PG Lab ディスカッションペーパーシリーズ
第20号　　2007年10月

Discussion Paper Series
No.20　October, 2007

ISSN　1880-9146

Performance Improvement of GridRPC-based Multi-Agent Simulation Software

Hiroshi ARIKAWA, Sen-ichi MORISHITA, Tadahiko MURATA

Performance Improvement of GridRPC-based Multi-Agent Simulation Software

Hiroshi ARIKAWA[1], Sen-ichi MORISHITA[2], and Tadahiko MURATA[1, 2]

Abstract

This paper describes a method to improve the performance of GridRPC-based Multi-Agent Simulation (MAS) software. To show realistic simulation results to policy makers in governments and local communities, a large number of agents should be implemented in the software. So far we have been developing a large-scale MAS software using two traditional programming model, Message Passing Interface (MPI) and Grid Remote Procedure Call (GridRPC), for policy making assistance on parallel and distributed computing system such as cluster or computing grid system. In our previous work, we have seen that it is difficult to reduce the computation time using the GridRPC-based MAS application. In this paper, we propose a new programming technique for MAS to reduce the computation time of GridRPC-based MAS application. Our proposed technique is to combine GridRPC approach with MPI approach. We firstly explain the programming technique for combining GridRPC approach with MPI approach. Then we evaluate MAS software with our proposed technique. We show that the performance of GridRPC-based MAS software can be improved.

Keyword: Large Scale Multi-Agent Simulation, Remote Procedure Call for Grid (GridRPC), Combined Approach, Message-Passing Interface (MPI), Performance Evaluation

1 Policy Grid Computing Laboratory, Kansai University

2 Department of Informatics, Kansai University

1. Introduction

Multi-Agent Simulation (MAS) is one of many promising research field in artificial intelligence. Since 1990s, computer simulations in the social sciences have attracted many social scientists [1]. Social science includes various research fields such as sociology, economics, social psychology, organization theory, political science, demography, anthropology and archaeology. One of reasons why MAS is employed for social simulation is its ability to imitate social system behaviors [2-5]. Consider economics for example. The advantage of MAS for economics is that MAS can help to investigate an economic model with social interaction. Manski [6] investigates the influence of social interactions in economics and explains three social interactions: 1) constraint interactions, 2) expectation interactions, and 3) preference interactions. In addition, Manski suggests that an economic analysis should consider the social interactions.

Our goal is to assist social scientists to improve their model for policy making. To improve their simulation model with considering each subject, they need huge computing power to implement their models. Therefore we have to implement the MAS software on high performance computing system. Recently, several studies have been made on how to apply the MAS software for high performance computing system [7-8,12-14]. Timm & Pawlaszczyk [12] show the speedup ratio using their grid architecture. But their applications do not have a communication between agents a different computing resource. Lee *et al.*[13] proposed a way to decentralize MAS to share the information among multiple agents on the computing grid system. However, they did not show any experimental results in their paper. Takahashi & Mizuta [14] show how to reduce the amount of transmission among multiple nodes in supercomputer. They simply proposed to make groups of agents who communicate heavily each other in order to reduce the transmission among group. They show their expectation to reduce the computation time using multiple nodes, but did not show it using real computational resources and how to implement MAS program in supercomputer.

We have been designing and implementing various MAS software for policy making assistance on computing grid system [7-8]. Arikawa & Murata [7] employed computing grid systems to examine the huge number of parameters in an application of day care center allocation problem. Murata *et al.*[8] found the effective condition of the large-scale MAS in homogeneous cluster computer, and they proposed how to parallelize MAS softwares using message passing approach and remote procedure call approach. They confirmed that the computation time of MAS software with message passing approach can be effectively improved on homogeneous cluster computer. However, they did not show how to reduce the computation time of MAS software with remote procedure call approach.

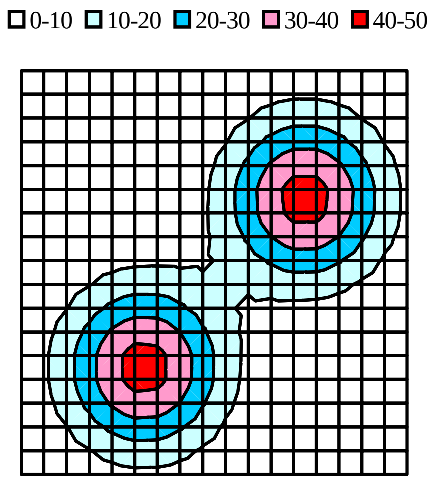

Figure 1. An Example of Sugarscape Model.

