2007 1st Annual IEEE Systems Conference
Waikiki Beach, Honolulu, Hawaii, USA April 9-12, 2007

A DEPENDABLE SYSTEM CONSIDERING INTERDEPENDENCE BETWEEN AGENTS

Keinosuke Matsumoto, Tomoaki Maruo, Akifumi Tanimoto, Naoki Mori
Graduate School of Engineering, Osaka Prefecture University
1-1 Gakuen-cho, Nakaku
Sakai, Osaka 599-8531, Japan
+81-72-254-9272
matsu@cs.osakafu-u.ac.jp

Abstract - A multiagent system (MAS) has recently gained public attention as a method to solve competition and cooperation in distributed systems. However, MAS's vulnerability due to the propagation of failure prevents from applying to large-scale system. This paper proposes a general composition method to improve its reliability easily applied to existent MASs. The method monitors messages between agents to detect undesirable behaviors such as failure. Collecting the information, the method generates global information of interdependence between agents and expresses it in a graph. This interdependence graph enables us to detect or predict undesirable behaviors. This paper also shows that the method can optimize performance of MAS and improve its reliability under complicated and dynamic environment by applying the global information acquired from the interdependence graph to a replication system.

INTRODUCTION

A multiagent system (MAS) [1] has recently gained public attention as a method to solve competition and cooperation in distributed systems. However, MAS's vulnerability due to the propagation of failure prevents from applying to a large-scale system. It is indispensable for constructing ubiquitous network society to raise a large-scale MAS's reliability.

Then, this paper proposes a general composition method to improve its reliability easily applied to existent MASs. The method monitors messages between agents to detect undesirable behaviors such as failure. Collecting and reserving the information, it generates global information of interdependence between agents and expresses it in a graph. This interdependence graph enables us to detect or predict undesirable behaviors. The method has been applied to an e-market MAS that deals with electronic dealings. Simulation results show that the method can optimize performance of MASs and improve reliability under complicated and dynamic environment by applying the global information acquired from the interdependence graph to replication systems [2].

Replication systems may arrange multiple replicas with the same contents on a network, and it is not necessary to change the code of the existing system drastically. They are regarded as an effective technique for raising fault tolerance of distributed systems. Such systems have replicas replaced with faulty agents in order to realize fault tolerant applications. However, they are not suitable for large-scale systems because replication cost may increase in proportion to the number of replicas. Therefore, this paper also proposes an adaptive replication system and replication policies that can reduce efficiently replication cost and raise performance.

PROBLEMS OF CONVENTIONAL MONITORING METHODS

Monitoring is usually adopted as a technique for raising reliability of MAS that includes undesirable behaviors like failure. Some techniques [3] – [5] are proposed and problems of these techniques are shown below: An immunity network [3] depends on knowledge agents have locally. It is a self-diagnostic system, and it becomes very complicated because fault tolerance and performance may be dependent on network

configuration. Moreover, a method of Kaminka et al. [4] is based on a recognition model of procedure to identify disagreement of states. As the procedure is static and premised on closed systems, this approach cannot be adapted for changing environments. Finally, a method of Horling et al. [5] is a diagnostic system of distributed systems that uses certain fixed failure models. It can optimize system's performance locally, but not globally. As a result, this approach is not suitable for the multiagent system under open environments where agents are added or deleted on large scale and in real time.

MONITORING METHOD USING INTERDEPENDENCE GRAPH

In order to solve the problems stated in the previous chapter, this chapter describes a monitoring method that enables us to analysis MAS in system level.

Interdependence Graph

Collecting related information, a monitoring system generates global information of interdependence between agents and expresses it in a graph. A domain agent is related to a node as shown in Figure 1. This situation is expressed as labeled graph (N, L, W). It is called an interdependence graph. Where, N is a node set, Li,j is a link from node Ni to Nj, and Wi,j is a weight labeled to a link Li,j. Wi,j corresponds to importance of the interdependence between agent i and j, and n is total number of nodes. If domain agents are added or deleted, the graph is updated dynamically. This interdependence graph enables us to detect or predict undesirable behaviors.

