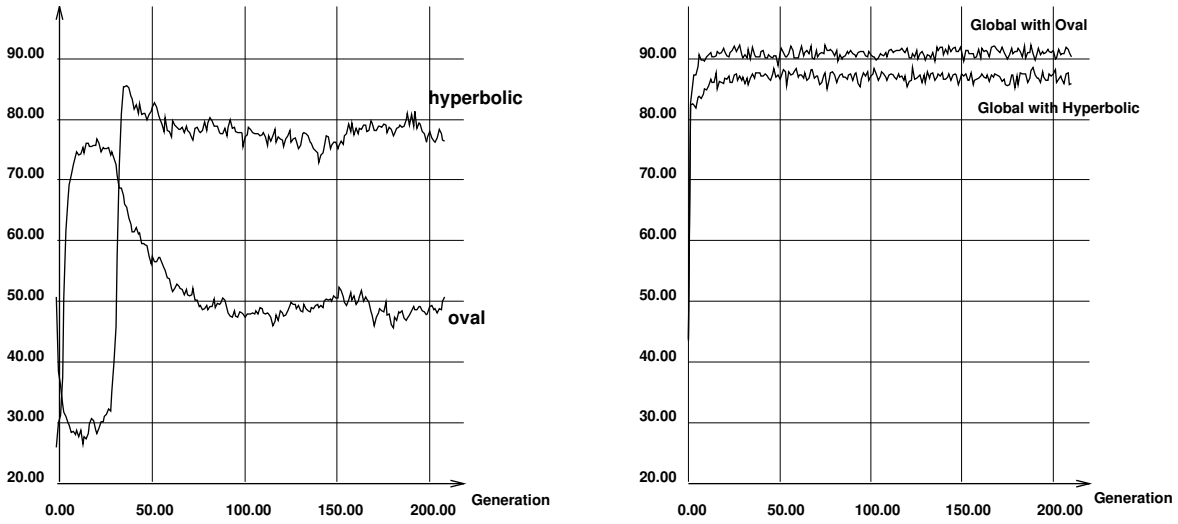
In the case of the Local decision making, the population of agents using hyperbolic function can achieve about 80 percent of accuracy. However, average capturing ratio of agents using oval function sharply declines while it achieves 75 percent in earlier generations. In case of the Global decision making, both functions lead agents to higher accuracy. In any case, average fitness increased and average steps acquired for capturing the Prey decreased.

We can see the reason of this in Figure 8(a),(b) which show dynamics of population of agents making distributed and centralized decision respectively with oval function. In the

[4]Scenario 1 is the most difficult of 6 scenarios. In this scenario, it takes time for Blue agents to capture the Prey.

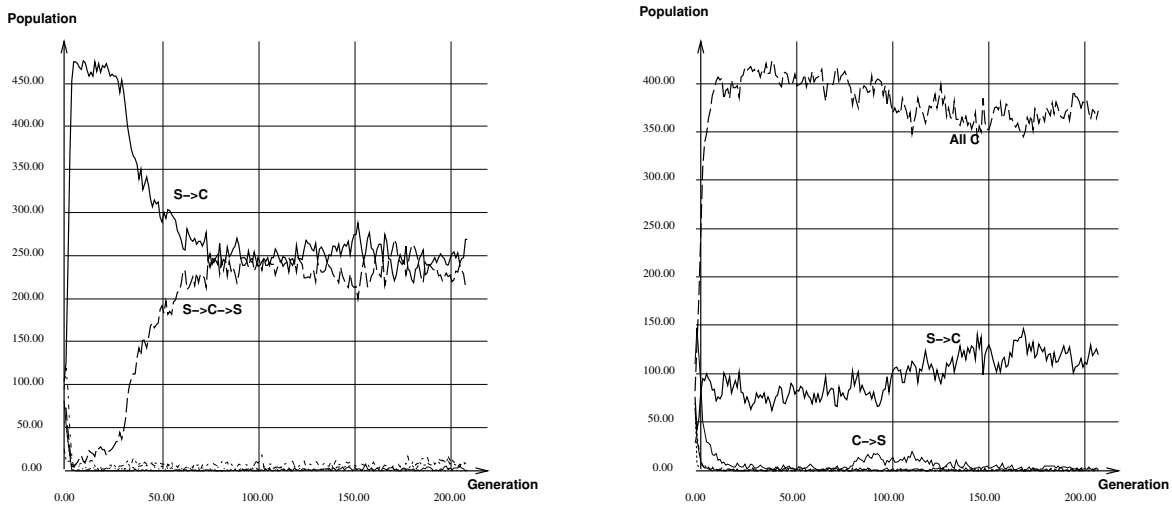
case of Centralized agents (Figure 8 (b)), All-C agents dominate the population in earlier generation and the rest of Blue agents are AC agents, *i.e.* most agents in the population are cooperative. Therefore, high accuracy are retained. In the case of Distributed agents (Figure 8 (a)), most agents take SC action in earlier generation, and the capturing ratio shows high value. However, Selfish agents (SCS agents) exploit cooperative agents and increase their number in the population and co-evolve with SC agents. Average accuracy thus sharply falls.