We propose a parallelization approach to reduce the computation time of MAS software on computing grid system. Our proposed approach combines remote procedure call approach with message passing approach, and it can achieve the parallelization of MAS software as much as possible. In this paper, we describe a performance improving approach of MAS software on computing grid system. Then, we evaluate the performance of MAS software with our proposed approach, and we show its effectiveness using experiments with our cluster.

2. Implementation of a Large-Scale MAS Software on Computing Grid System

2.1 Target Simulation

We employ Sugarscape model as an environment in MAS [9]. Sugarscape model has multiple agents in the environment where sugar-like resources are available for the agents. Agents should take resources as a food or living expense. When their savings become enough, they can take some actions according to the amount of savings. Fig. 1 shows an example of distributions of resources for agents. Agents try to find resources, take some actions in order to benefit from the environment. In this environment, each agent tries to take resources as many as possible with conflicting with other agents.

In this paper, we apply a MAS model proposed by Nisizaki *et al.*[10]. They examined three financing methods to preserve the global commons in the simulated community. They introduced several rules between agents and an environment to implement the environmental maintenance problem. They defined some pollutants or toxic substances that are produced by agents according to their activity. Since the resource recovery rate in the environment decreases with the amount of the pollutants, the community should have some measures against the pollutants. In order to establish some measures, the community should prepare some financing plan. As for the financing methods, they proposed three ways: voluntary contribution, lottery, and no measures. Their MAS-based approach showed that the financing methods by the lottery may have the positive effect on the transition of the population in the community.

The procedure of each agent in their MAS-based approach is as follows:

Step 1: Each agent decides to move to a neighboring cell according to the information within its eyeshot.

Step 2: According to the movements of all agents, the avoidance of conflict between agents should be made in order not to exist several agents in a single cell.

Step 3: Each agent computes the utility function when the agent makes contribution or purchases lottery tickets. Then the probability of making contribution or purchasing lottery tickets is defined according to the calculated utility.

Step 4: All the contribution or a part of profit from the lottery system are spent on the cost of preserving the global commons. Then resources are restored in the environment.

Step 5: In the contribution system, according to the effectiveness of the financing system to preserve the global commons, the consciousness of each agent to make contributions changes. In the lottery system, the probability for each agent to joint the system depends on win or loss in the lottery.

Step 6: Back to Step 1 until the pre-specified condition is satisfied.

2.2 Parallelization Using GridRPC

Many parallelized software are implemented generally using Message Passing Interface (MPI) or Grid Remote Procedure Call (GridRPC). We have already proposed how to implement the parallelization technique for MAS-based social simulation software using MPI approach and GridRPC approach in [8]. We have shown the effectiveness of those approaches on parallel or distributed computing system. In this paper, we show how to improve GridRPC-based MAS software.

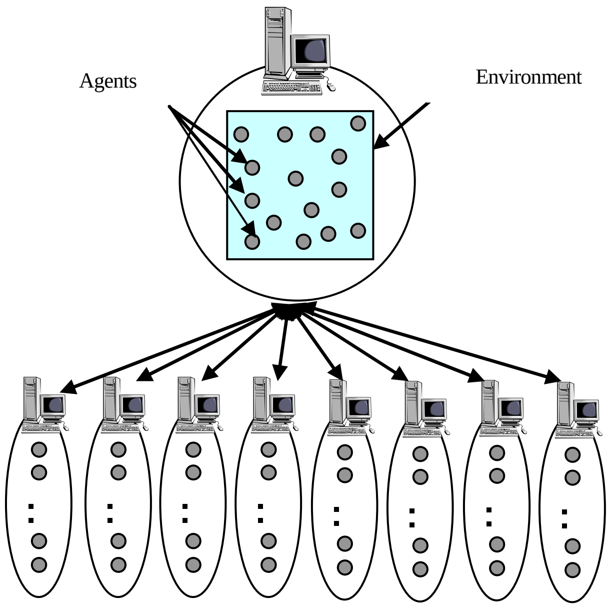

Figure 2. MAS programming model by GridRPC

GridRPC [11] is a programming model based on a remote procedure call mechanism modified for the computing grid system. Subprograms are prepared in remote computing resources, and the main program is invoked subprogram on remote computing resources. That is, the feature of programming model by GridRPC is a client-server model. Programmers can easily implement a task parallel program for computing grid system by using GridRPC library.