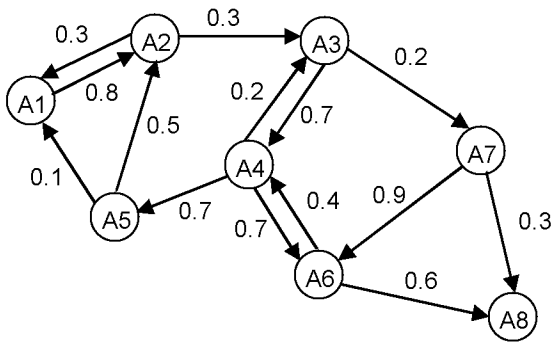

Figure 1. Interdependence Graph

$$N = \{Ni\}\ i=1,\ldots,n \quad (1)$$

$$L = \{Li,j\}\ i=1,\ldots,n,\ j=1,\ldots,n \quad (2)$$

$$W = \{Wi,j\}\ i=1,\ldots,n,\ j=1,\ldots,n \quad (3)$$

Monitoring Architecture

Monitoring task consists of two processes: acquisition of the information for updating interdependence graph, and the graph analysis for controlling domain agents. This information is standard indicators like communication load and processing time, agents' characteristics and so on. Conventional systems use centralized information collecting mechanisms, and they correct or analyze behavior off line. These centralized monitoring architectures are not suitable for large-scale and complicated systems because they cannot deal with changing environments in real time.

This section describes a distributed monitoring architecture to raise efficiency and reliability. The architecture is shown in Figure 2. This distributed observation mechanism corresponds to the organization of agents that react in adaptation to changing environments. The monitoring system has the following two roles:

- Observing and controlling domain agents
- Generating global information

These two roles are assigned to two kinds of agents: monitoring agents and a host monitoring agent. A monitoring agent is associated with each domain agent, and a host monitoring agent is associated with each monitoring agent. The agents constituting this monitoring system make hierarchical structure. Each monitoring agent communicates only with one host monitor agent, and transmits local information acquired by monitoring. The monitoring agent collects the individual information as a monitor, and the host monitor agent makes it into global information such as the total number of exchanged information. The monitoring agent reflects various changes of the domain agent in the interdependence graph. Because agent environment changes in real time, this graph is not static, and it is updated dynamically if agents in the domain level are added or deleted.

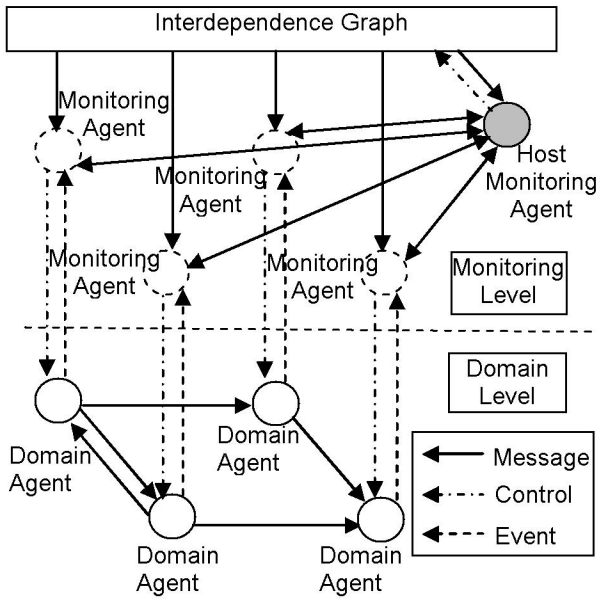

Figure 2. Monitoring Architecture

Updating Algorithm of Weights

Indexes for updating algorithm of weights are shown below:

- monitoring time interval Δt
- communication load Q(Δt)

$$Q(\Delta t) = \text{operator1}(Q_{1,1}(\Delta t), \ldots, Q_{n,n}(\Delta t)) \quad (4)$$

- the number of transmitting messages NM(Δt)

$$NM(\Delta t) = \text{operator2}(NM_{1,1}(\Delta t), \ldots, NM_{n,n}(\Delta t)) \quad (5)$$

Where $Q_{i,j}(\Delta t)$ and $NM_{i,j}(\Delta t)$ express respectively the amount of communication data and the number of transmitting messages from agent i to j in time interval Δt, operator1 and operator 2 are set operators and they are usually average operators.

