Biologically-Inspired Design of Autonomous and Adaptive Grid Services

Chonho Lee and Junichi Suzuki
Department of Computer Science
University of Massachusetts, Boston
{chonho, jxs} @ cs.umb.edu

Abstract—*This paper describes and evaluates a biologically-inspired network architecture that allows grid services to autonomously adapt to dynamic environment changes in the network. Based on the observation that the immune system has elegantly achieved autonomous adaptation, the proposed mechanism, the iNet artificial immune system, is designed after the mechanisms behind how the immune system detects antigens (e.g., viruses) and specifically reacts to them. iNet models a set of environment conditions (e.g., network traffic and resource availability) as an antigen and a behavior of grid services (e.g., migration and replication) as an antibody. iNet allows each grid service to autonomously sense its surrounding environment conditions (an antigen) to evaluate whether it adapts well to the sensed conditions, and if it does not, adaptively perform a behavior (an antibody) suitable for the sensed conditions. Simulation results show that iNet allows grid services to autonomously adapt their population and location to environmental changes for improving their performance (e.g., response time and throughput) and balancing workload.*

1. Introduction

Grid computing applications are expected to be *autonomous* and *adaptive* to dynamic environmental changes (e.g., workload surges) in order to improve user experience, expand application's operational longevity and reduce maintenance cost [1, 2, 3]. As inspiration for a new design paradigm for grid applications, we observe various biological systems have elegantly achieved autonomy and adaptability. We believe if grid applications are designed after certain biological concepts and mechanisms, they may be able to attain autonomy and adaptability.

BEYOND[1] is a network architecture that applies key biological mechanisms to design autonomous and adaptive grid applications. In BEYOND, a grid application is modeled as a decentralized group of software agents. This is analogous to a bee colony (an application) consisting of multiple bees (agents). Each agent implements a functional service related to the application and follows biological behaviors such as migration, replication, energy exchange and death.

This paper focuses on an adaptation mechanism for agents (i.e., grid applications) to autonomously adapt to dynamic environmental changes. The proposed adaptation mechanism, iNet, is designed after the mechanisms behind how the immune system detects antigens (e.g., viruses) and produces specific antibodies to kill them. iNet models a set of environment conditions (e.g., network traffic) as an antigen and a behavior of agents as an antibody. iNet allows each agent to autonomously sense its local environment conditions (an antigen) to evaluate whether it adapts well to the sensed conditions, and if it does not, adaptively perform a behavior (an antibody) suitable for the conditions. For example, agents may invoke migration behavior for moving towards network hosts that accept a large number of user requests for their services. This leads to the adaptation of agent locations, and agents can reduce response time for users. Simulation results show iNet allows agents to autonomously adapt to dynamic environmental changes for improving their performance (e.g., response time and throughput) and balancing workload. This paper is organized as follows: Section 2 overviews the design principles of BEYOND. Section 3 describes the design of iNet. Section 4 shows simulation results. Sections 5 and 6 conclude with comparison with related work.

2. BEYOND

In BEYOND, agents are designed based on the four principles described below.

Decentralization**:** Agents are decentralized. There are no central entities to control and coordinate agents (i.e. no directory servers and no resource managers). Decentralization allows grid services to be scalable and simple by avoiding performance bottleneck and any central coordination in deploying them [4, 5].

Autonomy**:** Agents are autonomous. Agents monitor their local network environments and autonomously behave and interact without any interventions from/to other agents and human users.

Adaptability**:** Agents are adaptive to dynamic environment conditions (e.g., user demands and resource availability). Each agent contains iNet, which allows it to adaptively behave against the current environment conditions (Figure 1).

Symbiosis**:** Agents are grouped as species depending on the services they provide. Different species (groups of agents) usually compete with each other for resources such as CPU cycles and memory space.

[1] Biologically-Enhanced sYstem architecture beyond Ordinary Network Design

However, in some circumstances, they share available resources to live together in a cooperative manner.

Agents run (or live) on a middleware platform in a network host. Each platform provides a set of runtime services that agents use to perform their services and behaviors.

Each agent consists of *attributes*, *body* and *behaviors*. Attributes carry descriptive information regarding the agent (e.g., agent ID). The body implements a service the agent provides. For example, an agent may implement a genetic algorithm for an optimization problem, while another agent may implement a physical model for scientific simulations. Behaviors implement non-service related actions inherent to all agents. This paper focuses on the following six behaviors.

Migration: Agents may move between platforms.

Energy exchange and storage: Agents may receive and store *energy* in exchange for providing services to other agents or users. Agents may also expend energy for services that they receive from other agents, and for resources available on a platform (e.g. memory space).

Communication: Agents may communicate with other agents for the purposes of, for example, requesting a service or exchanging energy.

