PG Lab ディスカッションペーパーシリーズ
第26号　2008年４月

ISSN 1880-9146

Discussion Paper Series
No.26 April, 2008

Policy-Making Assistance Using ELASTIC: Enhanced Large-Scale Agent Simulation Toolkit for Innovative Community

Hiroshi ARIKAWA, Sen-ichi MORISHITA and Tadahiko MURATA

文部科学省私立大学社会連携研究推進拠点
関西大学政策グリッドコンピューティング実験センター

Policy Grid Computing Laboratory,
Kansai University
Suita, Osaka 564-8680 Japan
URL : https://www.pglab.kansai-u.ac.jp/
e-mail : pglab@jm.kansai-u.ac.jp
tel. 06-6368-1228
fax. 06-6330-3304

関西大学政策グリッドコンピューティング実験センターからのお願い

本ディスカッションペーパーシリーズを転載，引用，参照されたい場合には、
ご面倒ですが、弊センター（pglab@jm.kansai-u.ac.jp）宛に
ご連絡いただきますようお願い申しあげます。

Attention from Policy Grid Computing Laboratory, Kansai University

Please reprint, cite or quote WITH consulting Kansai University Policy Grid Computing Laboratory (pglab@jm.kansai-u.ac.jp).

Policy-Making Assistance Using ELASTIC: Enhanced Large-Scale Agent Simulation Toolkit for Innovative Community

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

Abstract

This paper introduces ELASTIC (Enhanced Large-scale Agent Simulation Toolkit for Innovative Community) that is a large-scale multi-agent simulation (MAS) toolkit for policy-making assistance. Although designers of a multi-agent simulation can employ any programming language to write their program on computing resources, they have some difficulty to describe the relation of agents and an environment where those agents take their actions. Through our experience to develop some MAS-based softwares, we have found some requirements to develop an effective MAS toolkit. In this paper, we illustrate the design and implementation of ELASTIC. And, we show how to apply ELASTIC to a social simulation software for policy-making assistance.

Keywords: multi-agent simulation, large-scale simulation, social interaction among agents.

1 Policy Grid Computing Laboratory, Kansai University

2 Department of Informatics, Kansai University

1 Introduction

Multi-Agent Simulation (MAS) is one of promising research fields in artificial intelligence. Since 1990s, computer simulations in social science have attracted many social scientists [1]. Social science includes various research fields such as sociology, economics, social psychology, organization theory, political science, demography, anthropology, archaeology, and so on. One of reasons why MAS is employed for social simulation is its possibility to imitate social system behaviors [2-5].

To imitate a society using MAS, researchers should design a social simulation model and prepare the MAS-based simulation software. Many researchers have proposed MAS toolkits and libraries [7-12]. They have discussed how to design the social simulation model. However, they have unclearly discussed how to implement the large-scale social simulation software by MAS for policy-making assistance.

Our primary objective is to demonstrate the effectiveness to develop a large-scale social simulation software for policy-making assistance. We have proposed how to implement the large-scale social simulation software by MAS for policy-making assistance [13-15].

A large-scale social simulation can be categorized to two groups. One is to simulate with a huge number of parameter specifications. In social simulations, there are a huge number of specifications such as the number of agents, the distribution of resources for agents, a decision making process in each agent, and so on. In order to find the best specifications for the social simulation, the designer of the simulation should examine the best parameter settings for it. In [13], we show a way to find the good parameter specifications with the aid of users' interaction.

The other group of a large-scale social simulation is to simulate a society with a huge number of agents. While it depends on the size of memory in a computing resource, the number of agents that can be tractable in a single computing resource may be 10^5 - 10^7. The number is enough for a simulation for a city or a region, though, an alternative should be proposed for a simulation for a larger society such as a state, a nation, or a globe. A way of implementing a MAS software with a huge number of agents is to parallelize it using parallel and distributed computing environment. In our previous research, we have developed two distinct MAS software using two traditional parallelized programming approaches: message-passing approach and remote procedure call approach [14]. The MAS software applied by the former approach is called MPI-based MAS software. On the other hand, that applied by the latter approach is called GridRPC-based MAS software. We found that the MPI-based MAS software enables to get large-scale memory space from several computing resources. We have also proposed an approach to improve a GridRPC-based MAS software [15]. The approach combines a remote procedure call approach with

a message passing approach, and it can dynamically allocate the large-scale memory space from several computing resources.