The outline of the updating algorithm that updates the weights $W_{i,j}$ of the interdependence graph is shown below. This algorithm is performed by each monitoring agent to manage a related node.

1. Repeat the following for agent j ($j \neq i$).
2. Calculate formulas (6) and (7)

$$Qtemp = (Q_{i,j}(\Delta t) - Q(\Delta t))/Q(\Delta t) \quad (6)$$

$$NMtemp = (NM_{i,j}(\Delta t) - NM(\Delta t))/NM(\Delta t) \quad (7)$$

3. Updating weight $W_{i,j}$

$$W_{i,j}(t+\Delta t) = W_{i,j}(t) + \alpha \times \text{Operator3}(Qtemp, NMtemp) \quad (8)$$

4. End of repetition

Where, parameter α is a discount rate at which new situation is reflected to the existing weight. If the weights should react sensitively to environmental changes, this parameter is set up highly. On the other hand, if the data obtained experientially is regarded as important, this is set up low. This algorithm can be easily extended by changing the indexes.

ADAPTIVE MULTIAGENT ARCHITECTURE

This chapter shows a multiagent architecture for a replication system using the information the monitoring system collects.

Architecture

Our adaptive multiagent architecture is shown in Figure 3. This architecture consists of a monitoring system and a replication server that manage domain agents and replicas. The replication, which arranges replicas with the same contents on a network, is an effective technique for improving reliability of distributed systems. In our adaptive replication system, you can adjust dynamically the number of replicas according to each agent's importance obtained from the interdependence graph.

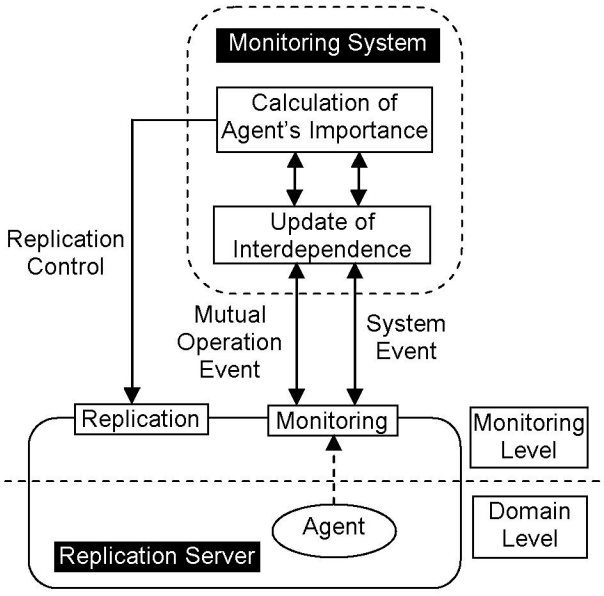

Figure 3. Adaptive Multiagent Architecture

Our simulations show that the system can optimize performance of a MAS and improve adaptively its reliability under complicated and dynamic environment by applying the global information acquired from the interdependence graph to a replication system.

Adaptive Replication System

We propose an adaptive replication system that changes dynamically its architecture according to environmental changes. The replication system creates some replication groups, and each group consists of one domain agent and some replicas. The algorithm adjusts adaptively the number of replicas in proportion to the importance of a node. If a leader replica in a replication group breaks down, the system is restored by changing another replica to a new leader replica.

The replication server that manages replication groups always performs the following processes: First, the server evaluates generated time of replicas, and computes the degree of coincidence of backup data. The replica with high degree of coincidence is chosen as a new leader replica if the leader replica of a domain breaks down. Second, the server generates or deletes replicas at the direction of the monitoring system.

