CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2007; **19**:1047–1064
Published online 28 September 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe.1095

Distribution of mobile agents in vulnerable networks

Wenyu Qu[1,*,†], Masaru Kitsuregawa[1], Hong Shen[2] and Yingwei Jin[3]

[1] *Institute of Industrial Science, The University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo 153-8505, Japan*
[2] *Graduate School of Information Science, Japan Advanced Institute of Science and Technology, 1-1 Asahidai, Tatsunokuchi, Ishikawa 923-1292, Japan*
[3] *Department of Computer Science and Engineering, Dalian University of Technology, 2 Linggong Road, Ganjingzi District, Dalian 116024, People's Republic of China*

SUMMARY

Advances in the Internet and the computer industry have created many new application areas for network routing such as Grid computing and also brings new challenges to traditional routing techniques. In this paper we propose a mobile agent-based routing model in vulnerable networks for these applications. To characterize the behaviors of mobile agents and their effects on the network performance, we analyze the population distribution of mobile agents as a measurement of the computational resource consumption. Our analysis reveals theoretical insights into the statistical behaviors of mobile agents and provides useful tools for effectively managing mobile agents in large networks. Copyright © 2006 John Wiley & Sons, Ltd.

Received 26 April 2006; Accepted 5 May 2006

KEY WORDS: routing; mobile agents; distribution; Markov process

1. INTRODUCTION

Routing is a key factor for network management. It is the process of moving a packet from the source node to the destination node over the network, which can be broadly classified into two activities, i.e. path determination and packet transmission. Although packet transmission is relatively straightforward, optimal path determination can be very complex. In this paper, we address the problem of path determination.

In a large, constantly changing network, routers communicate with each other and maintain their routing tables through the transmission of a variety of routing messages. The existing routing algorithms can be mainly categorized into two kinds, namely link state algorithms (also known as open shortest path first algorithms) and distance vector algorithm (also know as Bellman–Ford algorithms).

*Correspondence to: Wenyu Qu, Institute of Industrial Science, The University of Tokyo, 4-6-1 Komaba Meguro-ku, Tokyo 153-8505, Japan.
†E-mail: quwenyo@tkl.iis.u-tokyo.ac.jp

Both types of algorithm use flooding as the main method of message transmission [1]. Link state algorithms flood routing information to all nodes in the network to build a picture of the entire network in each node's routing table, while distance vector algorithms call for each node to send updating information to its neighborhood such that each node knows information about their neighbors. The continuous updating of the routing tables on each node consumes a large portion of the scarce network computational resource when exchanging large amounts of routing table data. This reduces the available capacity of the network for actual data communication.

Based on different kinds of routing information available in the routing table, the desired path for sending a packet is determined by either source routing or hop-to-hop routing [2]. In source routing, all the routing information is first collected at the source node, which puts it into the packets that it launches toward the destination node. The job of the intervening network is simply to read the routing information from the packets and act on it faithfully. While in hop-by-hop routing, the source node is not expected to have all the routing information; it is sufficient for the source node to know only how to get to the 'next hop' and for that node to know how to get the 'next hop', and so on until the destination is reached. The job of the intervening network in this case is more complicated; it has only the address of the destination with which to figure out the best 'next hop' for each packet. Hop-by-hop routing is used in OSI and TCP/IP networks. The performance of hop-by-hop routing depends greatly on the convergence speed to a network event. When a network event causes routes to either go down or become available, routers distribute routing update messages that permeate networks, stimulating recalculation of optimal routes and eventually causing all routers to agree on these routes. A slow convergence may cause routing loops or network outages.

With the explosive growth in popularity of the World Wide Web, locating Web content promptly is becoming increasingly important [3,4]. Meanwhile, the amazing advances in the Internet and the computer industry have created many new application areas for network routing such as Grid computing and P2P computing [5,6]. Grid computing was proposed in the mid 1990s owing to the increasing demand of large-scale scientific computation in the fields of life sciences, biology, physics, and astronomy [6]. Since a Grid connects numerous geographically distributed computers, and tasks are submitted to Grid nodes in a distributed fashion, important issues that have to be solved include routing, task scheduling, and resource allocation. To adopt and take advantage of the changes of today's Internet, many new network routing technologies have been proposed in recent years that have created exciting new opportunities for both business and science [7]. Mobile agents, decision-making programs that are capable of migrating autonomously from node to node in a computer network, are features of many of these new technologies [8,9]. The properties of this technology, such as autonomous, adaptive, reactive, mobile, cooperative, interactive, and delegated [10], have drawn a great deal of attention in both academia and industry. Mobile agents are continuously running processes with personalities and emotional states that are formalized by knowledge. They can play an important role in the high-level applications of the knowledge Grid [11]. In turn, the knowledge Grid technology can help the organization of knowledge of the agent and sharing of knowledge among agents.

An agent's actions is goal-orientated. It is capable of transporting itself from one execution environment to another, learning and improving with experience, choosing actions that can bring it closer to its goal, communicating with other agents or people, and acting together with other agents for a common purpose. It can sense its environment, reflect upon it and respond in a timely fashion to changes in its environment, acts accordingly, and take actions to achieve a goal. Based on these properties, using mobile agents can bring the following benefits. They can reduce network load, overcome network latency, encapsulate protocols, execute asynchronously and autonomously, and adapt changes dynamically. They are naturally heterogeneous, robust, and fault-tolerant [12]. Although none of these strength are unique to mobile agents, no competing technique shares all of them. Mobile agents provide a general framework in which distributed and information oriented applications can be implemented effectively and easily, with the programming burden spreads evenly among information, middleware, and client providers. Many mobile agent systems have become sufficiently well developed in recent years. For example, Ajanta [13], Bee-gent [14], D'agents [15], Odyssey [16], and so on. Successful examples using mobile agents can be found in [13–15]. In this paper, we propose an extended routing model based on our previous work [20]. We analyze the population distribution of mobile agents and compared our results with existing solutions.

In the following sections, we first introduce related works of our research, followed by the extended mobile agent-based routing algorithm, theoretical analysis, experimental results, and conclusion.