Another way of implementing a MAS software with a large number of agents is to modify MAS software to get a large-scale memory space from a computer resource. As for existing MAS toolkits and libraries such as SOARS [7], PlatBox Simulator (former Boxed Economy Simulation Platform) [9], RePast [10], MASON [11] and Swarm[12] are well-known MAS toolkit, we find difficulties to modify social simulation software with a large number of agents using these libraries. We see a need of a tool that enables us to develop a MAS software with a large number of agents. Therefore, we have developed a large-scale MAS toolkit called "ELASTIC." In this paper, we introduce several features of our proposed toolkit. Then, we illustrate how to implement social simulation software for policy-making assistance using ELASTIC.

2 Requirements of MAS Software for Policy-Making Assistance in Implementing

We have designed simulation software for policy-making assistance with economists, social psychologists and political scientists, in Policy Grid Computing Laboratory (PG Lab), Kansai University, and we have also implemented social simulation software using MAS. Through our activity we obtained some empirical knowledge. We found that there is a gap in designing MAS software between theory and practice. In this section, we explain requirements of MAS software for policy-making assistance.

2.1 Reality

To solve various problems in governments and local communities, stakeholders want to obtain results of social simulation with reality. They need to obtain detail information on who will benefit from a new scheme, or what by-effect should be expected in planning region. More concrete results give better suggestions to stakeholders and decision makers.

An issue of developing MAS software for policy-making assistance is to obtain more concrete results. To design a simulation model using MAS, designers should consider behaviors of agents, environments, interactions among agents, and those between agents and environments.

2.2 Programmability for Memory Allocation

To implement a large-scale MAS software, software developers should have some technique. According to our empirical knowledge, we should achieve how to get a large-scale memory space from a computing resource. To allocate a memory space dynamically on a computing resource is the most difficult in implementing a large-scale MAS software. Software developers will get memory

spaces that exceed the maximum of memory space on a computing resource for a large-scale simulation. However, if they do not have a technique which allocates large-scale data to memory space on a computing resource, they can not implement a large-scale MAS software. Generally speaking, software developers may use a MAS toolkit without advanced technique to obtain a large-scale memory from a computing resource. They need a toolkit that can be used easily.

3 ELASTIC: Enhanced Large-Scale Agent Simulation Toolkit for Innovative Community

3.1 Motivation

For reducing the gap between theory and practices in developing social simulation softwares for policy-making assistance, we have promoted interdisciplinary researches in Policy Grid Computing Laboratory (PG Lab), Kansai University. We are providing an implementation approach in designing social simulation models for policy-making assistance. As a research in PG Lab, we have developed a large-scale MAS toolkit that is called ELASTIC (we call it OpenMAST previously).

Our goal in developing ELASTIC is to facilitate the implementation of the large-scale multi-agent simulation software with social interactions for innovative community. We define a innovative community as a group with research and development members from various background including social science, computer science, computation science, government, nonprofit organization and others. ELASTIC can be provided for innovative community to design and implement MAS software simply.

3.2 Simulation Model in ELASTIC

Figure 1 illustrates a simulation model in ELASTIC. In this paper, “Simulation Environment” is defined as an initial dataset which provides with circumstances in running simulation software. For example, a simulation environment in the MAS with a classical sugarscape model [16] regards as the space of sugarscape, the location of agents on sugarscape in initial and the amount of sugar in each cell.

In MAS, an agent gets information from other agents and an environment where the agent exists, then it decides next actions according to the obtained information. To implement a simulation software using ELASTIC, designers need to design two agent behavior rules clearly: (1) how to get information and (2) decision-making process.