Replications: Agents may make their copies in response to higher energy level, which indicates higher demand for the agents. A replicated agent is placed on the platform that its parent agent resides on, and it receives the half amount of the parent's energy level.

Death: Agents die due to energy starvation. If energy expenditure of an agent is not balanced with energy gain, the agent cannot pay for the resources it needs; thus, it dies from lack of energy. When an agent dies, an underlying platform removes the agent and releases all resources allocated to the agent.

Swapping: Different groups of agents (species) may cooperatively separate their locations (habitats) in the network when a group receives much larger number of service requests than another group. Agents in higher demand can ask other agents in lower demand to swap their locations. As a result, agents in higher demand can preferentially migrate towards users or certain resources; thereby increasing their performance (e.g., response time for users and throughput).

3. The iNet Artificial Immune System

This section overviews how the natural immune system works (Section 3.1), and describes how the iNet artificial immune system is designed after the natural immune system (Section 3.2).

3.1. Natural Immune System

The immune system is an adaptive defense mechanism to regulate the body against dynamic environment changes (e.g. antigen invasions). Through a number of interactions among various white blood cells (e.g. macrophages and lymphocytes) and molecules (e.g. antibodies), the immune system evokes two responses to antigens: *innate* and *adaptive* immune response.

In the innate immune response, the immune system performs *self/non-self discrimination* to detect antigens. This response is initiated by macrophages and T-cells, a type of lymphocytes. Macrophages move around the body to ingest antigens and present them to T-cells so that T-cells can recognize them. T-cells are produced in thymus and trained through the *negative selection process*. In this process, thymus removes T-cells that react with the body's own (self) cells. The remaining T-cells are used as detectors to identify non-self cells (i.e. antigens). When T-cells detect non-self cells, they secrete chemical signals to activate the second immune response: adaptive immune response.

In the adaptive immune response, the immune system produces antibodies that specifically react and kill an antigen identified by T-cells. Antibodies form a network structure and communicate with each other [6]. This network is formed with stimulation and suppression relationships among antibodies. Thus, the adaptive immune response is offered by multiple types of antibodies, although a single type of antibody (the best matched with an antigen) may play the dominant role. The immune network also helps to keep the quantitative balance of antibodies. Through the stimulation and suppression interactions, the population of specific antibodies rapidly increases following the recognition of an antigen and, after eliminating the antigen, decreases again. Performed based on this self-regulation mechanism, the adaptive immune response is an emergent product from many interactions among antibodies.

3.2. Design and Implementation of iNet

The iNet artificial immune system consists of the environment evaluation (EE) facility and *behavior selection* (BS) facility (Figure 1) corresponding to the innate and adaptive immune response, respectively. EE allows each agent to continuously sense a set of current environment conditions as an antigen and examine

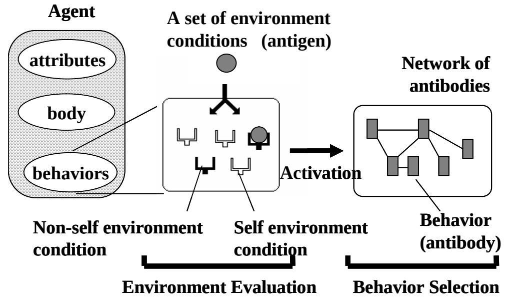


Figure 1. Organization of the iNet artificial immune system

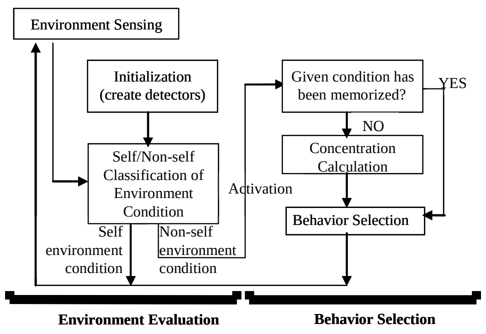

Environment Sensing
Initialization (create detectors)
Self/Non-self Classification of Environment Condition
Given condition has been memorized?
YES
NO
Concentration Calculation
Activation
Behavior Selection
Self environment condition
Non-self environment condition
Environment Evaluation
Behavior Selection

Figure 2. iNet Adaptation Process

whether it is self or non-self. A self antigen indicates that the agent adapts to the current environment conditions well, and a non-self antigen indicates it does not. When EE detects a non-self antigen, EE activates BS (Figure 1). BS allows each agent to choose a behavior as an antibody that specifically matches with the detected non-self antigen.