2. RELATED WORKS

Advanced research in mobile agents has brought in some new methods for network routing [21]. An ant-like agent-based routing algorithm is a recently proposed routing algorithm for use in large dynamic networks [22,23]. The idea is similar to the shortest path searching process during ants' foraging activities. In the classical ant routing algorithm [24], once a server receives a request for sending a packet to a remote destination node, it generates a number of mobile agents. These agents will then move out from the server and search for the destination in the network. They probabilistically prefer nodes that are connected immediately. Once a mobile agent finds the destination, it returns back to the server and submits a report of the searched path. After all agents have been back, the server will calculate the cost of those collected paths and pick up the best one. The packet will be sent to the destination along the optimal path and the routing table will be updated at the same time. The efficiency of the ant-like agent-based routing technique has been shown in [25]. It can be seen that mobile agents will be generated frequently and dispatched to the network. If there are too many mobile agents running in the network, they will consume too much computational resource, which will affect the network performance due to the limited network resource and ultimately block the entire network. Therefore, analysis of the distribution of mobile agents is not only important but necessary for network management. Unfortunately, little work has been done on this aspect. In [24], a death rate is assumed for the purpose of limiting the number of agents to within the system's limitations. In [26], the number of agents is analyzed for the classical ant routing algorithm that characterizes the evolution behavior of mobile agents and reflects the consumption of computational resources. In the previous version of this paper [20], an improved mobile agent-based routing model is proposed for use in vulnerable networks in which any component may fail during an agent's searching process while in [24] systems are assumed reliable. In this model, mobile agents are assumed to have a life-span limit instead of the death rate as assumed in [24] and the population distribution of the agents is analyzed. The analytical results show that this model has less population than that in [26]. In this paper, we extend the model in [20] by allowing agents on a node that can know routing information of not only the host node but also the neighboring nodes of the host node. This change makes agents' searching process much more efficient compared to that in [20,24]. We also analyze the population distribution of mobile agents.

The analytical results show that the number of agents is decided by the connectivity degree of the network, the possibility of a system failure at each step, the number of requests received per time unit, the number of mobile agents generated per request, and the agents' life-spans. Our model consumes less network resources due to a less number of agents in the network.

3. MOBILE AGENT-BASED ROUTING MODEL

Our model is built on a vulnerable distributed network whose topology may change dynamically due to failures of network components or terminals joining or leaving the network at any time as they like. In this paper, we assume that the topology of the network is a connected graph, i.e. each node in the network has at least one neighboring node. If there is a direct link between two nodes, these two nodes are called neighboring nodes. When a node cannot link to any other nodes in the network, we say this node is down. Nodes in the network periodically broadcasts their routing information to neighboring nodes. Suppose that each node provides mobile agents with an execution environment and can generate mobile agents to complete some tasks when necessary. A node that generates mobile agents is called the server of these agents. Once a request is received by a server, the server generates a number of mobile agents. These agents will then move out from the server to search for the destination. Once an agent reaches a node, it will check whether its destination is among the node and its neighboring nodes. If the agent cannot find its destination node locally, it will select a neighboring node to continue its searching activity. To avoid the explorer agents getting stuck in any terminal node, agents in this case can return back to the previous node from which it just left. After an agent finds its destination, it will turn back to the server and submit a routing report about the searched path. During an agent's searching process, it may be subject to a system failure as any component in the network may fail at any time. Note that routing applications dispatch multiple agents simultaneously for one single task to tolerate the loss of some dispatched agents compared with applications such as e-shopping in which only one agent is generated per request so that the agent should return a failure message to the user if it cannot complete its task. Therefore, if an agent is subject to a system, it will die. In an asynchronous distributed system (e.g., the Internet) there are no bounds on migration delays of messages and no relative process speeds. Thus, when a mobile agent is blocked because of a failure in an asynchronous distributed system, the agent's owner cannot correctly determine whether the agent has failed or is merely running slowly [27]. To release users from waiting for an excessive time period for his request, we set a bound of searching time, namely life-span limit, to mobile agents in our model by using timer technique. If an agent cannot find its destination within its life-span limit, it will die. When a certain number of agents have gone back, the server will compare all the collected paths and pick up the optimal path. Then, the packet is sent out to the destination along the optimal path. At the same time, the server updates its routing table.

To estimate the situation of resource consumption during mobile agent routing process, we analyze the population of mobile agents in the following section for both one single node and the network as a whole.

4. MATHEMATICAL ANALYSIS

First, we consider the case in which the agents run in the network with an infinite life-span. Mobile agents on a node can be divided into two parts: some are generated on this node, and others

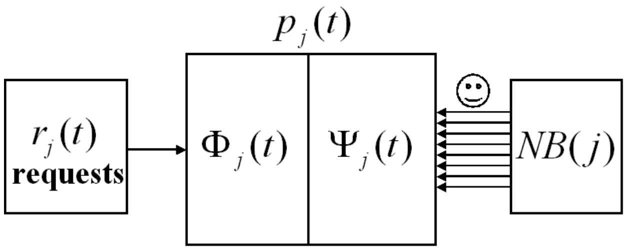

Figure 1. Mobile agents in one node can be divided into two parts. Some mobile agents are generated by requests received on the node, others come from the neighboring nodes of the current node. Here, $p_j(t)$ denotes the total number of mobile agents running on the jth node at time t, $r_j(t)$ denotes the number of requests received by the jth node at time t, $\Phi_j(t)$ denotes the number of agents newly generated on the jth node at time t, and $\Psi_j(t)$ denotes the number of agents on the jth node came from neighboring nodes at time t. $NB(j)$ is the set of neighboring nodes of the jth node.

come from the neighboring nodes of the current node. If we denote the number of agents on the jth node at time t by $p_j(t)$, the number of agents newly generated by the received requests at time t by $\Phi_j(t)$, and the number of agents coming from neighboring nodes at time t by $\Psi_j(t)$, then the dynamical change of the agents in the network can be given by the following equation (as shown in Figure 1):

$$p_j(t) = \Phi_j(t) + \Psi_j(t) \tag{1}$$

Let $NB(j)$ be the neighboring set of the jth node that is constructed by the neighboring nodes of the jth node and $\psi_{ij}(t)$ be the number of agents on the jth node at time t that move from the ith node. Equation (1) can be further expressed as follows:

$$p_j(t) = \Phi_j(t) + \Psi_j(t) = \Phi_j(t) + \sum_{i \in NB(j)} \psi_{ij}(t) \tag{2}$$

Assume that there are $r_j(t-1)$ requests received by the jth node at time $t-1$ and $k_j(t)$ agents will be generated at time t for each request, then $\Phi_j(t)$ satisfies

$$\Phi_j(t) = k_j(t) \cdot r_j(t-1) \tag{3}$$

For estimating $\psi_{ij}(t)$, we define a binary valued variable $\lambda_{ijl}(t)$ to indicate the selection of the lth agent on the ith node for the next jump as follows:

$$\lambda_{ijl}(t) = \begin{cases} 1 & \text{if the } l\text{th agent selects the } j\text{th node to go} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Obviously, $\lambda_{ijl}(t)$ satisfies

$$\sum_{j \in NB(i)} \lambda_{ijl}(t) = 1 \tag{5}$$

since each agent selects only one neighboring node once to go. Thus, we have

$$\psi_{ij}(t) = \sum_{l=1}^{p_i(t-1)} \lambda_{ijl}(t-1) \tag{6}$$

Therefore, we have

$$p_j(t) = \Phi_j(t) + \Psi_j(t) = k_j(t) \cdot r_j(t-1) + \sum_{i \in NB(j)} \sum_{l=1}^{p_i(t-1)} \lambda_{ijl}(t-1) \tag{7}$$