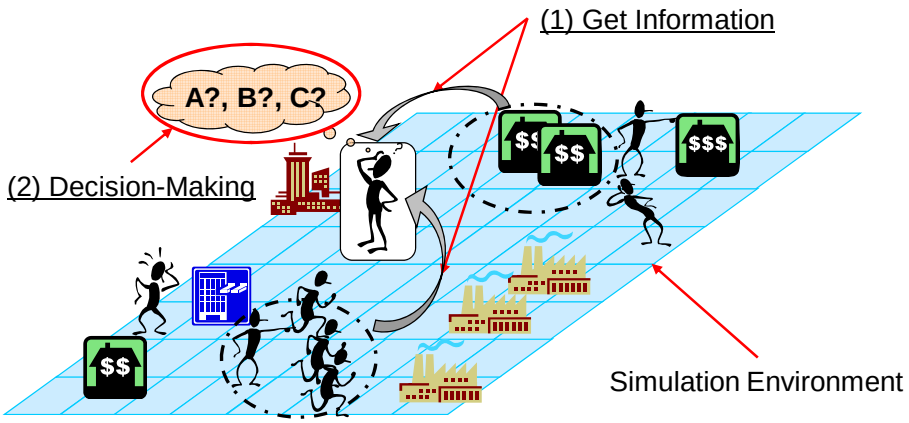

Figure 1: A Simulation Model in ELASTIC

3.3 Relationship Agent Behavior And Information

To design simulation models using ELASITC for policy-making assistance, model designers should define an agent behavior rule based on the social interaction model. Many social scientists proposed various social interactions. For example, Manski [6] has proposed three social interactions: constraint interaction, preference interaction, and expectation interaction. In this subsection, we explain a relation between agent behavior rule and social interactions. Constraint interaction is regarded as an agent behavior influenced by the environment. Designers of a simulation describe it as a rule to obtain information from the environment. Preference interaction is regarded as an agent behavior influenced by actions of other agents. To describe an agent behavior rule with preference interaction, designers define it as a rule for an agent to obtain information and change the priority of actions according to actions of other agents.Expectation interaction is regarded as an agent behavior influenced by a trend of actions and information from other agents. Designers describe a rule with expectation interaction by defining ways of obtaining information from other agents.

3.4 Simulation Environment in ELASTIC

Users who compute social simulations using MAS software need to define some simulation environments that represent spaces where agents exist. Many MAS toolkits have a restriction in modeling simulations. Sugarscape model is a well-known model in MAS. However, we find that it is not enough for simulation in policy-making. To design simulation model for policy-making assistance, we consider that designers and users should clearly define simulation environments.

ELASTIC provides two types of simulation environments. Users utilize these environments appropriately according to the simulation model. In this subsection, we explain how to define simulation environments in computing social simulation using ELASTIC.

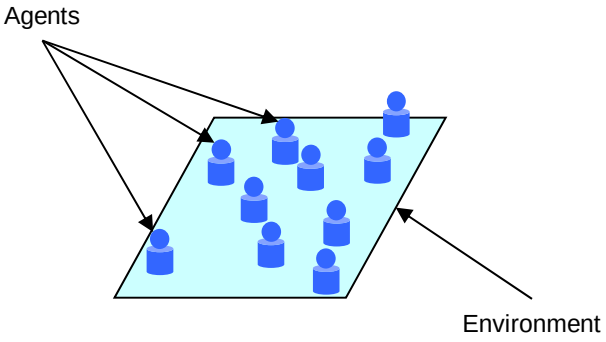

Figure 2: Simulation Condition in ELASTIC (Physical Environment)

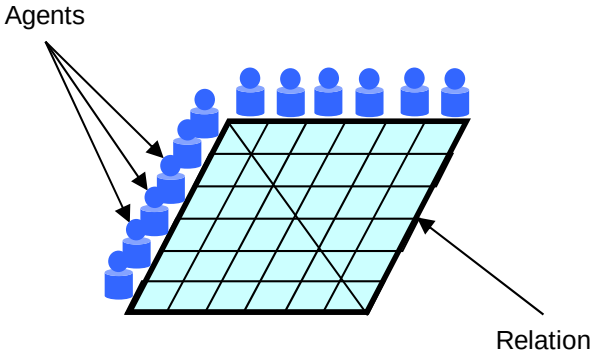

Figure 3: Simulation Condition in ELASTIC (Non-Physical Environment)

Physical Environment:

Figure 2 shows an example of physical environment in ELASTIC. This environment can be used to investigate the influence from neighboring agents and the effect of distance between agents in an environment. To apply the simulation with constraint interaction, users can employ this environment.