3.2.1. Environment Evaluation Facility (EE)

EE performs two steps: initialization and self/non-self classification (Figure 2). The initialization step produces detectors that identify self and non-self antigens (i.e. environment conditions). In iNet, an antigen (i.e. a set of environment conditions) is implemented as a *feature vector*. Each feature vector (*X*) consists of a set of features (*F*) and a class value (*C*). *F* contains a series of environment conditions. If an agent senses agent population on a local platform, resource utilization on a local platform and workload (the number of user requests) on a local platform, a feature vector may be represented such as X_{curent}*=((Low: Agent population, Low: Resource utilization, Heavy: Workload), C). C* indicates whether a given antigen (i.e. a set of environment conditions) is self (0) or non-self (1).

To evaluate whether an antigen (i.e. feature vector) is self or non-self, the initialization step produces detectors that identify it (Figure 2). This step is designed after the negative selection process in the immune system. In the initialization step, EE first generates feature vectors randomly, and separates them into *self detectors*, which closely match with self antigens (feature vectors), and *non-self detectors* (T-cells in the immune system), which do not closely match with self antigens (feature vectors). This separation is performed via vector matching between randomly generated feature vectors and self antigens that human users supply (Figure 3). Currently, EE uses the Euclidean vector matching algorithm. After vector matching, both self and non-self detectors are stored in a feature table (Figure 3)[2].

[2] The immune system removes non-self detectors through negative selection process. However, in iNet, both self and non-self detectors are kept in a feature table to perform self/non-self classification.

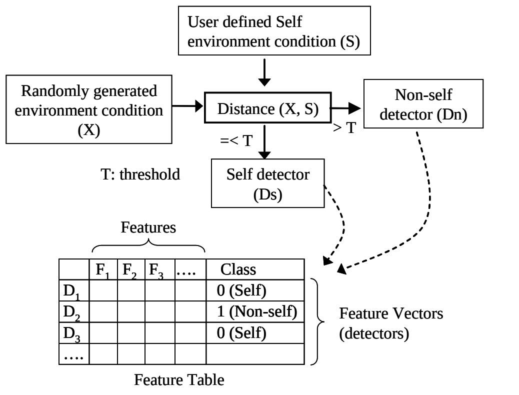


Figure 3. Initialization Step in EE

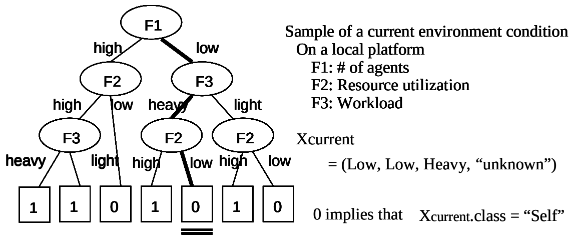


Figure 4. An Example Decision Tree

The second step in EE performs self/non-self classification of environment conditions (Figure 2). It uses the detectors in a feature table to classify the current environment conditions into self or non-self. The self/non-self classification step is performed with a decision tree built from detectors in a feature table. Figure 4 shows an example decision tree. EE starts to examine a set of given current environment conditions, $X_{current}$, at the root of the decision tree. Each node in the tree specifies which feature is considered. Based on the value of the specified feature in $X_{current}$, EE follows down along the branch indicating the value. This process is repeated until EE reaches at the leaf of tree which notices the class value of $X_{current}$. Once EE detects a non-self antigen, it activates BS immediately.

The reasons for using decision tree as a classifier are ease of implementation and algorithmic efficiency. Since a decision tree is easy to understand and implement, iNet can maintain a lower barrier for developers to design adaptive grid applications. Also, a decision tree performs classification much faster than other algorithms such as clustering, support vector machine and Markov model algorithms [7, 8]. The efficiency of classification is one of the most important requirements in iNet because each agent periodically senses and classifies its surrounding environment conditions.

3.2.2. Behavior selection facility (BS)

Once EE classifies the current environment conditions as a non-self antigen, it activates BS. BS selects an antibody (i.e. agent's behavior) suitable for the detected non-self antigen (i.e. environment conditions).

Each antibody is structured as shown in Figure 5. It consists of *Paratope*, precondition under which it is selected (one of environment conditions), *Behavior ID*, one of agent behaviors, and *Idiotope*, relationships to other antibodies (one or more links). Antibodies are linked with each other using stimulation and suppression relationships (see Section 3.1). Each antibody has its own concentration value corresponding to the number of the antibody. The value is used to prioritize antibodies (behaviors). BS identifies candidate antibodies (behaviors) suitable for a given non-self antigen (environment conditions), prioritizes them based on their concentration values, and selects the most suitable one from the candidates. When prioritizing antibodies (behaviors), stimulation relationships between them contribute to increase their concentration values, and suppression relationships contribute to decrease it. Each relationship has its own strength (affinity), which indicates the degree of stimulation or suppression.

Figure 6 shows a generalized network of antibodies.

