Bio-inspired Load Balancing Routing for Delay-Guaranteed Services in Ever-Changing Networks

Young-Min Kim, Hak Suh Kim, Boo-Geum Jung, Hea-Sook Park, and Hong-Shik Park

We consider a new load balancing routing for delay-guaranteed services in the network in which the traffic is dynamic and network topologies frequently change. For such an ever-changing network, we propose a new online load balancing routing called AntLBR, which exploits the ant colony optimization method. Generally, to achieve load balancing, researchers have tried to calculate the traffic split ratio by solving a complicated linear programming (LP) problem under the static network environment. In contrast, the proposed AntLBR does not make any attempt to solve this complicated LP problem. So as to achieve load balancing, AntLBR simply forwards incoming flows by referring to the amount of pheromone trails. Simulation results indicate that the AntLBR algorithm achieves a more load-balanced network under the changing network environment than techniques used in previous research while guaranteeing the requirements of delay-guaranteed services.

Keywords: Ant colony optimization, ever-changing networks, load balancing, virtual subcolony.

Manuscript received May 11, 2012; revised Jan. 4, 2013; accepted Jan. 6, 2013.

This work was supported by the IT R&D program of MKE/KEIT (10043437, Development of Virtual Private Cloud Technology).

Young-Min Kim (phone: +82 42 860 5583, injesus@etri.re.kr), Hak Suh Kim (tuple@etri.re.kr), Boo-Geum Jung (bgjung@etri.re.kr), and Hea-Sook Park (parkhs@etri.re.kr) are with the Communications & Internet Research Laboratory, ETRI, Daejeon, Rep. of Korea.

Hong-Shik Park (parkhs@ee.kaist.ac.kr) is with the Department of Electrical Engineering, KAIST, Daejeon, Rep. of Korea.

http://dx.doi.org/10.4218/etrij.13.1812.0114

I. Introduction

With the advance in a wide range of multimedia services, the difficulty achieving load balancing for delay-guaranteed services stands out as a major problem in communication networks. As we already know, a significant amount of research has been done regarding this problem, and researchers have successfully achieved their objectives. However, to achieve their objectives, they generally assumed the following two conditions. First, the traffic demand of each source-destination pair is given. Based on the given information, define the linear programming (LP) problem for load balancing and find the solution by heuristically optimizing it. Note that the required time to solve the LP problem is not negligible. Second, the network status is temporarily stable (that is, the traffic information does not change over time) and the given traffic demand hence describes the current network status accurately. The accuracy of the given information should last until the time at which the next traffic demand is given. Accordingly, it is difficult to apply this approach to the changing network status in an online manner.

It is widely accepted that a wide range of multimedia services are becoming increasingly popular on the Internet, and that this popularity will exponentially increase with the advent of such social network services as Twitter [1]. In addition, various current networks will integrate into a general unified (or converged) network, that is, the next-generation network [2]. These trends will accelerate the dynamic nature of Internet traffic and the changes of network topologies, hence intensifying the ever-changing state of the network. That the Internet is ever-changing seriously threatens the accuracy of the information regarding the network status and also exponentially increases the computational complexity of solving a complicated LP problem in a large-scale network. It may require an emerging load balancing routing algorithm to be capable of handling new challenging issues for the ever-changing network: it may 1) promptly gather the information that precisely describes the ever-changing network and 2) efficiently achieve the load balancing in a negligible amount of time.

Current Internet service providers usually over-provision the network capacity (at a peak rate), so as to deal with the Internet's quality of being ever-changing. However, this not only leads to an under-utilized network throughput (as low as 20% [3]) but also does not properly handle the frequent changes of topologies.

Some routing schemes [4]-[7] have been proposed to solve the under-utilization of networks while coping with the variety of the traffic. The authors in [4], [5] proposed two-phase traffic oblivious routing wherein the traffic split ratio is predetermined and does not change irrespective of rapidly varying traffic patterns. However, it requires too much complexity in practice owing to the nondeterministic polynomial time (NP) completeness of the problem. The authors in [6], [7] reduced this complexity using shortest path routing in each of the two phases. Basically, these routing schemes are based on the experiment results presented in [8]: with fairly limited or no knowledge of the traffic demand, it is possible to design a routing scheme to obtain desirable utilization only by committing the static preconfiguration. However, as described in [9], it cannot avoid the degradation of network performances due to the lack of knowledge about the network status, which is explained as follows. First, the end-to-end hop count as a result of the two-phase routing is about 1.4 to 1.6 times that of shortest path routing. Note that the increase of the hop count causes more consumption of resources, and it hence leads to the degradation of the network throughput. Second, the end-to-end delay of the two-phase routing is generally about twice that of shortest path routing. That is, the two-phase routing scheme may not be suitable for delay-sensitive services. In addition, the authors in [10] mentioned that the normalized congestion ratio of two-phase routing is twice as big as that of Multiprotocol Label Switching traffic engineering (MPLS-TE).

Recently, to efficiently cope with the ever-changing state of the Internet, there were attempts to design networks equipped with self-organizing capabilities (the authors in [11] provided principles and benefits of self-organizing networks). Such networks promptly adapt to ever-changing environments (network flexibility), and the adapted networks then effectively perform their tasks while minimizing the impact of unpredictable network failures (network robustness).