Also, as shown in Figure 1, mobile agents on the ith node at time $t-1$ can be divided into two parts: those newly generated on the ith node by received requests, denoted by $\Phi_i(t-1)$; and others coming from the neighboring nodes of the ith node, denoted by $\Psi_i(t-1)$. That is, $p_i(t-1) = \Phi_i(t-1) + \Psi_i(t-1)$. Agents in each part have a different probability to select the jth node to go compared with those belonging to the other part, which are denoted by $\Phi Pro\{j|i\}$ and $\Psi Pro\{j|i\}$, respectively. When $j \notin NB(i)$, the probability that an agent on the ith node will go to the jth node at the next jump is defined to be 0. The average number of agents on the jth node at time t that move from the ith node, denoted by $E[\psi_{ij}(t)]$, can be calculated as

$$\begin{aligned} E[\psi_{ij}(t)] &= E\left[\sum_{l=1}^{p_i(t-1)} \lambda_{ijl}(t-1)\right] \\ &= \Phi Pro\{j|i\}E[\Phi_i(t-1)] + \Psi Pro\{j|i\}E[\Psi_i(t-1)] \\ &= \Phi Pro\{j|i\}E[\Phi_i(t-1)] + \Psi Pro\{j|i\}E[p_i(t-1) - \Phi_i(t-1)] \\ &= (\Phi Pro\{j|i\} - \Psi Pro\{j|i\})E[\Phi_i(t-1)] + \Psi Pro\{j|i\}E[p_i(t-1)] \end{aligned} \tag{8}$$

Taking expectation on both side of (7), the evolution behavior of the agents satisfies

$$\begin{aligned} E[p_j(t)] &= E[\Phi_j(t)] + E[\Psi_j(t)] \\ &= E[k_j(t)] \cdot E[r_j(t)] + \sum_{i \in NB(j)} \Psi Pro\{j|i\}E[p_i(t-1)] \\ &\quad + \sum_{i \in NB(j)} (\Phi Pro\{j|i\} - \Psi Pro\{j|i\})E[k_i(t-1)] \cdot E[r_i(t-2)] \end{aligned} \tag{9}$$

based on the fact that variables $k_j(t)$ and $r_j(t)$ are independent with each other. Let $\hat{\cdot}$ be the expectancy $E[\cdot]$ and k be the average number of agents generated per request, Equation (9) can be rewritten as

$$\begin{aligned} \hat{p}_j(t) &= k\hat{r}_j(t) + \sum_{i \in NB(j)} \Psi Pro\{j|i\}\hat{p}_i(t-1) \\ &\quad + \sum_{i \in NB(j)} (\Phi Pro\{j|i\} - \Psi Pro\{j|i\})k\hat{r}_i(t-2) \end{aligned} \tag{10}$$

In compact form, it can be expressed as

$$\mathbf{p}(t) = k\mathbf{r}(t-1) + kA\mathbf{r}(t-2) + B\mathbf{p}(t-1) \tag{11}$$

where $A = (a_{ji})_{n \times n}$ is an n by n matrix with the elements defined as

$$a_{ji} = \begin{cases} \Phi Pro\{j|i\} - \Psi Pro\{j|i\} & \text{if } j \in NB(i) \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

and $B = (b_{ji})_{n\times n}$ is an n by n matrix with the elements defined as

$$b_{ji} = \begin{cases} \Psi Pro\{j|i\} & \text{if } j \in NB(i) \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

$\mathbf{p}(t)$ and $\mathbf{r}(t)$ are column vectors defined as

$$\mathbf{p}(t) = \begin{pmatrix} \hat{p}_1(t) \\ \hat{p}_2(t) \\ \cdots \\ \hat{p}_n(t) \end{pmatrix} \quad \text{and} \quad \mathbf{r}(t) = \begin{pmatrix} \hat{r}_1(t) \\ \hat{r}_2(t) \\ \cdots \\ \hat{r}_n(t) \end{pmatrix} \tag{14}$$

In particular, the jth node receives $r_j(0)$ requests at time $t = 0$ and generates $kr_j(0)$ agents at time $t = 1$. Therefore, from the previous discussion and by the assumption that there are no agents running in the network at time t, we have $\mathbf{p}(0) = 0$, $\mathbf{p}(1) = k\mathbf{r}(0)$, and $\mathbf{p}(2) = k\mathbf{r}(1) + kA\mathbf{r}(0) + Bk\mathbf{r}(0) = k\mathbf{r}(1) + k(A + B)\mathbf{r}(0)$. Without loss of generality, we assume that requests keyed in a network follow a unique probability distribution at any time unit and the average number of requests received by a node per time unit is r. Thus, the evolution behavior of mobile agents can be expressed as

$$\mathbf{p}(t) = \begin{cases} 0 & t = 0 \\ kr\mathbf{e} & t = 1 \\ kr\mathbf{e} + kr(A + B)\mathbf{e} & t = 2 \\ kr(I + A)\mathbf{e} + B\mathbf{p}(t - 1) & t \geq 2 \end{cases} \tag{15}$$

which comes down to a Markov process as it depends on only the state of the previous step. Here, $\mathbf{e}$ is a unit column vector defined as $\mathbf{e} = (1, 1, \ldots, 1)^{\mathrm{T}}$.

Now, we consider the case in which the agents run in the network with life-span limit. For this case, we have the following theorem.