The reason why the hyperbolic function results in better accuracy than oval function is that in case of hyperbolic function, difference of utility between success and failure is very large, while penalty for failure is relatively small in oval function.



(a) Local decision making (%) (b) Global decision making (%)

Figure 7: Average Success Ratio in Capturing the Prey



(a) Local decision making (%) (b) Global decision making (%)

Figure 8: Population Dynamics of agents using oval utility function

4 Conclusion and Future Work

We have presented some experiments to discuss problems of implementing a Distributed meta-reasoning architecture: First, it is important to decide "*time to reorganize*" correctly because improper reorganization could lead to unexpected inefficiency. Of course, this is also the case with the Centralized meta-reasoning. Second, "*make plans consistent with each other*" is difficult because the partiality of information, communication delay, or failures of some agents may cause serious difference between the viewpoints of agents. Inconsistent meta-level controls could make the system less accurate.

Finally, it is difficult to associate the local effect with the global efficiency and accuracy. As shown in Section 3.3, the behavior of the population of agents which make their plan locally depends on which Utility function the agents use, *i.e.* "*how to translate the global efficiency and accuracy into the utility of agent*". An inadequate utility function would provoke self-centered agents to exploit cooperative agents. Same kind of problems were introduced by Hogg *et al.* as *Social Dilemmas* between the global accuracy and the local efficiency [4, 5].

In spite of these difficulty, today a lot of computational resources are distributed in the world-wide networks and there are more needs of autonomous systems. It is no longer feasible to cope with such an situation using centralized architectures. We have to study a new framework for Open Systems. To do so, those problems mentioned above have to be addressed properly in more practical domain. In such domain, we will have to include more factors, for example, of communication costs and delay in utility of the agents. In particular, implementation of architectures in which agents estimate the environment correctly and translate into their utility is important problem.

Acknowledgment

This project is funded by the Advanced Information Technology Program (AITP) of Information-technology Promotion Agency (IPA), Japan.

References

[1] E. H. Durfee and V. R. Lesser. Using partial global plans to coordinate distributed problem solvers. In *Proceedings of IJCAI-87*, pages 875–883, 1987.

[2] E. H. Durfee and V. R. Lesser. Predictabilty versus responsiveness: Coordinating problem solvers in dynamic domains. In *Proceedings of AAAI-88*, pages 66–71, 1988.

[3] L. Gasser, R. W. Hill N. Rouquette, and J. Lieb. Representing and Using Organizational Knowledge in Distributed AI Systems. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, pages 55–78. Morgan Kaufmann Publishers, Inc., 1989.

[4] N.S. Glance and T. Hogg. Dilemmas in computational societies. In *Proceedings of ICMAS-95*, pages 117–124, 1995.

[5] T. Hogg. Social Dilemmas in Computational Ecosystems. In *Proceedings of IJCAI-95*, pages 711–716, 1995.

[6] T. Ishida, L. Gaser, and M. Yokoo. Organization Self-Design of Distributed Production System-Based Agents. In *Proceedings of AAAI-90*, pages 52–58, 1990.

[7] T. Ishida and L. Gasser. A Dynamic Organizational Architechture for Adaptive Problem Solving. In *Proceedings of AAAI-92*, pages 185–190, 1992.

[8] R. E. Korf. A simple solution to pursuit games. In *Proceedings of the Eleventh Workshop on Distributed Artificial Inteligence*, pages 183–194, 1992.

[9] J. R. Koza. *Genetic programming.* MIT Press, 1992.

[10] J. R. Koza. *Gentic programming II.* MIT Press, 1994.

[11] Mauro Manela and J. A. Campbell. Designing good pursuit problems as testbeds for distributed AI: a novel application of genetic algorithms. In *Proceedings of MAA-MAW93*, pages 1–21, 1993.

[12] K. Matsubayashi and M. Tokoro. A Collaboration Mechanism on Positive Interactions in Multi-agent Environments. In *Proceedings of IJCAI-93*, 1993.

[13] K. Matsubayashi and M. Tokoro. A Collaboration Strategy for Repetitive Encounters. In *Proceedings of MAAMAW-94*, 1994.

[14] E. Osawa. A metalevel coordination strategy for reactive cooperative planning. In *Proceedings of ICMAS-95*, pages 297–303, 1995.

[15] L. M. Stephens. The Effect of Agent Control Strategy on the Performace of a DAI Pursuit Problem. In *Proceedings of the 10th Workshop on Distributed Artificial Intelligence*, 1990.

[16] L. M. Stephens and M. Merx. Agent Organization as an Effector of DAI System Performance. In *Proceedings of the Ninth Workshop on Distributed Artificial Inteligence*, pages 263–292, 1989.