This environment is defined as cellular-based space as sugarscape model. If a change of state in this environment occurs, users define transition rules in the environment at each cell.

Non-Physical Environment:

Figure 3 shows an example of non-physical environment in ELASTIC. Non-physical environment can be represented by relations among agents. The figure illustrates the existence of a relation between one agent and another. To describe rules with preference interaction and expectation interaction, users can employ this environment.

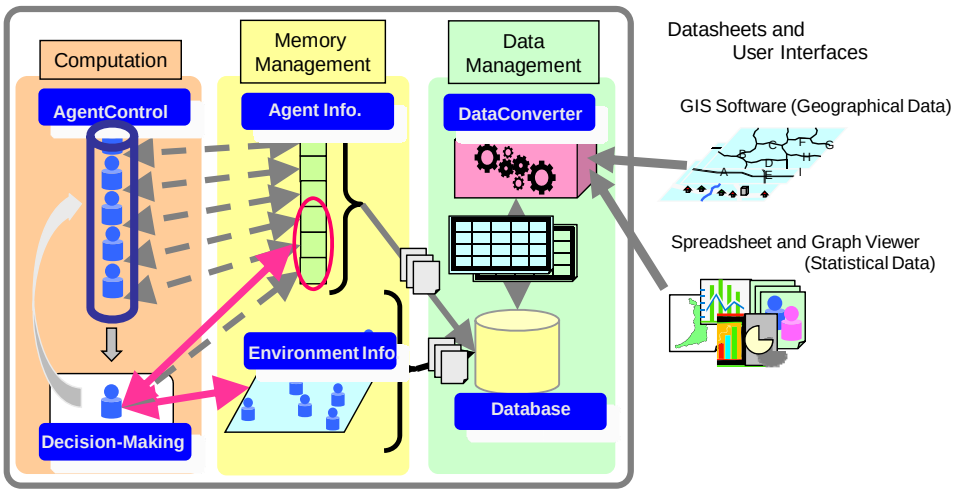

Figure 4: ELASTIC Architecture

3.5 Architecture

Figure 4 shows the Architecture of ELASTIC. ELASTIC consists of three modules: computation module, memory management module, and data management module. These three modules can be explained as follows:

Computation Module:

Computation Module computes decision-making process using the agent and environmental information, and it controls decision-making process of each agent.

Memory Management Module:

Memory Management Module allocates the area of agents and environmental information to memory on a computing environment dynamically.

Data Management Module:

Data Management Module supports for users to convert geographical data and statistical data into the agent information and environmental information, and it archives the agent and environmental information to databases.