Theorem 1. *The population distribution of mobile agents in the network can be rewritten as follows:*

$$\mathbf{p}(t) = \begin{cases} 0 & t = 0 \\ kr\mathbf{e} & t = 1 \\ kr\mathbf{e} + kr(A + B)\mathbf{e} & t = 2 \\ \sum_{i=0}^{t-2} B^i kr(I + A)\mathbf{e} + krB^{t-1}\mathbf{e} & 2 \leq t \leq L \\ \sum_{i=0}^{L-2} B^i kr(I + A)\mathbf{e} + krB^{L-1}\mathbf{e} & t > L \end{cases} \tag{16}$$

where $\mathbf{p}(t)$*, defined in (14), denotes the distribution of mobile agents in the network, A and B are matrices defined by (12) and (13),* $\mathbf{e} = (1, 1, \ldots, 1)^{\mathrm{T}}$ *is a unit column vector, k is the average number of agents generated per request, r is the average number of requests received by a node per time unit, and L is the life-span limit of a mobile agent.*

Proof. For mobile agents with life-span limit, if the searching time of mobile agents is less than the life-span limit, they will follow the same distribution as (15). However, as the searching process continues, those agents with searching times greater than the life-span limit will expire, which results in a different distribution. Let L be the life-span limit of a mobile agent. We first analyze the distribution of agents after they made L jumps by using the jth node as a spot. Since the number of received requests on the jth node at time $t-1$ is $r_j(t-1)$ and the number of agents generated per request at time t is $k_j(t)$, the number of newly generated mobile agents on the jth node at time t is $k_j(t)r_j(t-1)$. These agents will then roam in the network and search for their destinations node by node. At time $t+1$, these generated agents will either find their destinations inside the neighboring set of the jth node or move to a neighboring node of the jth node. Since we only consider the distribution of mobile agents generated at time t in the network and assume that there are no newly generated agents in the network from time t, at time $t+1$, all the agents on the jth node are from the neighboring nodes of the jth node. Denoting the number of these agents on the jth node by $p_j^*(\cdot)$ and the number of agents among these agents come from the neighboring nodes of the jth node by $\Psi_j^*(\cdot)$, we have

$$p_j^*(t+1) = \Psi_j^*(t+1) = \sum_{i \in NB(j)} \sum_{l=1}^{p_i^*(t)} \lambda_{ijl}(t) \tag{17}$$

where $p_i^*(t) = k_i(t) \cdot r_i(t-1)$. Taking expectation on both side of (16), we have

$$\hat{p}_j^*(t+1) = \sum_{i \in NB(j)} \Phi Pro\{j|i\}kr \tag{18}$$

In compact form, it is expressed as

$$\mathbf{p}^*(t+1) = kr(A+B)\mathbf{e} \tag{19}$$

From (11), it is easy to get the following results:

$$\begin{aligned} \mathbf{p}^*(t+2) &= B\mathbf{p}^*(t+1) \\ \mathbf{p}^*(t+3) &= B\mathbf{p}^*(t+2) = B^2\mathbf{p}^*(t+1) \\ &\vdots \\ \mathbf{p}^*(t+L) &= B^{L-1}\mathbf{p}^*(t+1) = krB^{L-1}(A+B)\mathbf{e} \end{aligned} \tag{20}$$

Combining the results of (15) and (20), the population distribution of mobile agents with life-span limit follows a Markov process as:

$$\mathbf{p}(t) = \begin{cases} 0 & t=0 \\ kr\mathbf{e} & t=1 \\ kr\mathbf{e} + kr(A+B)\mathbf{e} & t=2 \\ kr(I+A)\mathbf{e} + B\mathbf{p}(t-1) & 2 \le t \le L \\ kr(I+A)\mathbf{e} + B\mathbf{p}(t-1) - krB^{L-1}(A+B)\mathbf{e} & t > L \end{cases} \tag{21}$$

Furthermore, with the fact that when $2 \leq t \leq L$

$$\begin{aligned}
\mathbf{p}(t) &= kr(I + A)\mathbf{e} + B\mathbf{p}(t-1)\\
&= kr(I + A)\mathbf{e} + B[kr(I + A)\mathbf{e} + B\mathbf{p}(t-2)]\\
&= kr(I + B)(I + A)\mathbf{e} + B^2\mathbf{p}(t-2)\\
&= kr(I + B)(I + A)\mathbf{e} + B^2[kr(I + A)\mathbf{e} + B\mathbf{p}(t-3)]\\
&= kr(I + B + B^2)(I + A)\mathbf{e} + B^3\mathbf{p}(t-3)\\
&= \cdots\\
&= \sum_{i=0}^{t-3} B^i kr(I + A)\mathbf{e} + B^{t-2}\mathbf{p}(2)\\
&= \sum_{i=0}^{t-3} B^i kr(I + A)\mathbf{e} + B^{t-2}[kr\mathbf{e} + kr(A + B)\mathbf{e}]\\
&= \sum_{i=0}^{t-2} B^i kr(I + A)\mathbf{e} + krB^{t-1}\mathbf{e}
\end{aligned} \tag{22}$$

and when $t > L$

$$\begin{aligned}
\mathbf{p}(t) &= kr(I + A)\mathbf{e} - krB^{L-1}(A + B)\mathbf{e} + B\mathbf{p}(t-1)\\
&= \mathbf{X} + B\mathbf{p}(t-1) \quad (\text{where } \mathbf{X} = kr(I + A)\mathbf{e} - krB^{L-1}(A + B)\mathbf{e})\\
&= \cdots\\
&= \sum_{i=0}^{t-L-1} B^i\mathbf{X} + B^{t-L}\mathbf{p}(L)\\
&= \sum_{i=0}^{t-L-1} B^i[kr(I + A)\mathbf{e} - krB^{L-1}(A + B)\mathbf{e}] + B^{t-L}\left[\sum_{i=0}^{L-2} B^i kr(I + A)\mathbf{e} + krB^{L-1}\mathbf{e}\right]\\
&= \sum_{i=0}^{t-L-1} B^i kr(I + A)\mathbf{e} - \sum_{i=L-1}^{t-2} B^i kr(A + B)\mathbf{e} + \sum_{i=t-L}^{t-2} B^i kr(I + A)\mathbf{e} + krB^{t-1}\mathbf{e}\\
&= \sum_{i=0}^{t-2} B^i kr(I + A)\mathbf{e} - \sum_{i=L-1}^{t-2} B^i kr(A + B)\mathbf{e} + krB^{t-1}\mathbf{e}\\
&= \sum_{i=0}^{t-2} B^i kr\mathbf{e} + \sum_{i=0}^{t-2} B^i krA\mathbf{e} - \sum_{i=L-1}^{t-2} B^i krA\mathbf{e} - \sum_{i=L-1}^{t-2} B^i krB\mathbf{e} + krB^{t-1}\mathbf{e}\\
&= \sum_{i=0}^{L-1} B^i kr\mathbf{e} + \sum_{i=0}^{L-2} B^i krA\mathbf{e}\\
&= \sum_{i=0}^{L-2} B^i kr(I + A)\mathbf{e} + B^{L-1}kr\mathbf{e}
\end{aligned} \tag{23}$$

the population distribution of mobile agents in the network can be rewritten as

$$\mathbf{p}(t) = \begin{cases} 0 & t = 0 \\ kr\mathbf{e} & t = 1 \\ kr\mathbf{e} + kr(A+B)\mathbf{e} & t = 2 \\ \displaystyle\sum_{i=0}^{t-2} B^i kr(I+A)\mathbf{e} + krB^{t-1}\mathbf{e} & 2 \le t \le L \\ \displaystyle\sum_{i=0}^{L-2} B^i kr(I+A)\mathbf{e} + krB^{L-1}\mathbf{e} & t > L \end{cases} \tag{24}$$

Therefore, the theorem is proven. □

It is easy to see that the number of agents in our model will not increase infinitely with the searching time. Thus, by tuning the number of agents generated per request and the agent life-span limit, the population of mobile agents can be controlled within the system limitations.