Fig. 2 shows an example structure of computing resources using GridRPC approach. Their model is suitable for the development of the software on computing grid system.

The MAS software based on the Sugarscape model consists generally of the decision making procedure and environment maintenance procedure. The decision making procedure (Step 1) is to execute that each agent decides to move to a neighboring cell. On the other hand, the environment maintenance procedure in Subsection 2.1 is to execute to modify environment information. To design a GridRPC-based MAS software, decision making procedure is enable to distribute in assigned computing resources. In the GridRPC approach, we divide the algorithm of MAS in Subsection 2.1 as follows (See Fig. 3):

(1) Create the information for the environment.
(2) Create the initial information for agents.
(3) Send the information of agents to each assigned processor from the master processor (e.g., in Fig. 3) using an asynchronous remote procedure call module (e.g. , grpc_call_async()).

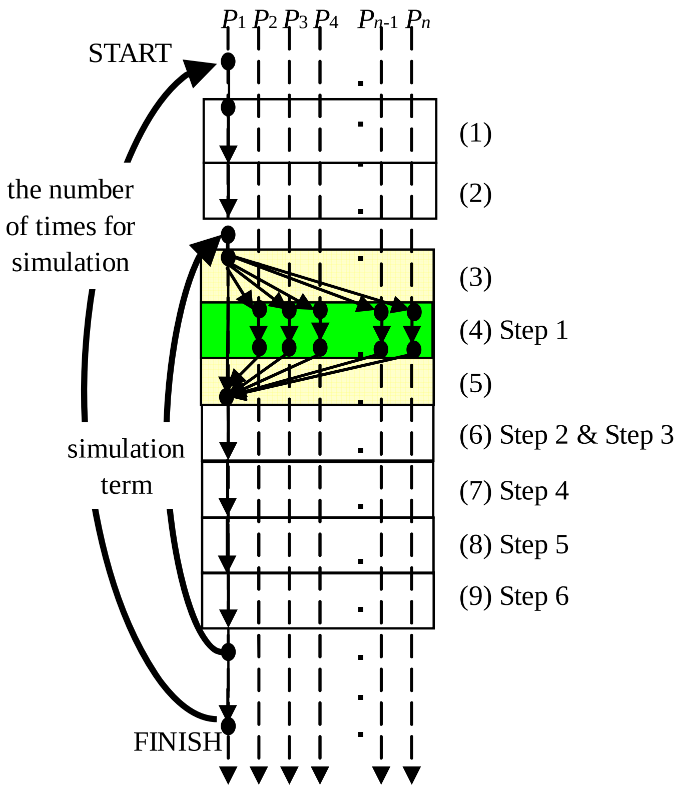

Figure 3. Simulation flow of MAS with GridRPC approach.

(4) **Step 1:** Each agent decides to move to a neighboring cell according to the information within its eyeshot.

(5) Collect the decisions of agents to the master processor using a wait module (e.g., grpc_wait_all()).

(6) **Step 2:** According to the movements of all agents, the conflict between agents should be avoid in order not to permit several agents in a single cell.

Step 3: Each agent computes the utility function when the agent makes contribution or purchases lottery tickets. Then the probability of making contribution or purchasing lottery tickets is defined according to the calculated utility.

(7) **Step 4:** Collect contributions or profits from the lottery. All the contribution or a part of profit from the lottery system are spent on the cost of preserving the global commons. Apply measures to reduce the pollutants.

(8) **Step 5:** Modify the consciousness of each agent to make contributions changes or modify the probability for each agent to joint the system depends on win or loss in the lottery.