Antibody

Precondition under which this antibody is selected	Agent Behavior ID	Relationships to other antibodies (behaviors)
Paratope	Behavior	Idiotope

Figure 5. Antibody Structure

The antibody *i* stimulates *M* antibodies and suppresses *N* antibodies. m_{ji} and m_{ik} denote affinity values between antibody *j* and *i*, and between antibody *i* and *k*. m_i is an affinity value between an antigen and antibody *i*. The concentration of antibody *i*, denoted by a_i, is calculated with the following equations.

$$\frac{dA_i(t)}{dt} = \left(\frac{1}{N} \sum_{j=1}^{N} m_{ji} \cdot a_j(t) - \frac{1}{M} \sum_{k=1}^{M} m_{ik} \cdot a_k(t) + m_i - k \right) a_i(t) \ldots (1)$$

$$a_i(t) = \frac{1}{1 + \exp(0.5 - A_i(t))} \ldots\ldots (2)$$

In the equation (1), the first and second terms in a big bracket denote the stimulation and suppression from other antibodies. The affinity values between antibodies (i.e. m_{ji} and m_{ik}) are positive between 0 and 1. m_i is 1 when antibody *i* is stimulated directly by an antigen, otherwise 0. *k* denotes the dissipation factor representing the natural death of an antibody. This value is 0.1. The initial concentration value for every antibody, $a_i(0)$, is 0.01. The equation (2) is a sigmoid function used to squash the $A_i(t)$ value between 0 and 1.

Every antibody's concentration is calculated 200 times repeatedly. This repeat count is obtained from a previous simulation experience [9, 10]. If no antibody exceeds a predefined threshold (0.7) during the 200

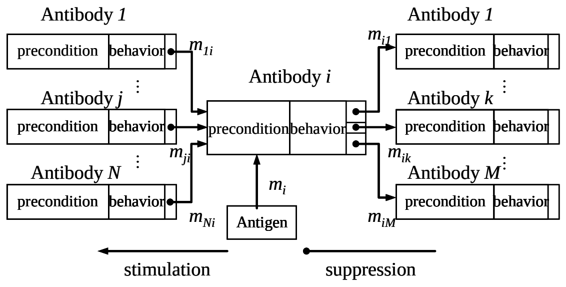
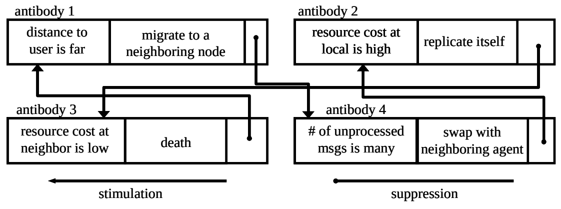


Figure 6. A Generalized Network of Antibodies



antibody 1
distance to user is far
migrate to a neighboring node
antibody 2
resource cost at local is high
replicate itself
antibody 3
resource cost at neighbor is low
death
antibody 4
# of unprocessed msgs is many
swap with neighboring agent
stimulation
suppression

Figure 7. An Example Network of Antibodies

calculation steps, the antibody whose concentration value is the highest is selected (i.e. winner-tales-all selection). If one or more antibodies' concentration values exceed the threshold, an antibody is selected based on the probability proportional to the current concentrations (i.e. roulette-wheel selection).

Figure 7 shows an example network of antibodies. It contains four antibodies, which represent migration, replication, swapping and death behaviors. Antibody 1 represents the migration behavior invoked when distance to users is far. Antibody 1 suppresses Antibody 3 when it is stimulated Now, suppose that a (non-self) antigen indicates (1) user location is far, (2) network traffic is heavy on the local platform and (3) resource availability is high at local platform. This antigen stimulates Antibodies 1, 2 and 4 simultaneously. Their population increases, and it is likely that Antibody 2's concentration value becomes highest because Antibody 2 suppresses Antibody 4, which in turn suppresses Antibody 1. As a result, Antibody 2 (i.e. replication behavior) would be selected.

4. Simulation Results

This section shows simulation results to examine the autonomous adaptability of agents developed with iNet. The simulations are carried out on the BEYOND simulator[3]. Figure 8 shows a simulated network. A server farm consists of network hosts connected in a 15 x 15 grid topology, and service requests travel from users to agents via user access point. This simulation study assumes that a single (emulated) user runs on the access point and sends service requests to agents.

[3] The BEYOND simulator contains 13,490 lines of Java code. It is available for researchers who investigate autonomous and adaptive grid applications (http://dssg.cs.umb.edu).