Since mobile agents are generated frequently and dispatched to the network, it is important to estimate the number of mobile agents running in the network and on each node. When there are too many agents in the network, they will introduce too much computational overhead to node machines, which will eventually become very busy and indirectly block the network traffic. Based on the population distribution, we further analyze the total number of agents in the network and the number of agents on a node to estimate the consumption of computational resource of our model.

First, we introduce some notations used in our analysis. As assumed at the beginning of this section, the network topology is a connected graph so that there is at least one path (directly or indirectly) between any two nodes. Matrix $C = (c_{ij})_{n\times n}$ is the connectivity matrix that describes the connectivity of the graph, i.e. if there is a direct link between node i and node j, then $c_{ij} = c_{ji} = 1$; otherwise, $c_{ij} = 0$. In particular, $c_{ii} = 0$ for any i. Let c_j be the jth column vector of matrix C. That is, $C = (c_1, c_2, \ldots, c_n)$. Furthermore, $d_j = \|c_j\|_1 = \sum_{i=1}^{n} |c_{ij}|$, $\sigma_1 = \max_{1\le j\le n} d_j$, and $\sigma_n = \min_{1\le j\le n} d_j$. Obviously, $D = \operatorname{diag}(d_1, d_2, \ldots, d_n)$ is a diagonal matrix that indicates the connectivity degree of the network topology. It is easy to see that d_j is the number of neighboring nodes of the jth node including itself, and $\|C\|_1 = \max_{1\le j\le n} \|c_j\|_1 = \sigma_1$. For example, suppose that the graphical structure of a small network is shown in Figure 2.

Accordingly, we have $n = 5$, $\sigma_1 = 4$, $\sigma_5 = 1$. Matrices C and D can be given as follows:

$$C = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Note that once an agent reaches a node, it has a chance to find its destination in the neighboring set of the host node and also a risk of dying due to a system failure on the node. As our model considers the problem of routing for a remote destination, that is, the server has no routing information about the destination, agents will search node by node in the network and each node in the network has the same possibility to be the destination of an agent. For an intermediate node, each unsearched neighboring

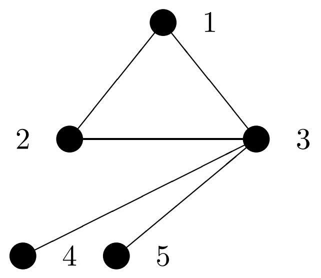

Figure 2. An example of a small network.

node has same possibility of being selected by an agent to go at the next hop. Thus, the probability that a newly generated agent on the ith node can find its destination on the server is $(d_i + 1)/n$ since the agent can check addresses of both the server and its d_i neighboring nodes. If the agent cannot find its destination locally, the probability that it can move out of the ith node is $(1 - p)[1 - (d_i + 1)/n]$ where p is the probability that the host will turn down when an agent stays on it. Then the probability that the agent will select the jth node to go is

$$\Phi Pro\{j|i\} = \frac{1-p}{d_i}\left(1 - \frac{d_i + 1}{n}\right) \quad \text{where } j \in NB(i) \tag{25}$$

For an agent on the ith node that is not newly generated locally, since the ith node and the previous node from which the agent came has been checked when the agent was on the previous node, the probability that the agent can find its destination is equal to $(d_i - 1)/n$ and the probability that it will move to the jth node is given by

$$\Psi Pro\{j|i\} = \frac{1-p}{d_i}\left(1 - \frac{d_i - 1}{n}\right) \quad \text{where } j \in NB(i) \tag{26}$$

From the definitions in (12) and (13), we have

$$A = -\frac{2(1-p)}{n} CD^{-1} \tag{27}$$

and

$$B = (1-p)CD^{-1}\left(I - \frac{D - I}{n}\right) \tag{28}$$

From these results, we can estimate the number of agents as follows.

Theorem 2. *The total number of agents in the network is no more than*

$$kr\theta \Big/ \left[1 - (1-p)\left(1 - \frac{\sigma_n - 1}{n}\right)\right]$$

where

$$\theta = n - 2(1-p) - (1-p)^L\left(1 - \frac{\sigma_n - 1}{n}\right)^{L-1}(n - \sigma_n - 1)$$

Proof. The total number of agents in the network can be calculated as

$$\sum_{i=1}^{n} p_j(t) = \|\mathbf{p}(t)\|_1 \tag{29}$$

Thus, from Theorem (4), the total number of agents can be estimated as follows:

$$\begin{aligned}\|\mathbf{p}(t)\|_1 &\leq \left\| \sum_{i=0}^{L-2} B^i kr(I+A)\mathbf{e} + krB^{L-1}\mathbf{e} \right\|_1 \\ &\leq kr \cdot \left(\sum_{i=0}^{L-2} \|B\|_1^i \cdot \|I+A\|_1 + \|B\|_i^{L-1} \right) \|\mathbf{e}\|_1 \end{aligned} \tag{30}$$

Due to the fact that

$$\begin{aligned}\|I+A\|_1 &= \left\| I - \frac{2(1-p)}{n} CD^{-1} \right\|_1 = 1 - \frac{2(1-p)}{n} \\ \|B\|_1 &= \left\| (1-p)CD^{-1}\left(I - \frac{D-I}{n}\right) \right\|_1 = (1-p)\left(1 - \frac{\sigma_n - 1}{n}\right)\end{aligned} \tag{31}$$

and $\|\mathbf{e}\|_1 = n$, the total number of agents satisfies

$$\begin{aligned}\|\mathbf{p}(t)\|_1 &\leq nkr\left(\sum_{i=0}^{L-1} \|B\|_1^i - \frac{2(1-p)}{n} \sum_{i=0}^{L-2} \|B\|_1^i \right) \\ &= \frac{kr\theta}{1-(1-p)(1-(\sigma_n-1)/n)}\end{aligned} \tag{32}$$

where θ is defined as

$$\begin{aligned}\theta &= n - n\left[(1-p)\left(1-\frac{\sigma_n-1}{n}\right)\right]^L - 2(1-p) + 2(1-p)\left[(1-p)\left(1-\frac{\sigma_n-1}{n}\right)\right]^{L-1} \\ &= n - 2(1-p) - (1-p)^L\left(1-\frac{\sigma_n-1}{n}\right)^{L-1}\left[n\left(1-\frac{\sigma_n-1}{n}\right)-2\right] \\ &= n - 2(1-p) - (1-p)^L\left(1-\frac{\sigma_n-1}{n}\right)^{L-1}(n-\sigma_n-1) \\ &\leq n - 2(1-p)\end{aligned} \tag{33}$$

Thus, the theorem is proven. □

Similarly, the number of agents running on a node can be estimated as follows.