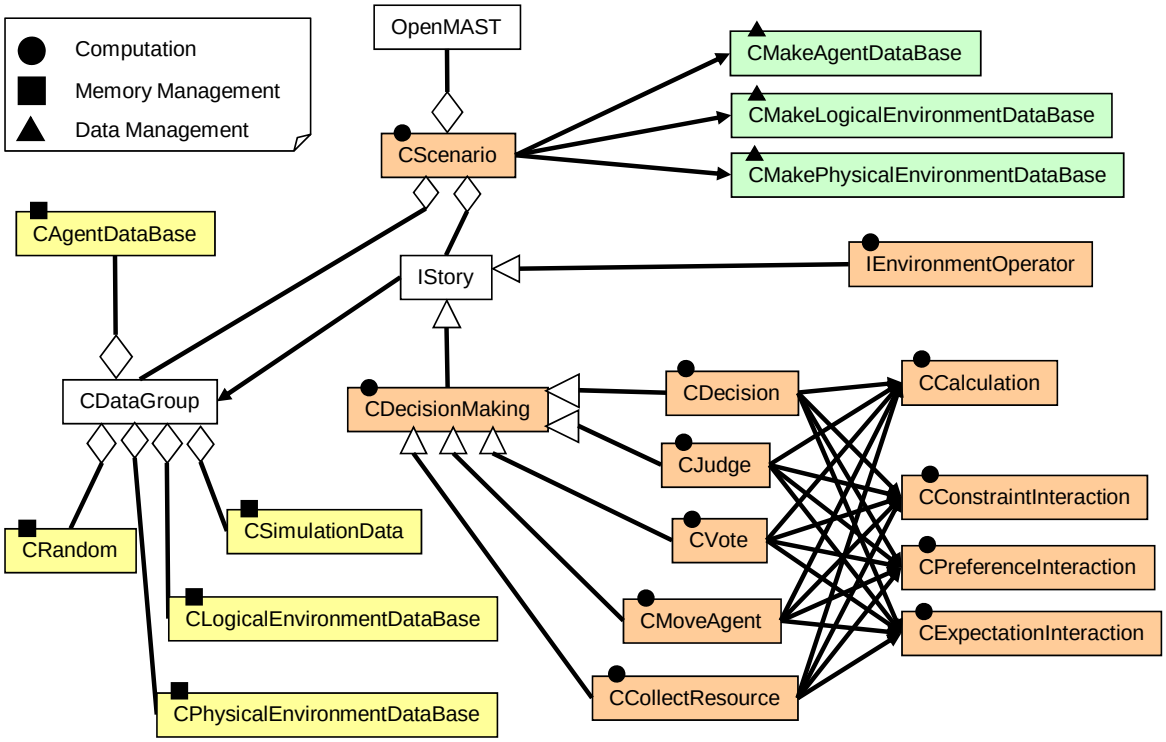

Figure 5: ELASTIC Class Diagram

3.6 ELASTIC class and APIs

ELASTIC is written in C++ in order to take advantage of its general versatility. Software developers can implement their MAS software based on Object-Oriented Programming (OOP) using ELASTIC. By applying OOP to ELASTIC, users can utilize useful codes which they implemented previously.

To implement various MAS software easily, ELASTIC provides useful application programming interfaces (APIs) for MAS software developers. Figure 5 illustrates the ELASTIC class diagram. Through our previous experience, many developers had designed agent behavior-based MAS software in implementing simulation software by MAS. Therefore, ELASTIC mainly provides decision-making process for users. Users choose necessary API in implementing MAS software for policy-making assistance.

Agent behavior rules for the decision-making, the judgment, the vote, the moving and the correction in an environment are implemented as CDecision class, CJudge class, CVote class, CMoveAgent class, and CCollectResource class, respectively. If an environment has a state as sugarscape-based MAS, developers implement transitions in a physical environment using some APIs in IEnvironmentOperator class.

CLogicalEnvironmentDataBase class, CPhysicalEnvironmentDataBase class and CAgentDataBase class are provided for getting the large-scale memory spaces from a computing resource. Their classes allocate memory spaces dynamically to a computing resource according to the data size. Developers do not have to allocate memory spaces themselves in implementing the large-scale MAS software. CSimulationData class is provided for developers to show values (e.g., counter in a simulation and agent information on memory space).

CMakeLogicalEnvironmentDataBase class, CMakePhysicalEnvironmentDataBase class and CMakeAgentDataBase class are provided to input and output data files. All data files in a simulation define the comma-separated values (CSV) format in ELASTIC. For example, simulation condition data of each agent is described that each record in a CSV file. Thus, the number of lines in a CSV file is equal to the number of agents.

4 Case Study

We have already implemented some MAS-based social simulation software for policy-making assistance using ELASTIC. In this section, we illustrate how to apply ELASTIC to the social simulation for policy-making assistance.

4.1 Target Simulation

In this paper, we employ a day care center allocation simulation model that was proposed by Murata *et.al* [4,5]. Their simulation model is developed using real survey data.

Figure 6 shows an outline of the simulation model. In this figure, there is one office and seven agents in a two-dimensional world $[0, 5000]^2$. Each agent can work at an office if the agent's utility function enables the agent to offer itself to a job market, and if the office manager employs the agent. They assume that the labor supply always matches the labor demand for simplification. The agent depicted by a circle in Figure 6 shows an agent that works at an office. When there are several work places, the agents chose to work at the nearest office. A triangle indicates an agent that does not work. If the agent cannot work because the agent has a child or children, the agent reduces the number of children by bringing them to a day care center. If the agent's reason to be at home is not due to child care, the agent cannot benefit from the day care center.

Their MAS model for the day care center allocation problem considered three rules of agent behaviors: utility function for decision-making, utilization of the day care center, and influence from neighboring agents.