An Algorithm to Get the Number of Replicas Using Interdependence

The interdependence graph is useful for grasping influence of failure and fault tolerance of multiagent systems. The proposed monitoring system estimates an agent's importance by carrying out set operation operator4 to the weights $W_{i,j}$ that are labeled to each agent's input-and-output links. It is shown in formula (9).

$$w_i = \text{operator4}\ (W_{i,j}\ j=1, ---, m) \quad (9)$$

Where, w_i is the importance of agent i and m is the degree of node i. The w_i is used to compute the number of replicas (rep_i) for agent i in our adaptive replication system. Formula (10) shows this relation.

$$rep_i = \text{round}\ [r_0 + r_{max} \times w_i / W] \quad (10)$$

Where, r_0 is an initial number of replicas, r_{max} is the maximum value of the total number of replicas a system designer sets up beforehand, and W is the sum of all agents' w_i.

The higher the importance of an agent is, the more replicas are preferentially assigned to the agent. It means that a strong system against failure of the important agents that affects the whole system can be obtained, and resource efficiency of system can be also attained at the same time.

SIMULATION EXPERIMENT

This chapter describes a simulation experiment that verifies validity of the proposed techniques.

Virtual eMarket Multiagent System

The simulation is carried out for a virtual eMarket multiagent system (eMarket MAS):

1. Parts purchasing agents belonging to product factories purchase parts from parts sales agents belonging to parts factories.
2. Product factories process the purchased parts and make products.
3. Product sales agent belonging to product factories sell off products to retailer agents.

A model of two stage market used in the simulation is shown in Figure 4. In this model, there are two markets:

- Market A: Dealings between parts factories and product factories
- Market B: Dealings between product factories and retailers

Furthermore, agents constituting markets are classified into the following four kinds of agents according to their roles: Sales agents, purchasing agents, purchasing/sales agents, and market

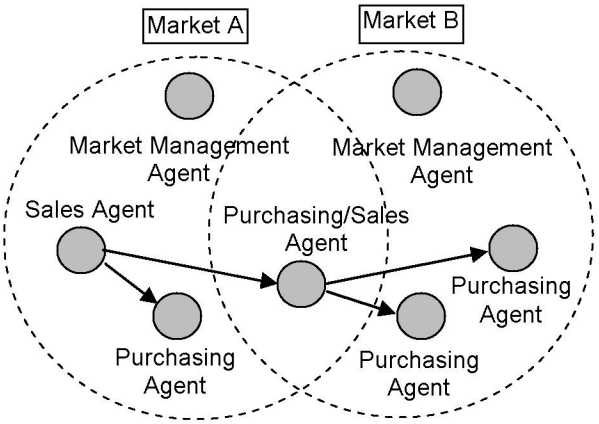

Figure 4. Two Stage Market Model

management agents. Sales and purchasing agents only sell parts or purchase products respectively. The purchasing/sales agents play different roles in each market. Finally, one market management agent exists in each market, and it controls registration or deletion of agents constituting markets. The market management agents mediate dealings performed in auction.

Table 1 shows the specifications of the computers used for the experiment and running applications. Replication server, eMarket MAS, and fault generator run on the three machines connected to Ethernet. The monitoring system runs only on Machine A.

The markets deal with dealings by auctions, and the scenario is described below:

1. A sales agent gives a market management agent minimum sales price.
2. The market management agent accepts bids from purchasing agents during certain fixed time.
3. The market management agent cuts a deal with the purchasing agent that presented the highest bid price exceeding the minimum sales price in a bid deadline.
4. If there is no corresponding bid, dealings are abortive and the sales agent presents again the price that is less than the previous one.

Uniform random numbers determine timings of price presenting and bidding, and the sales price and bid prices. The agents determined with the uniform random numbers are put to stop by the fault generators.