Theorem 3. *The number of agents running on a node is no more than*

$$\frac{kr\delta/n\sigma_1}{1-((1-p)/\sigma_n)(1-(\sigma_n-1)/n)}$$

where

$$\delta = n\sigma_1 - 2(1-p) - \left(\frac{1-p}{\sigma_n}\right)^L \left(1 - \frac{\sigma_n-1}{n}\right)^{L-1} [n\sigma_1 - \sigma_1(\sigma_n - 1) - 2\sigma_n]$$

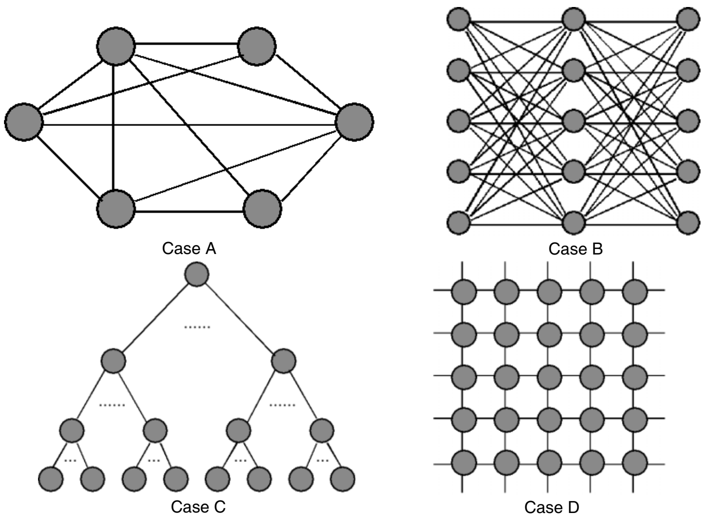

Figure 3. Simple examples for each case.

Proof. From Theorem 4, the number of agents running on a node can be estimated as

$$\begin{aligned}\max_{1\le i\le n} p_j(t) = \|\mathbf{p}(t)\|_\infty &\le \left\| \sum_{i=0}^{L-2} B^i kr(I+A)\mathbf{e} + krB^{L-1}\mathbf{e} \right\|_\infty \\ &= kr\left\| \sum_{i=0}^{L-2} B^i(I+A) + B^{L-1} \right\|_\infty \le kr\left(\sum_{i=0}^{L-2} \|B\|_\infty^i \cdot \|I+A\|_\infty + \|B\|_\infty^{L-1} \right) \end{aligned} \tag{34}$$

Due to the fact that

$$\|I+A\|_\infty = 1 - \frac{2(1-p)}{n\sigma_1} \quad \text{and} \quad \|B\|_\infty = \frac{1-p}{\sigma_n}\left(1 - \frac{\sigma_n - 1}{n}\right) \tag{35}$$

the maximum number of agents running on a node satisfies

$$\|\mathbf{p}(t)\|_\infty \le kr\left(\sum_{i=0}^{L-1} \|B\|_\infty^i - \frac{2(1-p)}{n\sigma_1} \sum_{i=0}^{L-2} \|B\|_\infty^i \right) = \frac{kr\delta/n\sigma_1}{1-(1-p)/\sigma_n(1-(\sigma_n-1)/n)} \tag{36}$$

where δ is defined as

$$\begin{aligned}\delta &= n\sigma_1\left(1 - \left[\frac{1-p}{\sigma_n}\left(1-\frac{\sigma_n-1}{n}\right)\right]^L\right) - 2(1-p)\left(1 - \left[\frac{1-p}{\sigma_n}\left(1-\frac{\sigma_n-1}{n}\right)\right]^{L-1}\right) \\ &= n\sigma_1 - 2(1-p) - \left(\frac{1-p}{\sigma_n}\right)^L \left(1 - \frac{\sigma_n-1}{n}\right)^{L-1} [n\sigma_1 - \sigma_1(\sigma_n - 1) - 2\sigma_n] \\ &\le n\sigma_1 - 2(1-p) \end{aligned} \tag{37}$$

Thus, the theorem is proven. □

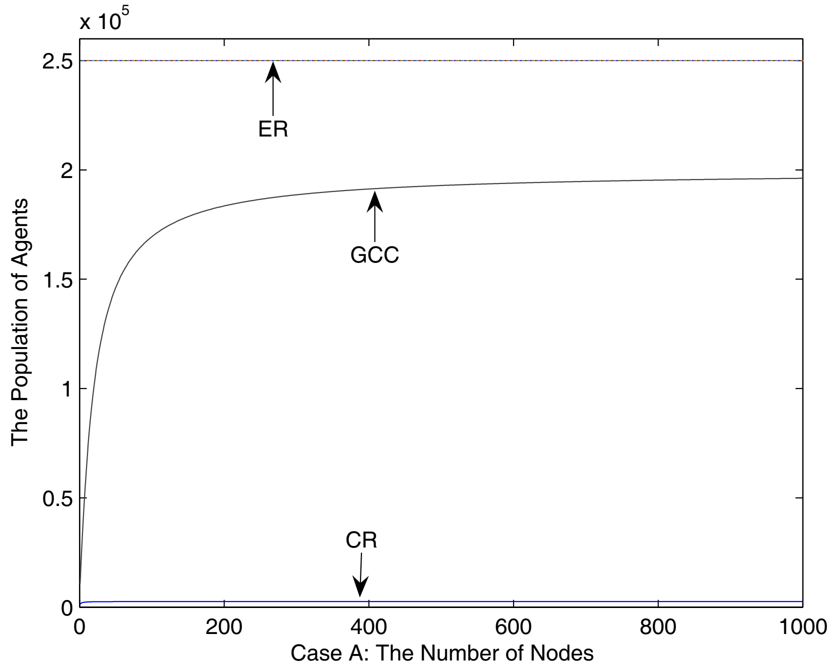

Figure 4. Results for Case A.

From the above analytical results, we can claim that both the number of mobile agents in the network and the number of mobile agents on each node will not increase infinitely over time t. The upper bound of these two numbers can be controlled by tuning the number of mobile agents generated per request.