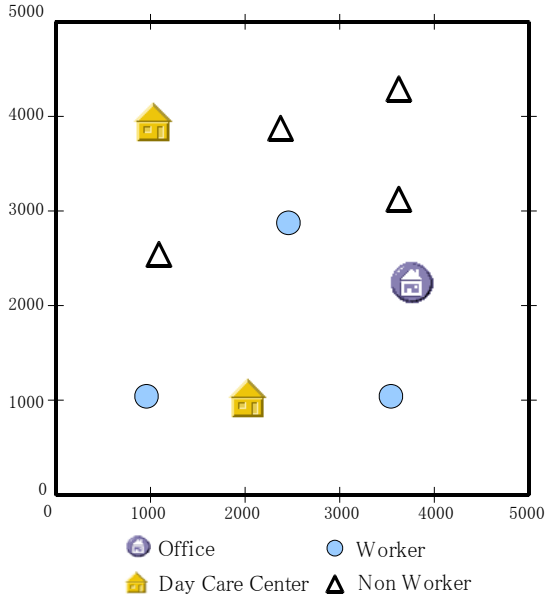

Figure 6: Outline of Simulation model for the day care center allocation problem

Utility function for decision-making

Every agent has the following utility function $U(C18, AGE, ED, HI)$ for making a decision as to whether or not to enter the job market.

$$U = U_W - U_H = \beta_1 + \beta_2 \times C18 + \beta_3 \times AGE + \beta_4 \times AGE^2 + \beta_5 \times ED + \beta_6 \times HI\,, \quad (1)$$

where U_W and U_H are the utility of working and not working, respectively. In addition, $C18$ is the number of children under 18, AGE is the agent's age, ED is the years of education, and HI is the mate's earnings. If $U_W \geq U_H$, the agent decides to enter the job market, and selects the nearest work place. On the other hand, the agent does not offer itself to the job market if $U_W < U_H$.

In order to define the coefficients β_i ($i = 1,...,6$) in (1), we estimate them using the logit model. The logit model is ordinarily employed for binary data analysis. The coefficients were estimated by using a data set based on the working status of 50 married women sampled from the *Current Population Survey in 1993* by the US Bureau of the Census. In [4,5], the value of coefficients β_i are shown as follows:

$$\beta_1 = -2.302\,,\ \beta_2 = -0.667\,,\ \beta_3 = 0.245\,,$$
$$\beta_4 = -0.004\,,\ \beta_5 = 0.085\,,\ \beta_6 = -8.2 \times 10^{-6}\,. \quad (2)$$

Since the coefficient of β_2 is less than zero, the value of the utility function can be increased by decreasing the value of $C18$. The agent can decrease its value of $C18$ by leaving its children in a day care center during its working time. Therefore the number of agents who have the utility value $U \geq 0$ will increase, and the number of workers also will increase.

Utilization of day care center for agents

An agent who does not work because of its children has the possibility to become a worker using a day care center. In this model, they allow only agents with the following condition to utilize a day care center. In the following, $D_{NW}(A)$, $D_{ND}(A)$ and $D_{ND-W}(A)$ denote the distance between Agent *A* and the nearest workplace, the distance between *A* and the nearest day care center, and the distance between the nearest day care center and the nearest working place from the day care center.

$$D_{ND}(A) + D_{ND-W}(A) \leq tol \times D_{NW}(A), \tag{3}$$

where *tol* is a tolerance factor for Agent *A* to utilize the nearest day care center. That is, if $tol = 1$, the total distance of $D_{ND}(A)$ and $D_{ND-W}(A)$ should be equal to or less than the direct distance to the nearest work place $D_{NW}(A)$ in order to utilize the day care center.

Influence from neighboring agents

Each agent is influenced by the decisions of neighboring agents. Their model was assigned two-ways for the influence of neighboring agents: a leader type and a follower type. Each agent can get the information on the working status of other agents in its perception area. Each agent can gather the information within the area of 500×500. That is, if an agent locates in a two dimensional world at [2500, 2500], it can perceive in a two dimensional world within the square surrounded by the four points [2250, 2250], [2250, 2750], [2750, 2250], and [2750, 2750]. This square covers 1% of whole area of the world.