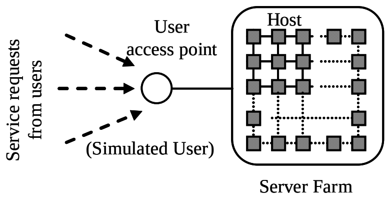


Figure 8. Simulated Network

When a user issues a service request, the service request is passed to the local platform where the user resides on, and the platform performs a discovery process to search a target agent that can process the issued service request. The platform (discovery originator) forwards a discovery message to its neighboring platforms, asking whether they host a target agent. If a neighboring platform hosts a target agent, it returns a discovery response to the discovery originator. Otherwise, it forwards the discovery message again to its neighboring platforms. Figure 9 shows pseudo code for this agent discovery through platform connectivity[4].

```
While (not simulation last cycle)
   If ( Discovery messages arrived)
      For each of discovery msgs (under the max # of messages to be
            processed in each simulation cycle) Do
            If ( Discovery message matches one of the local agents)
                Returns a discovery response to discovery originator
            Else
                Forward the discovery message to neighboring platforms
            End If
      End For
   End If
End While
```

Figure 9. Pseudo Code for Agent Discovery Process in each Simulation Cycle

4.1. Local-Area Grid

In the first simulation study, each agent implements web server functionality in its body and possesses an iNet immune network that implements three behaviors (migration, replication and death), five environment conditions (energy level, workload on the local/neighboring platforms, and resource availability on the local/neighboring platforms), 10 antibodies and 7 stimulation/suppression relationships. An assumed application in this simulation study is web servers hosted on a server cluster. At the beginning of simulation, a single agent is deployed on a platform at the right bottom corner of server farm (Figure 8).

Figure 10 (1) (All figures are shown in the last page.) shows how service request rate changes over time. It starts with 1000 requests/min, spikes to 60,000 requests/min at 2:00, and drops to 1,000 requests/min at 5:00. The peak-to-trough ratio is 60:1. This ratio is designed based on a workload trace of the 1998 World Cup official web site [16].

[4] Note that there is no centralized directory to keep track of agents.

Figure 10 (2) shows how agents autonomously

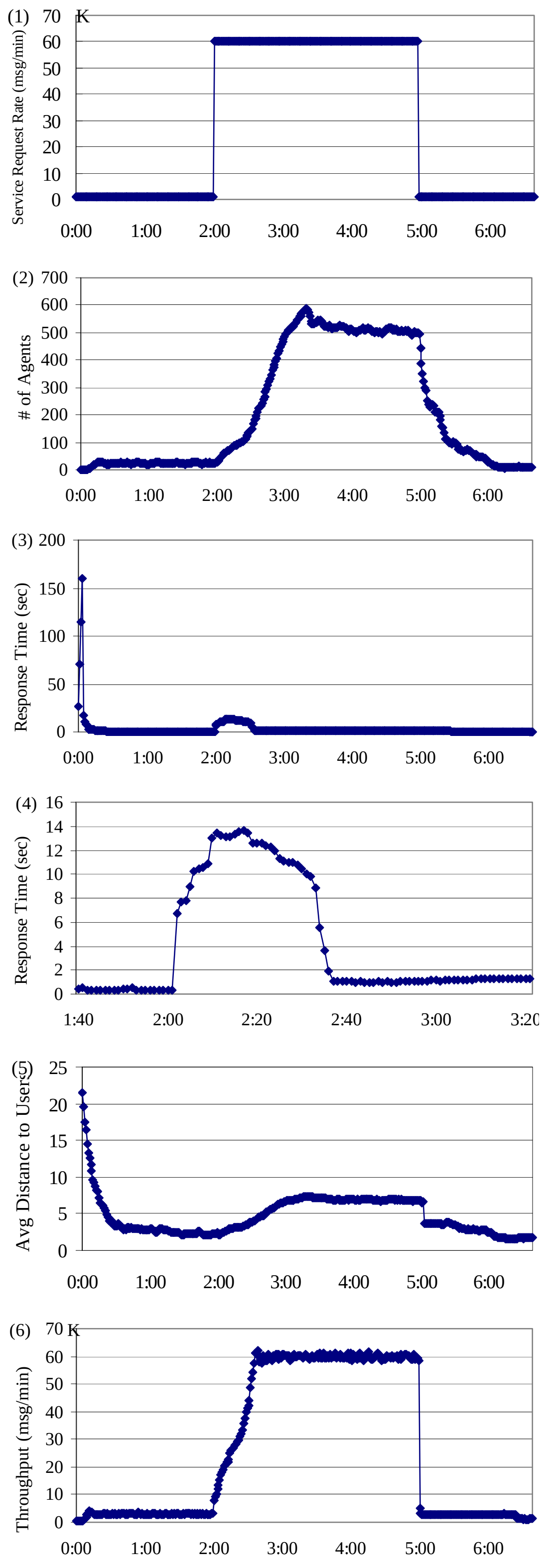


Figure 10. Simulation results in 15x15 network

adapt their population to workload changes. When the workload spikes, agents gains more energy from users and perform replication behavior to increase their population. On the other hand, when the workload drops, some agents die due to energy starvation because they cannot balance their energy gain from users and energy expenditure to utilize the resources. As a result, agents shrink their population by eliminating idle ones.