(9) **Step 6:** Collect the information to keep records of the simulation. Back to (3) until the pre-specified condition is satisfied

3. Improving Approach for GridRPC-based MAS software

3.1 Motivation

In this section, we show details of our proposed approach to reduce the computation time of a MAS software implemented by GridRPC approach (say, GridRPC-based MAS software). It can be executed by an assembly of geographically distributed computers, however, the parallelization by GridRPC approach distributes only the computation of agents' decision making to several computing resource as shown in Subsection 2.2. That is, GridRPC-based MAS software couldn't reduce the computation time on cluster computer when the decision making procedure for each agent is short.

The aim of our research is to reduce the computation time of GridRPC-based MAS software on computing grid system. To reduce the computation time of GridRPC-based MAS software, we propose a parallelization approach for procedures of GridRPC-based MAS software regardless of the length of agents' decision making procedure.

3.2 Programming Model of Improving Approach

Our design objective is to reduce the computation time of GridRPC-based MAS software. We propose an improving approach for GridRPC-based MAS software.

Fig. 4 shows an example structure of computing resources using an improving approach. It applies two complementary programming model, remote procedure call and message passing. To combine remote procedure call approach with message passing approach, our proposed model achieves task parallel processing and data parallel processing.

The computing resource of our proposed model consists of computation processors and task submission processors. The computation processors achieve the parallelization of decision making procedure using remote procedure call as well as the GridRPC-based MAS software. To enable the large scale simulation, our approach divides environment information to several task submission processors by message passing.

3.3 Implementation by Improving Approach

In this subsection, we describe how to implement MAS software by our improving approach. In the proposed approach, we divide the algorithm of MAS in Subsection 2.1 as follows (See Fig. 5):

(1) Create the information for the sub-environment, and distribute each task submission processor (e.g., - in Fig. 5) using a scatter module (e.g., MPI_Scatter()).

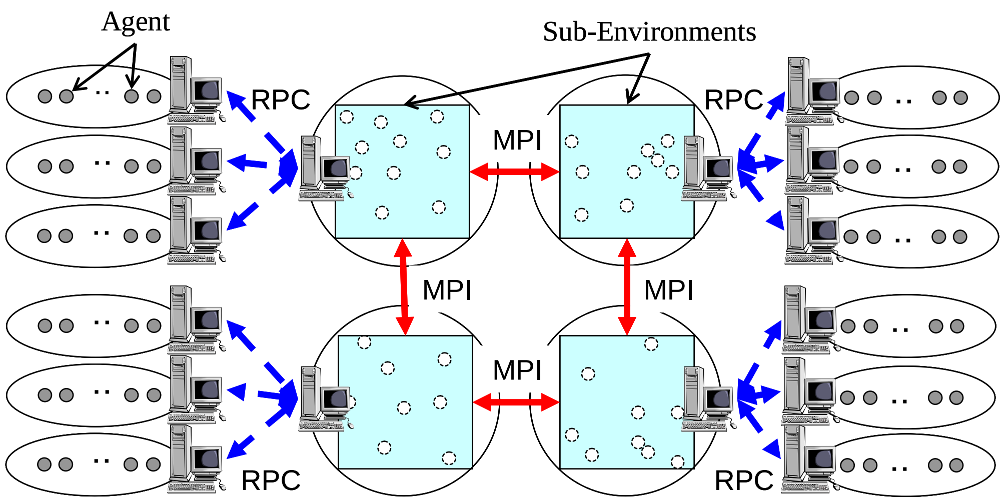

Figure 4. Proposed MAS Programming Model.

(2) Create the initial information for agents of each sub-environment. Since the size of the information depends on the number of agents in the sub-environment, define the amount of the information for each task submission processor by a scatter module (e.g., MPI_Scatter()), then send the information using a scatter module (e.g., MPI_Scatterv()).

(3) Exchange the information from neighboring sub-environments that is required for the processing in the sub-environment. In order to send the information from a task submission processor, use broadcast module (e.g., MPI_Bsend()). For receiving the information, first use a communication buffer allocation module (e.g., MPI_Probe()) to confirm the existence of the information, then receive the information using a receive module (e.g., MPI_Recv()).

(4) Send the information of agents to several computation processors (e.g., - in Fig.. 5) from the each task submission processor using an asynchronous remote procedure call module (e.g., grpc_call_async()).

(5) **Step 1:** Each agent decides to move to a neighboring cell according to the information within its eyeshot.