5. EXPERIMENTAL RESULTS

In this section, we present some simulation results on the population of mobile agents from OMNet++ simulator [28] in order to evaluate our model in comparison with existing models from [20,26]. Due to space limitation, only the following cases of network topologies are included in the experiments:

1. Case A: σ_1 large, σ_n medium;
2. Case B: σ_1 medium, σ_n medium;
3. Case C: σ_1 medium, σ_n small;
4. Case D: σ_1 small, σ_n small.

Figure 3 gives some simple examples for these cases.

We use ER to denote the result from [26], GCC to denote the result from [20], and CR to denote the result in this paper. Figures 4–7 show the results on the population of agents running in a host.

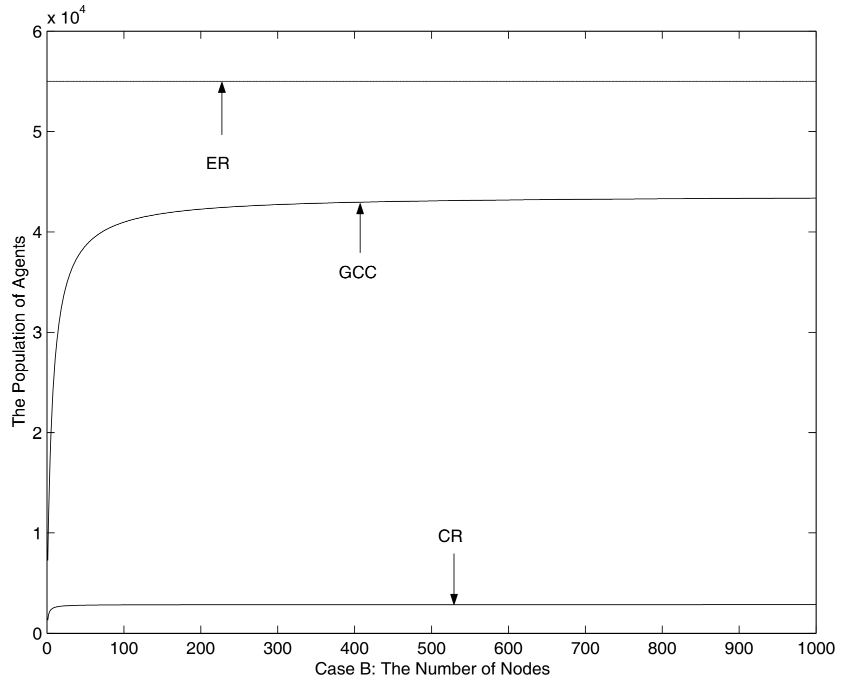

Figure 5. Results for Case B.

In detail, we set $\sigma_1 = 100$ and $\sigma_n = 30$ for Case A, $\sigma_1 = 30$ and $\sigma_n = 20$ for Case B, $\sigma_1 = 30$ and $\sigma_n = 4$ for Case C, and $\sigma_1 = 5$ and $\sigma_n = 3$ for Case D. From the experimental results we can obviously see that our model can outperform the existing results greatly. As [26] sets a death rate to mobile agents so that an agent may stop its searching activity at any step before finding the destination, this leads to a relatively poorer searching efficiency as compared with [20] and therefore a larger amount of agents are needed. Our model allows agents to check the routing addresses of multiple nodes at each jump, which results in a superior performance among these three models. In our experiments, there are 1000 nodes randomly generated by the simulator, for each time unit there are 50 requests received from a node, and for each node, 50 mobile agents are generated. Each mobile agent generated has a life-span limitation of 50 hops. The probability that a node may fail is set to be 0.001. From these figures we can also see that the number of agent is an increasing function on the size of the network.

6. CONCLUDING REMARKS

In this paper, we addressed the problem of network routing and management by deploying mobile agents. We proposed an extended routing algorithm and analyzed the population distribution of mobile agents under the proposed model. Based on our analytical results, the number of agents is

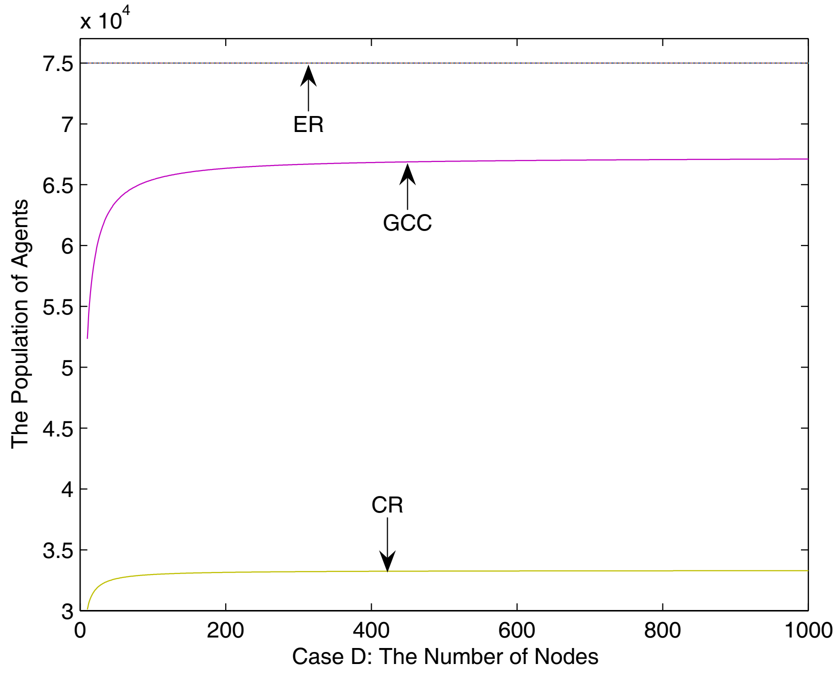

Figure 6. Results for Case C.

a monotonically increasing function on the number of nodes in the network, the number of agents generated per request, and the agent life-span limit and a monotonically decreasing function on the possibility system failures. Therefore, we can dispatch a small number of mobile agents and achieve a good network performance by selecting an optimal number of mobile agents generated per request and giving them an optimal life-span limit. We also compared our model with existing models. Since agents in our model can perform a more efficient search, the number of agents needed per request is less than that in existing models. On the other hand, with the same number of agents generated per request, the searching time of the agents is less than in existing models, which results in a superior performance compared with existing models. Our contributions are summarized as follows.

- We proposed a mobile agent-based routing model for use in vulnerable networks in which a mobile agent on a node knows information of its neighboring nodes and is assumed a life-span limit.
- We analyzed the population distribution of mobile agents, which provided a useful tool to reduce the computational resource consumption by adjusting the number of agents to be generated at individual nodes and the life-span of these mobile agents.