Figure 10 (3) shows how agents autonomously adapt response time for a user. At the beginning of simulation, response time becomes very high because only one agent processes 1,000 requests a minute and distance between the agent and users is long. However, after the agents accumulate enough energy from users and start migrating towards users and replicating themselves, they rapidly decrease response time (less than a second). When workload spikes at 2:00, the response time spikes up to 14 seconds (Figure 10 (4)), but agents decrease it to 1 second in 20 minutes by performing replication behavior according to the demand change.

Figure 10 (5) depicts the average distance between agents and users. Initially, a single agent is placed at the right bottom corner, and users are at the left most (Figure 8). Therefore, the initial distance between agents and users is 22. Agents autonomously decrease the distance by performing migration behavior.

Figure 10 (6) shows the throughput achieved by agents. It is measured as the number of responses that users receives a minute from agents. Figure 9 demonstrates that agents autonomously meet given workload by dynamically adjusting their population and locations through replication and migration behaviors.

4.2. Wide-Area Grid

The second simulation study is carried out to evaluate how agents autonomously balance workload placed on them and how symbiosis among agents (swapping behavior) impact agent performance. In this study, each agent implements multimedia streaming function in its body and possesses an iNet immune network that implements three behaviors (migration, replication, death and swapping), five environment conditions (energy level, workload on the local/neighboring platforms, and resource availability on the local/neighboring platforms), 13 antibodies and 10 stimulation/suppression relationships.

An assumed application in this simulation study is multimedia content distribution in a grid (a collection of servers) that covers a geographically wide area. For example, a large sport event such as Olympic game may deploy servers at multiple event sites such as athletics stadiums, swimming pools and gymnasiums. The servers are connected with each other to form a wide-area grid. Each event site has a wireless base station connected to this grid, and spectators in the event site can access realtime zoom-in video or playback video. This simulation study assumes that 15x15 (225) servers are deployed, each agent contain a multimedia steam (or a fragment of the stream), and user access point is a wireless base station through which users access agents for multimedia contents.

In this simulation, there are two types of agents that provide different types of multimedia contents: A and B. At the beginning of simulation, three agents for each service A and B are randomly deployed in the network. Figure 11 shows dynamic changes of service request rate for service A and B. Service A is constantly requested at the rate of 25K requests/min from 0:00 to 2:00. This simulates that spectators in an athletic stadium constantly request playback movie of an athletic match. At 2:00, service request rate spikes from 0 to 35K requests/min. This simulates that the spectators in an athletic stadium rapidly access the movie of a swimming final match that generated the world record.

Figure 12 shows how a swapping behavior effectively works in terms of response time for users. The black line indicates response time by agents with swapping behavior, and the gray line indicates response time by agents without swapping behavior. The first graph in Figure 11 shows that response time is

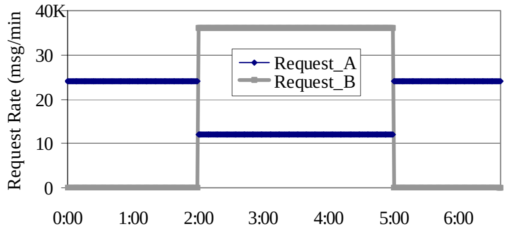


Figure 11. Service request rate

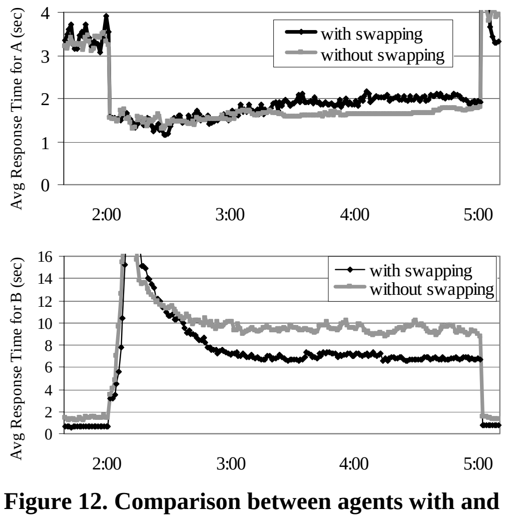


Figure 12. Comparison between agents with and without swapping behavior

slightly increasing (about 0.4sec) because agents for service A yield their resources to agents for service B. However, the second graph in Figure 11 shows that response time drops about 4 sec since agents for service B gain more resources by swapping behavior. So, the total average response time also reduces. As presented in section 2, a swapping behavior is performed between two different types of agents. Agents compare their workload with others, and if the workload is heavier then they try to swap their location and gain resources closer to users.