(6) Collect the decisions of agents to the task submission processor using a wait module (e.g., grpc_wait_all()).

(7) Exchange the information of the agents who moves to another sub-environment of the other task submission processor. Use a send module (e.g., MPI_Bsend()) for the sender, and use a receive module (e.g., MPI_Recv()) after employing a buffer allocation and a measurement module (e.g.,MPI_Probe() and MPI_Get_count( )).

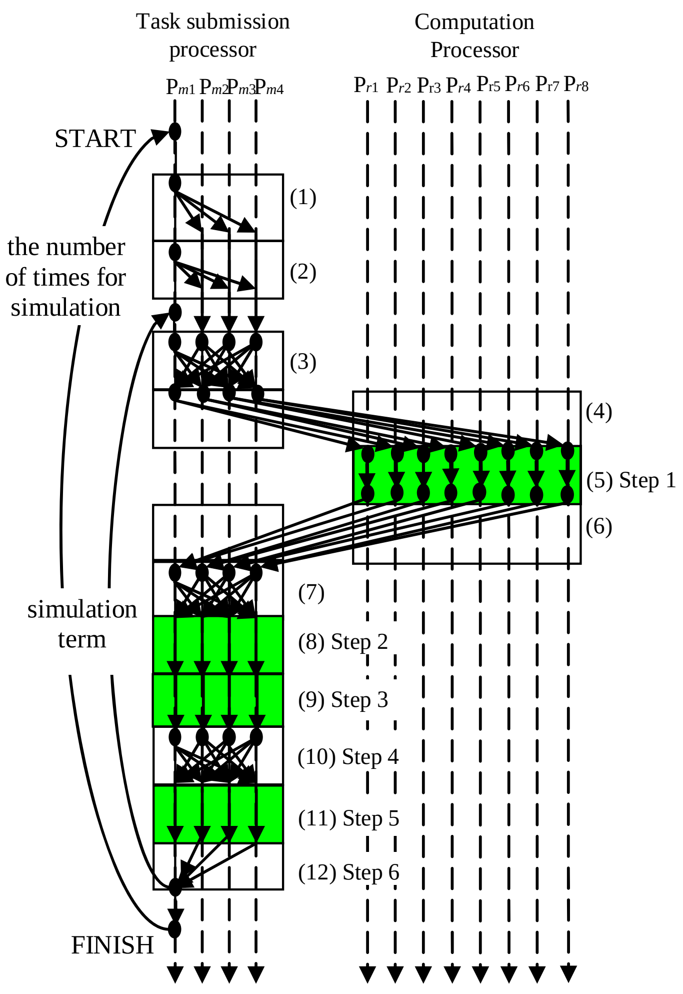

Figure 5. Simulation Flow of MAS with Proposed Approach.

(8) **Step 2:** According to the movements of all agents, the avoidance of conflict between agents should be made in order not to permit several agents in a single cell.

(9) **Step 3:** Each agent computes the utility function when the agent makes contribution or purchases lottery tickets. Then the probability of making contribution or purchasing lottery tickets is defined according to the calculated utility.

(10) **Step 4:** In order to collect contributions or profits from the lottery in each sub-environment, each processor uses a reduction module (e.g., MPI_Allreduce()) to calculate the information, then uses a gathering module (e.g., MPI_Allgather()) to define where the measure should be applied. All the contribution or a part of profit from the lottery system are spent on the cost of preserving the global commons.

(11) **Step 5:** In the contribution system, according to the effectiveness of the financing system to preserve the global commons, the consciousness of each agent to make contributions

changes. In the lottery system, the probability for each agent to joint the system depends on win or loss in the lottery.

(12) **Step 6:** Collect the information of each sub-environment to keep records of the simulation using a reduction module (e.g., MPI_Reduce()). Back to (3) until the pre-specified condition is satisfied.

4. Performance Evaluation

In order to evaluate the performance of MAS software with the proposed improving approach on computing grid system, we prepared two types of MAS software: GridRPC-based MAS software, MAS software with improving approach (say proposed MAS software).

We prepared a homogeneous computing grid system using commodity computers in our experiment. The cluster system has 40 processors with Intel Pentium 4 3.0EGHz processor and 2GByte memory in each computer. All resources are connected by a Gigabit Ethernet. We employed MPICH 1.2.7 and OmniRPC 1.0 as MPI and GridRPC library, respectively.