Each agent counts the number of agents and workers in its perception area. According to the ratio of the workers in the perception area, the value of the coefficient β_1 in the utility function of the agent is modified. Their model was assigned a different modification rule to a leader agent L and a follower agent F, respectively. Their model was defined $RW(A)$ as the ratio of the workers in the perception area of Agent *A*.

[Modification Rules for Leader Agents]

If $0 \leq RW(L) < 0.2$, then decrease β_1 by $\Delta\beta$,

If $0.2 \leq RW(L) < 1.0$, then increase β_1 by $\Delta\beta$.

[Modification Rules for Follower Agents]

If $0 \le RW(F) < 0.4$, then decrease β_1 by $\Delta\beta$,

If $0.4 \le RW(F) < 0.6$, then let β_1 be,

If $0.6 \le RW(F) \le 1.0$, then increase β_1 by $\Delta\beta$.

In these cases, a leader agent is motivated to work when there are few worker agents around it. On the other hand, a follower agent is motivated to work when workers are the majority around it. Each agent can utilize local information within its perception area. Therefore, if there are few worker agents around it, even a leader agent will become less interested in work and then the value of β_1 will be decreased. On the other hand, when almost all neighboring agents are working, there are no other choices for the leader agent than working. That is, it does not choice of not working.

4.2 Implementation using ELASTIC

We illustrate how to implement simulation software for a day care center allocation problem. In this subsection, we introduce the major parts of software in implementing their software.

Main function:

We entry a necessary functions and variables to main( ) in OpenMAST class. Figure 7 shows the code of main( ) in our implemented software.

Decision-making function:

We have implemented some codes for decision-making to each DecisionMaking( ) in CDecison, CJudge and CVote class. CDecision, CJudge and CVote class define the decision as to whether or not to enter the job market, the utilization of day care center for agent and the influence of neighboring agents, respectively. Figure 8 shows the code of CDecision::DecisionMaking ( ) in our implemented software. The code in Figure 8 is implemented the utility function for decision-making in (1).

```
int main(int argc, char* argv[])
{
    CScenario Scenario(argv[1], argv[2], argv[3],argv[4]);
    CCheckParameter CheckParameter;
    CDecision Decision("women");
    CJudge Judge("women");
    CVote Vote("women");
    CTakujishoOutPutData1 TakujishoOutPutData1;
    CTakujishoOutPutData2 TakujishoOutPutData2;
    CTakujishoOutPutData3 TakujishoOutPutData3;

    //initialization
    Scenario.AddInitStory(&CheckParameter);
    Scenario.AddInitStory(&Decision);
    Scenario.AddInitStory(&TakujishoOutPutData1);
    Scenario.AddInitStory(&Judge);
    Scenario.AddInitStory(&Decision);
    Scenario.AddInitStory(&TakujishoOutPutData2);
    // entry the class using the simulation
    Scenario.AddMainStory(&Vote);
    Scenario.AddMainStory(&Decision);
    Scenario.AddMainStory(&TakujishoOutPutData3);
    // execute the simulation
    Scenario.RunScenario();
}
```

Figure 7: Sample Code of main( )

```
void CDecision::DecisionMaking(CDataGroup& DataGroup)
{
    //Decision-making in working office
   if(DataGroup.AgentDataBase.GetPropaty("logit1")
      + DataGroup.AgentDataBase.GetPropaty("logit2")
      * DataGroup.AgentDataBase.GetPropaty("C18")
      + DataGroup.AgentDataBase.GetPropaty("logit3")
      * DataGroup.AgentDataBase.GetPropaty("AGE")
      + DataGroup.AgentDataBase.GetPropaty("logit4")
      * pow(DataGroup.AgentDataBase.GetPropaty("AGE"), 2)
      + DataGroup.AgentDataBase.GetPropaty("logit5")
      * DataGroup.AgentDataBase.GetPropaty("ED")
      + DataGroup.AgentDataBase.GetPropaty("logit6")
      * DataGroup.AgentDataBase.GetPropaty("HI") < 0.0)
      {
          DataGroup.AgentDataBase.SetPropaty("work", 0);
    }
    else{DataGroup.AgentDataBase.SetPropaty("work", 1);
    }
}
```

Figure 8: Sample Code of DecisionMaking( ) in CDecision API

4.3 Effectiveness of ELASTIC

We can implement a ELASTIC-based simulation software for a day care center allocation problem. We found that the ELASTIC-based simulation software was reduced 65% of code sizes in implementing software by comparison our previous implemented software in [13]. We also found that the ELASTIC-based simulation software could execute on a computing resources using 3×10^5 agents file.