Table 1. Specifications of Each Machine and Running Systems

	Specifications	Running Systems
Machine A	Intel Pentium4 3.4GHz Memory 1GB	Replication Server Monitoring System eMarket MAS Fault Generator
Machine B	AMD Athlon 1.2GHz Memory 512MB	Replication Server eMarket MAS Fault Generator
Machine C	Intel Celeron 1GHz Memory 256MB	Replication Server eMarket MAS Fault Generator
Network	100Base-T Ethernet	

Experiment

Purpose of this experiment is to show relation between total number of replicas r_{max} and reliability of the system. Fault generators put a total of 100 agents to stop within 10-minute simulations. If an agent exhausts replicas prepared beforehand, the dealing scenario cannot be completed. This means the simulation is failure. We count the number of successful simulations when we change r_{max} from 0 to 30. Simulations are performed 40 times in total, and evaluated by rate of success. Parameters used for the experiment are shown in Table 2.

The experiment results are shown in Figure 5. When r_{max} is set up as 10, rate of success reaches 80%. If r_{max} is set as 20 and over, the success rate becomes 100%. It proves that reliability of the system is maintainable by setting r_{max} as 20 and over in this experiment environment, and also means that replication cost can be held down efficiently. The parameter r_{max} is important for systems and must be set up suitably.

Table 2. Parameters Applied to the Experiment

Parameters	Values
(Market Management Agents) (Purchasing&Sales Agents) (Purchasing Agents) (Sales Agents) Total	2 50 24 24 100
Monitoring Interval	500ms
Discount Rate α	1.0
Initial Number of Replica r0	1

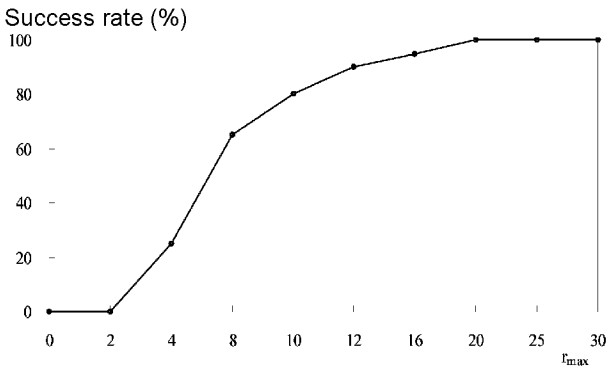

Figure 5. Relationship between r_{max} and Success Rate

CONCLUSIONS

This paper has proposed a global monitoring technique that can carry out analyzing systems in system level by introducing the interdependence graph which expresses the interdependence between agents in large multiagent systems. In addition, it also has proposed an adaptive replication system that uses global information acquired by monitoring to improve fault tolerance of multiagent systems.

To verify the approach, the proposed technique has been applied for eMarket MAS that expresses a virtual multiagent market system. This technique would give full scope to its ability by applying it to MAS consisting of almost equal and homogeneous agents and supervising agents' behaviors.

Followings are mentioned as future subjects.

- extension of set operators used for updating algorithm of weights
- extension of local information used for creating interdependence graphs
- dynamic changeover technique for replication policies in replication groups

REFERENCES

[1] G. Weiss, *Multiagent Systems -A Modern Approach to Distributed Artificial Intelligence-*. The MIT Press, pp.79-120 (1999).

[2] K. Hagihara, Algorithms for Fault-Tolerant Distributed Systems. Information Processing Society of Japan Magazine, Vol.34, No.11, pp.1336-1340 (1993) (in Japanese).

[3] Y. Ishida, An Immune Network Approach to Sensor-based Diagnosis by Self-organization. Complex Systems, Vol.10, No.1, pp.73-90 (1996).

[4] G. A. Kaminka, D. V. Pynadath, M. Tambe, Monitoring Teams by Overhearing: A Muti-agent Plan-Recognition Approach. Intelligence Artificial Research, Vol.17, pp.83-135 (2002).

[5] B. Horling, B. Benyo, V. Lesser, Using Self-Diagnosis to Adapt Organizational Structures. Proc. of 5th International Conference on Autonomous Agents, pp.529-536 (2001).