When we execute MAS software for the performance evaluation, the number of simulation time steps and initial agents are 2000 and 4320, respectively.

4.1 Computation Time

Figs. 6 and 7 show the computation time for each trial to execute GridRPC-based MAS software and proposed MAS software on multi-node system. And Fig. 8 shows the relationship speedup ratio and the environment size on multi-node system. To obtain the result in Fig. 8, each task submission processor was prepared on three computation processor when we execute the GridRPC-based MAS software and the proposed MAS software. Here “Combine 4 Nodes” and “Combine 9 Nodes” represent the experiment results of proposed MAS software in four or nine task submission processor. “GridRPC” represents the experiment results of GridRPC-based MAS software. In this paper, speedup ratio defines the following function.

$$Speedup = \left(1 - \frac{T_P}{T_G}\right) \times 100,$$

where T_P and T_G are the computation time of proposed MAS software and GridRPC-based MAS software, respectively.

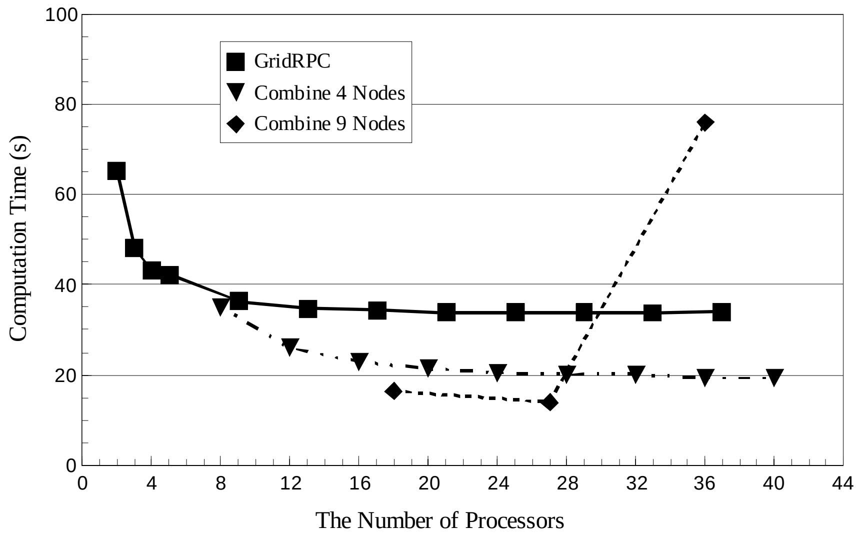

Figure 6. Computation Time (Environment Size: 180x180).

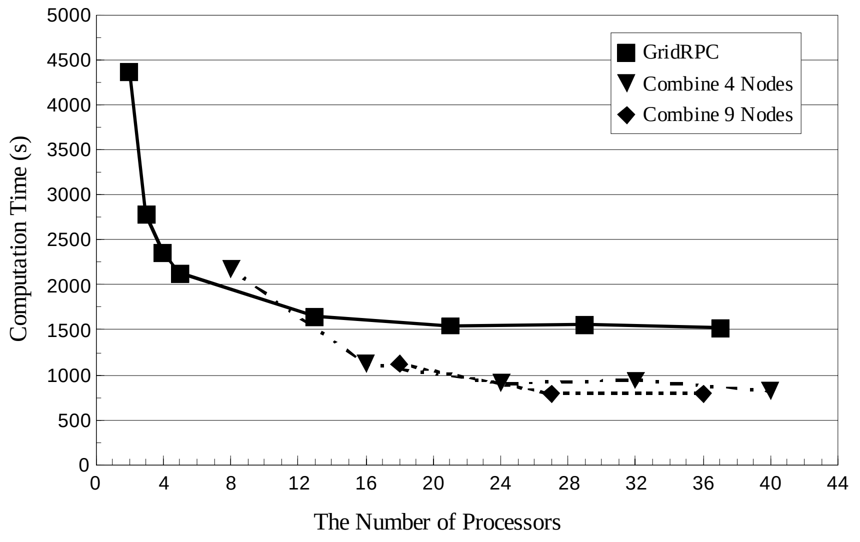

Figure 7. Computation Time (Environment Size: 720x720).