As a result, we found that developers can easily develop the large-scale social simulation software using ELASTIC

5 Conclusion

In this paper, we presented ELASTIC, a large-scale multi-agent simulation toolkit for policy-making assistance. We have developed our toolkit since previous researches still have difficulties to implement the large-scale simulation software for policy-making assistance. To provide ELASTIC, social scientists and software developers can easily develop the large-scale social simulation software with social interactions, and they can improve the social simulation software. We believe that developers will reduce the cost in developing software using ELASTIC.

Acknowledgement

This work was partially supported in part by the Japan MEXT (Ministry of Education, Culture, Sports, Science and Technology) under Collaboration with Local Communities Project for Private Universities starting 2005. In developing ELASTIC (we call it as OpenMAST in the project), we are funded by Exploratory Software Project, Information-technology Promotion Agency (IPA), Japan for 2007.

References

[1] N. Gilbert and K.G. Troitzsch, Simulation for the Social Scientists, Open University Press, Buckingham , UK, 1999.

[2] N. Tanida and M. Murakami, A study on Japanese Public Pension System using Multi Agent Based Simulation, In the Proceedings of the 10th Annual Workshop on Economic Heterogeneous Interacting Agent, pp.1-12, 2005.

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

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

[5] T. Murata, H. Kitano, Y. Kadooka, Y. Ukai, Political Multi-Agent Simulation with Grid Computing., RCSS Discussion Paper Series, No.17, Research Center of Socionetwork Strategies, The Institute of Economic and Political Studies, Kansai University, 2004.

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

[7] H. Deguchi, H. Tanuma and T. Shimizu, SOARS: Spot oriented agent role simulator – Design and agent based dynamical system, Proceedings of Third International Workshop on Agent-based Approaches in Economic and Social Complex Systems, pp.49-56, 2004.

[8] T. Takahashi and H. Mizuta, Efficient agent-based simulation framework for multi-node supercomputers, Proceedings of 2006 Winter Simulation Conference, pp.916-925, 2006.

[9] T. Iba, Y. Takabe, Y. Chubachi, J. Tanaka, K. Kamihashi, R. Tsuya, S. Kitano, M. Hirokane, Y. Matsuzawa, Boxed Economy Foundation Model: Toward Simulation Platform for Agent-Based Economic Simulations, New Frontiers in Artificial Intelligence, Springer-Verlag, pp.227-236, 2001.

[10] N.J. North, N.T. Collier and J.R. Vos, Experiences creating three implementations of the repast agent modeling toolkit, ACM Transaction on Modeling and Computer Simulation, Vol.16, No.1, pp.1-25, 2006.

[11] S. Luke, G.C. Balan, L.Panait, C. Cioffi-Revilla and S. Paus, MASON: A Java multi-agent simulation library", Proceedings of Agent 2003 Conference on Challenges in Social Simulation, 2003.

[12] N. Minar, R. Burkhart, C. Langton, M. Askenazi, The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations, SFI Working Paper 96-06-042, pp.1-11, 1996.

[13] H. Arikawa and T. Murata, Implementation Issues in a Grid-based Multi-Agent Simulation System used for Increasing Labor Supply, The Review of Socionetwork Strategies, No.1, pp.1-13, 2007.

[14] T. Murata, H. Arikawa, S. Morishita and T. Maeda, A Design of Problem Solving Environments for Policy Making Assistance Using MAS-Based Social Simulation, Proceedings of Third IEEE International Conference on e-Science and Grid Computing, pp. 521-528, 2007.

[15] H. Arikawa, S Morishita and T.Murata, Performance Improvement of GridRPC-based Multi-Agent Simulation Software, PG Lab Discussion Paper Series, No.20, Policy Grid Computing Laboratory, The Institute of Economic and Political Studies, Kansai University, pp.1-14, 2007.

[16] J.M. Epstein and R. Axtell, Growing Artificial Social Science from the Bottom Up, MIT Press, 1996