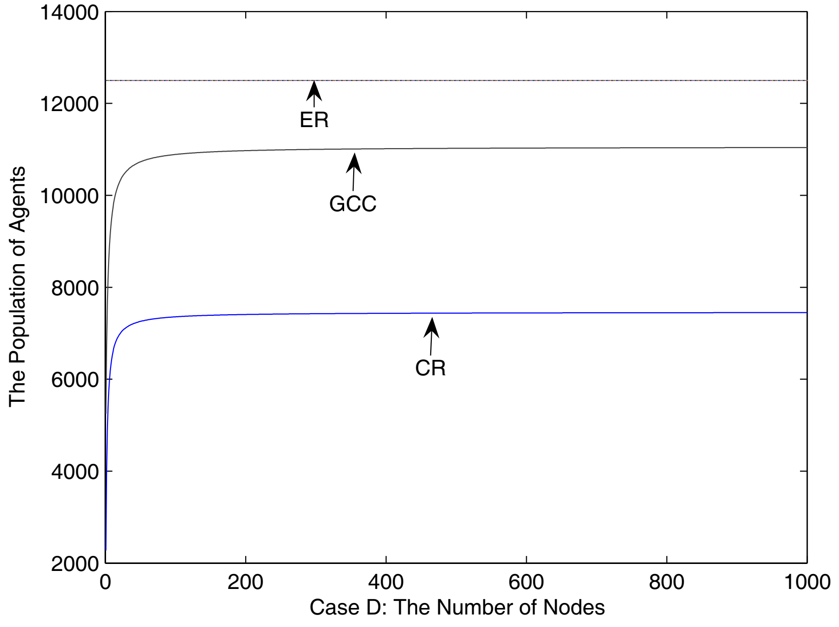

Figure 7. Results for Case D.

ACKNOWLEDGEMENT

This research is conducted as a program for the '21st Century COE Program' by the Ministry of Education, Culture, Sports, Science and Technology.

REFERENCES

1. Cisco—Routing Basics: A Technical Overview. http://www.cisco.com/warp/public/779/smbiz/community/routing_to.html [28 February 2006].
2. Piscitello D, Chapin B, Chapin A. Introduction to Routing. http://www.corecom.com/html/OSNconnexions.html [28 February 2006].
3. Li K, Shen H. Coordinated en-route multimedia object caching in transcoding proxies for tree networks. *ACM Transactions on Multimedia Computing, Communications and Applications* 2005; **5**(3):289–314.
4. Li K, Shen H, Chin F, Zheng S. Optimal methods for coordinated en-route Web caching for tree networks. *ACM Transactions on Internet Technology* 2005; **5**(3):480–507.
5. Zhuge H, Shi X, Fighting epistemology in knowledge and information age. *IEEE Computer* 2003; **36**(10):114–116.
6. Zhuge H, Sun X, Liu J, Yao E, Chen X. A scalable P2P platform for the knowledge Grid. *IEEE Transactions on Knowledge and Data Engineering* 2005; **17**(12):1721–1736.
7. Zhuge H, Liu J, Feng L, Sun X, He C. Query routing in a peer-to-peer semantic link network. *Computational Intelligence* 2005; **21**(2):197–216.

8. Brewington B, Gray R, Moizumi K, Kotz D, Cybenko G, Rus D. Intelligent information agents: Agents-based information discovery and management on the Internet. *Mobile Agents in Distributed Information Retrieval*, Klusch M (ed.), ch. 15. Springer: Berlin, 1999; 355–395.
9. Qu W, Shen H, Sum J. New analysis on mobile agents based network routing. *Proceedings of the 3rd International Conference on Hybrid Intelligence Systems (HIS'03)*, 2003; 769–778 (Best Student Paper Award).
10. Milojicic D. Guest editor's introduction: Agent systems and applications. *IEEE Concurrency* 2000; **8**(2):22–23.
11. Zhuge H. *The Knowledge Grid*. World Scientific: Singapore, 2004.
12. Lange D, Oshima M. Seven good reasons for mobile agents. *Communications of the ACM* 1999; **42**:88–89.
13. Tripathi A. http://www.cs.umn.edu/Ajanta/ [28 February 2006].
14. TOSHIBA Corporation. http://www.toshiba.co.jp/rdc/beegent/index.htm [28 February 2006].
15. Gray R, Cybenko G, Kotz D, Peterson R, Rus D. D'Agents: Applications and performance of a mobile-agent system. *Software: Practice and Experience* 2002; **32**(6):543–573.
16. General Magic, Inc. Odyssey Product Information, 1997. http://www.genmagic.com/agents/odyssey.html [25 June 2006].
17. Claessens J, Preneel B, Vandewalle J. (How) can mobile agents do secure electronic transactions on untrusted hosts? A survey of the security issues and the current solutions. *ACM Transactions on Internet Technology* 2003; **3**(1):28–48.
18. Karjoth G, Lange D, Oshima M. A security model for aglets. *IEEE Internet Computing* 1997; **1**(4):68–77.
19. Lange D, Oshima M. *Programming and Developing Java Mobile Agents with Aglets*. Addison-Wesley: Reading, MA, 1998.
20. Qu W, Shen H, Jin Y. Distribution of mobile agents in vulnerable networks. *Proceedings of the 4th International Conference on Grid and Cooperative Computing (GCC 2005)* (*Lecture Notes in Computer Science*, vol. 3795). Springer: Berlin, 2005; 894–905.
21. White T, Pagurek B, Oppacher F. Connection management using adaptive agents. *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'98)*. CSREA Press: Las Vegas, NV, 1998; 802–809.
22. Dorigo M, Gambardella L. Ant colonies for the traveling salesman problem. *BioSystems* 1997; **43**:73–81.
23. White T, Pagurek B, Oppacher F. ASGA: Improving the ant system by integration with genetic algorithms. *Proceedings of the 3rd Conference on Genetic Programming (GP/SGA'98)*, 1998; 610–617.
24. Caro G, Dorigo M. AntNet: A mobile agents approach to adaptive routing. *Technique Report IRIDIa/97-12*, Universite Libre de Bruxelles, Belgium, 1997.
25. Caro G, Dorigo M. AntNet: Distributed stigmergetic control for communication networks. *Journal of Artificial Intelligence* 1998; **9**:317–365.
26. Sum J, Shen H, Leung C, Young G. Analysis on mobile-agent based algorithm for network routing and management. *IEEE Transactions on Parallel and Distributed Systems* 2003; **14**(3):193–2002.
27. Fischer M, Lynch N, Paterson M. Impossibility of distributed consensus with one faulty process. *Journal of the ACM* 1985; **32**(2):374–382.
28. Varga A. OMNeT++: Discrete Event Simulation System User Manual. http://www.omnetpp.org/.