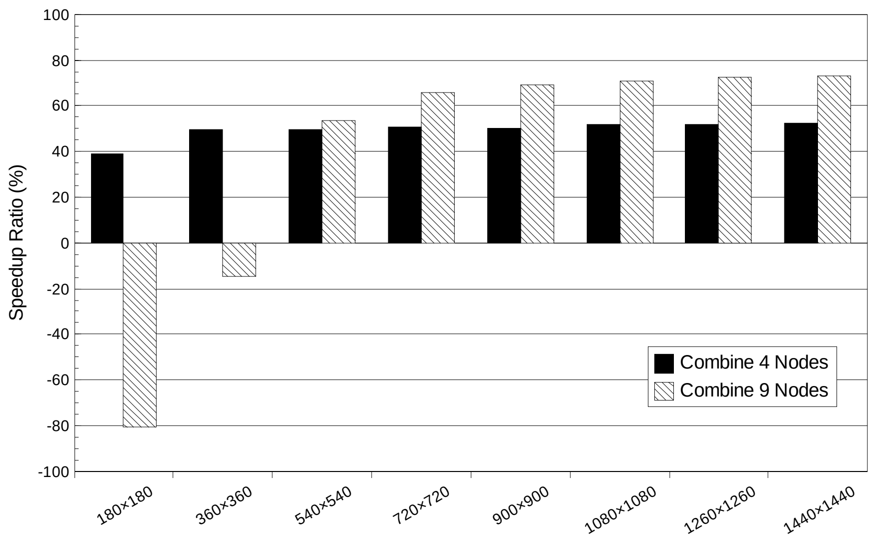

Figure 8. Relationship Speedup Ratio and the Environment Size.

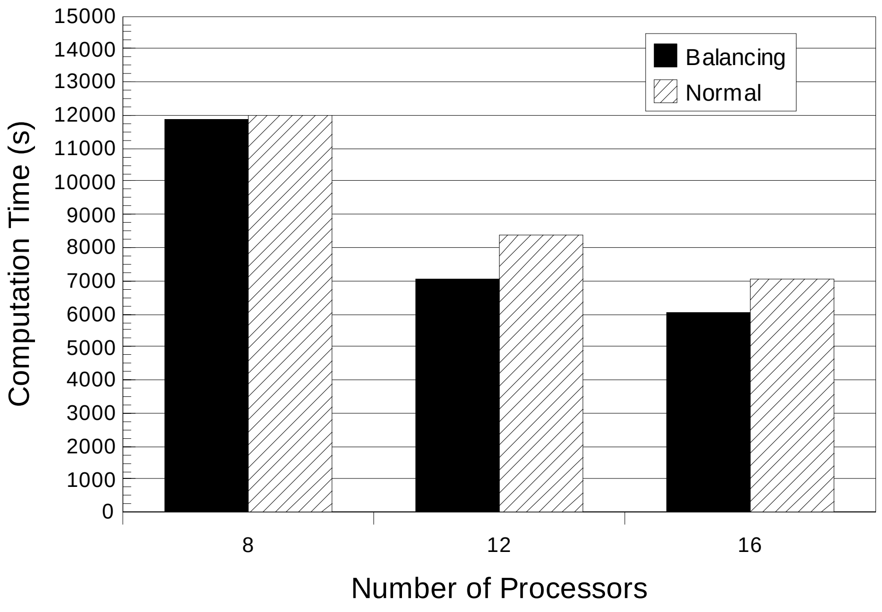

Figure 9. Computation Time by Effect of Load Balancing.

In Fig. 6, 7 and 8, we can see that the proposed MAS software reduces the computation time as a whole. That is, the effect of parallel processing in the part of step 2, 3, 4, 5 and 6 (see Fig. 5) has achieved. However, Fig. 6 and 8 indicate that the execution time will increase when the proposed MAS software was executed in nine task submission processors. Hence, we can explain that the number of processor need not be excessively increased.