Figure 13 shows that how agents dynamically distribute their workload over the simulated grid network. Load Balancing Index (LBI) is measured with

$$Load \quad Balancing \quad Index = \frac{\sum_i (X_i - \mu)^2}{N} \quad \ldots.. \quad (3)$$

X_i indicates the current workload of platform i per a resource unit (i.e. the # of messages processed by agents running on platform i divided by resource utilization on platform i). μ represents the expected average of X (i.e. the total # of messages processed by all agents divided by the total amount of resource utilization on all platforms N where those agents are running.) Hence, *LBI* represents a variance of workload on each platform. Figure 13 shows that *LBI* is decreasing from 2:00 to 3:00. This indicates that, when agents receive service B requests at 2:00, agents start replicating, and the workload is being distributed over the network.

Figure 14 describes how each group of agents (i.e. agents for service A and B) forms a cluster when they migrating toward users. Figure 14 (A) shows the cluster size of agents for service A. At 0:00, three agents are deployed and form a cluster with size 10. When they receive requests from users, they migrate toward users and reduce cluster size to about 2.5. Similarly, figure 14 (B) shows the cluster size of agents for service B. When they receive requests at 2:00, they start migrating toward users from up, right and bottom (due to initial location). At 2:15, they can no longer migrate (with cluster size 8) because of a boundary of agents for service A which are already consuming resources closer to users. However, agents for service B invoke swapping behavior and get close to each other (with cluster size 6) at 3:00. Swapping behavior helps to reduce cluster size of the same type of agents.

5. Related work

Artificial immune systems have been proposed and used in various application domains such as anomaly detection [11] and pattern recognition [12]. [11] mainly focuses on the generation of detectors for self/non-self classification and improves the negative selection process of the artificial immune system. [12] focuses on the accuracy for the matchmaking of an antigen and antibody. On the other hand, this paper proposes an artificial immune system to improve autonomous adaptability in grid network services. To the best of our knowledge, this work is the first attempt to apply an artificial immune system into the autonomic adaptation domain.

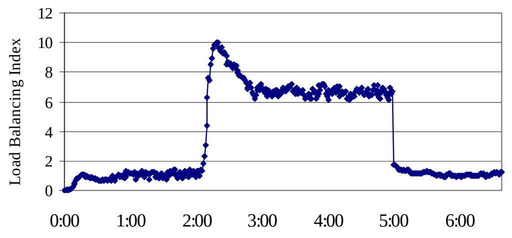


Figure 13. Load balancing

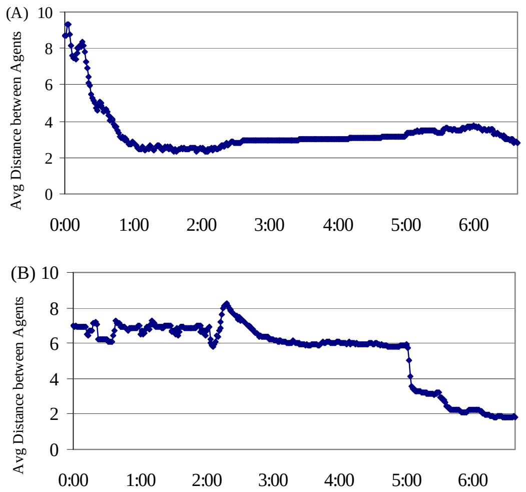


Figure 14. Distribution of agents for service A and service B

There are lots of research efforts to achieve decentralized network applications based on the concept of natural immune system. For example, [17] uses the concept of memory B-cells to reduce search response time. Once the target for query is found, B-cell (user node) remembers the query and the peers (nodes) who contain the target. They move close each other and form clusters so that reducing response time for the query encountered before. [18] follows the concept of stimulation/suppression interaction among antibodies to find the recommending items that have been never seen before by users. Stimulation affects to increase the concentration of similar items to a user profile as recommending items; suppression affects to keep items that have seen before by users far away from the recommending items. Similar to [17][18], iNet agent includes the concept of B-cells such as the network of antibodies and the memory of antibody's concentration in the BS facility. However, the iNet agent also includes the function of T-cells such as self/non-self discrimination in the EE facility. This additional function results in reducing the overhead of behavior selection.

Similar work has been proposed in Organic Grid [13] project. [13] attempts to the decentralized task scheduling for large-scale computation on grid environment over the Internet. Similar to iNet, mobile agents autonomously executes their services (e.g. computing subtasks) on the platform embedded in each host and perform their replication behavior to achieve their objectives (e.g. compute as fast as possible). Yet, iNet focuses on the grid service adaptation. Agents consider more behaviors for adaptation, and through those various adaptation decisions, the high adaptability of grid network services (i.e. agents) is achieved.

There are several research efforts that allow network systems to adapt to application and end-user requirements with a technique of runtime component replacement. For example, [14, 15] can dynamically replace running components (e.g. byte code) with others according to the monitored environment conditions. The difference is that [14, 15] assumes a centralized network architecture where a centralized server collects environment conditions from each component to make replacement decisions. In contrast, iNet assumes a decentralized network architecture where each agent monitors its surrounding environment and makes adaptation decisions.

6. Concluding Remarks

This paper describes and evaluates a biologically-inspired mechanism that allows grid services to autonomously adapt to dynamic changes in the network. The proposed adaptation mechanism, called the iNet artificial immune system, allows each grid service to autonomously sense its surrounding environment conditions to evaluate whether it adapts well to the conditions, and if it does not, adaptively perform a behavior suitable for the conditions. Simulation results show that iNet allows agents to autonomously adapt to dynamic environmental changes for improving their performance (e.g., response time and throughput) and balancing workload.

7. References

[1] P. Dini, W. Gentzsch, M. Potts, A. Clemm, M. Yousif and A. Polze, "Internet, Grid, Self-adaptability and Beyond: Are We Ready?," In *Proc. of the IEEE Int'l Workshop on Self-Adaptable and Autonomic Computing Systems*, Aug. 2004.

[2] Large Scale Networking Coordinating Group of the Interagency Working Group for Information Technology Research and Development (IWG/IT R&D), *Report of Workshop on New Visions for Large-scale Networks: Research and Applications*, Mar 2001.

[3] R. Sterritt and D. Bustard, "Towards an Autonomic Computing Environment," In *Proc. of 14th IEEE International Workshop on Database and Expert Systems Applications*, Sep 2003.

[4] N. Minar, K. H. Kramer and P. Maes, "Cooperating Mobile Agents for Dynamic Network Routing," In *A. Hayzelden and J. Bigham (eds.) Software Agents for Future Communications Systems*, Springer, 1999

[5] G. Cabri, L. Leonardi and F. Zambonelli, "Mobile-Agent Coordination Models for Internet Applications," In *IEEE Computer*, Feb 2000.

[6] N.K.Jerne, "Idiotypic Networks and Other Preconceived Ideas," In *Immunological Review*, vol. 79, 1984.

[7] P. Berkhin, "Survey of Clustering Data Mining Techniques," Accrue Software, Inc., San Jose, CA, 2002.

[8] T. Mitchell, *Machine Learning*, McGraw-Hill, 1997.

[9] J. Suzuki and Y. Yamamoto, "iNet: An Extensible Framework for Simulating Immune Network," In *Proc. of IEEE International Conference on Systems, Man, and Cybernetics*, October 2000.

[10] J. Suzuki and T. Suda, "Adaptive Behavior Selection of Autonomous Objects in the Bio-Networking Architecture," In *Proc. of 1st Annual Symposium on Autonomous Intelligent Networks and Systems*, May 2002.

[11] F. A. González, D. Dasgupta, "Anomaly Detection Using Real-Valued Negative Selection," *Genetic Programming and Evolvable Machines*, 4(4), 2003.

[12] L. N. de Castro, J. I. Timmis, "Artificial Immune Systems: A Novel Paradigm to Pattern Recognition," In *Artificial Neural Networks in Pattern Recognition*, J. M. Corchado, L. Alonso, and C. Fyfe (eds.), SOCO-2002, University of Paisley, UK, pp. 67-84.

[13] A.J. Chakravarti, G. Baumgartner, M. Lauria, "The Organic Grid: Self-organizing Computational Biology on Desktop Grid," In *A Zomaya, Parallel Computing for Bioinformatics*, John Wiley & Sons, 2005.

[14] S. Cheng, D. Garlan, B. Schmerl, P. Steenkiste, and N. Hu, "Software Architecture-based Adaptation for Grid Computing," In *the 11th IEEE Conference on High Performance Distributed Computing*, July 2002.

[15] K Shirose, S Matsuoka, H Nakada, and H Ogawa, "Autonomous Configuration of Grid Monitoring Systems," In *the 2004 Symposium on Application and the Inte*rnet (SAINT2004), Japan, January 2004.

[16] Martin Arlitt and Tai Jin, "A Workload Characterization Study of the 1998 World Cup Web Site," *IEEE Network*, Vol. 14, No. 3, pp. 30-37, May/June 2000.

[17] N. Ganguly, G. Canright and A. Deutsch, "Design of an Efficient Search Algorithm for P2P Networks Using Concepts from Natural Immune Systems," In *proc of Parallel Problem Solving from Nature 2004*, 491-500.

[18] S. Cayzer1 and U. Aickelin, "A Recommender System based on the Immune Network," In *proc of Congress on Evolutionary Computation 2002*, Honolulu, USA, 2002.