4.2 Effect of Load Balance

We investigate the effectiveness of load balancing in the proposed MAS software on four task submission processor. The computation time of the proposed MAS software is shown in Fig. 9. To obtain the result in Fig. 9, we execute the proposed MAS software using four task submission processors. The environment size and the number of initial agents are 360×360 and 4320, respectively. Here "Balancing" represents the experiment results of the proposed MAS software with the load balancing mechanism. "Normal" represents the experiment results of the proposed MAS software without the load balancing mechanism. In this paper, load balancing mechanism is a mechanism that each task submission processor changes the number of assigned computation processor according to the number of agents.

In Fig. 9, we can see that the proposed MAS software with the load balancing mechanism reduces the computation time.

5. Conclusion

We propose how to improve the performance of GridRPC-based MAS software for large-scale simulation. Our improving approach is to apply two generally programming model, message passing and remote procedure call. Then, we evaluate the MAS software with our proposed approach. We confirm that the computation time of GridRPC-based MAS software is improved by proposed approach. Our improving approach can dynamically obtain the memory space for large-scale simulation from several computing resources. So, we found that the sugarscape space of MAS can be increased by the proposed approach. Our proposed approach will be helped by programmers who should develop large-scale social simulation software for heterogeneous computing resources such as Grid.

References

[1] N. Gilbert, K. G. Troitzsch, Simulation for the Social Scientist (Open University Press, UK), 1999.

[2] R. Conte, N. Gilbert, J. S. Sichman, “MAS and social simulation: A suitable commitment,” Proc. of 1st International Workshop on Multi-Agent Systems and Agent Based Simulation (Lecture notes in computer science 1534, Springer, Berlin, Germany), pp. 1-9, 1998.

[3] T. Terano, S. Takahashi, D. L. Sallach, J. Rouchier (eds), Proc. of 1st World Congress on Social Simulation, August 21-25, 2006.

[4] N. Gilbert, “A simulation of the structure of academic sciences,” Sociological Research Online, Vol. 2, No. 2 <http://www.socresonline. org.uk/2/2/3.html>, 1997.

[5] T. Murata, H. Kitano, T. Nakashima, H. Ishibuchi, “Application of a Multi-Agent Model with Pioneers and Followers to a Day Care Center Allocation Problems,” Proc. of International Conference on Intelligent Technologies 2003, pp.179-186, 2003.

[6] C.F.Manski, “Economic Analysis of Social Interactions” Journal of Economic Perspectives, Vol.14, No.3, pp.115-136, 2000.

[7] H. Arikawa, T. Murata, “Implementation issues in a Grid-based multi-agent simulation system used for increasing labor supply,” Review of Socionetwork Strategies, Vol. 1, No. 1, pp.1-13, 2007.

[8] T. Murata, H. Arikawa, S. Morishita, Taiyo Maeda, “A Design of Problem Solving Environments for Policy Making Assistance Using MAS-based Social Simulation,” Proc. of 3rd IEEE International Conference on e-Science and Grid Computing, pp.521-528, 2007.

[9] J. M. Epstein, R. Axtell, Growing Artificial Societies: Social science from the bottom up, MIT Press, 1996.

[10] I. Nishizaki, Y. Ueda, T. Sasaki, “Lotteries as a means of financing for preservation of the global commons and agent-based simulation analysis,” Applied Artificial Intelligence, Vol. 19, No. 8, pp. 721-741, 2005.

[11] K. Seymour, H. Nakada, S. Matsuoka, J. Dongarra, C. Lee, H. Casanova, “Overview of GridRPC: a remote procedure call API for Grid Computing,” Proc. of 3rd International Workshop on Grid Computing, pp. 274-278, 2002.

[12] M. Lees, B. Logan, R. Minson, T. Oguara, G. Theodoropoulos, “Distributed simulation of MAS,” Proc. of the Joint Workshop on Multi Agent & Multi-Agent-Based Simulation, Autonomous Agents & Multi Agent Systems (AAMAS), pp. 21-31, 2004.

[13] I. J. Timm, D. Pawlaszczyk, “Large scale multiagent simulation on the grid,” Proc. of 5th International Symposium on Cluster Computing and the Grid, Vol. 1, pp. 334-341, 2005.

[14] T. Takahashi, H. Mizuta, “Efficient agent-based simulation framework for multi-node supercomputers,” Proc. of 2006 Winter Simulation Conference, pp. 919-925, 2006.