To handle the Internet's quality of being ever-changing, we pay attention to the swarm intelligence (SI) of social insects, especially in the artificial ant colony [12]: without any supervised control, the colony can solve very complex problems (such as NP-complete problems) by carefully coordinating individual ants via very simple primitive interaction. In addition, the SI of artificial ants brings about some amazing capabilities [13]-[16]: the colony promptly adapts to the changing environment (flexibility) and still performs its tasks even when one or more individual ants fail (robustness). These are very desirable characteristics for the ever-changing network.

In this paper, we propose a new load balancing routing called AntLBR, which supports the aforementioned self-organizing capabilities. The proposed AntLBR algorithm tries to achieve the load balancing while accommodating the delay-guaranteed services. For this purpose, AntLBR employs a single artificial ant colony for the ever-changing network; a number of artificial ants in the colony asynchronously explore the network and measure or gather its ever-changing status, such as sudden inputs of highly bursty traffic or unexpected network failures. Based on the gathered information, AntLBR organizes and maintains a small number of virtual subcolonies. AntLBR lets the subcolonies reconfigure themselves to promptly adapt to the network's changes without any supervised control. By using adapted subcolonies, AntLBR greatly enhances the load balancing performance by innovatively resolving the congested traffic. In contrast to algorithms presented in previous research, AntLBR has the following characteristics:

- Compared with previously proposed algorithms that should solve a complicated LP problem, the AntLBR algorithm does not make any attempt to solve the complicated problem; When an incoming flow arrives, AntLBR simply forwards the flow by referring to the highest pheromone trails; Then, as time passes, the network gradually evolves into a load-balanced network;
- Accordingly, the routing decisions in AntLBR can be made in an online manner;
- The AntLBR algorithm tries to understand the current changing network status as thoroughly as possible by utilizing the self-learning and self-adjusting properties;
- In previous research, the traffic split ratio did not change irrespective of the ever-changing network status; However, in AntLBR, the congested traffic is naturally distributed or split onto adjacent links;
- The AntLBR algorithm exhibits a high degree of self-organizing networking properties, such as network robustness and scalability.

II. Proposed AntLBR Routing — Concepts

A wide variety of emerging services in future networks will require that the requested quality is guaranteed. Accordingly,

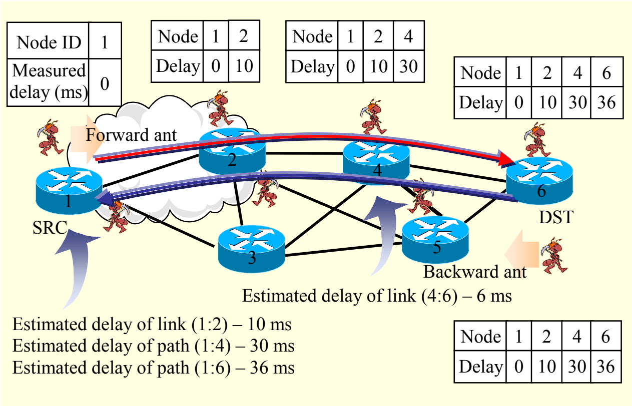

Fig. 1. Operation concept of artificial ants in AntLBR.

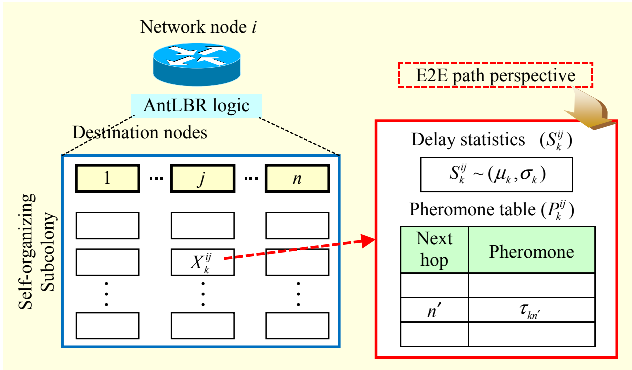

Fig. 2. AntLBR logic in node.

load balancing routing for future networks should not only evenly distribute the bursty traffic but also guarantee meeting the quality demands for emerging services.

For these purposes, AntLBR employs one single artificial ant colony on a network. At regular intervals, every network node asynchronously launches an artificial ant toward a randomly chosen destination node. Figure 1 describes the operation principle of the artificial ant. Source node 1 sends a forward ant toward destination node 6. The forward ant travels the network toward its destination node while memorizing such information as the identifier and arrival time of all visited nodes. The important thing is that the forward ant shares the same queue as the data packets to make it experience the same conditions of the data packets. When arriving at destination node 6, the forward ant generates a backward ant, transfers the memorized information to the generated backward ant, and dissipates. The backward ant moves through the high-priority queue (not the same queue as the data packets, so as to quickly update and propagate the memorized information) toward source node 1 by taking the same path of the corresponding forward ant but in the opposite direction. When receiving the backward ant, all intermediate nodes along the taken path can estimate the current network status using the information as follows:

- Node 4 estimates that the elapsed delay from node 4 to node 6 is close to 6 ms;
- Node 2 estimates that the delay of the link from node 2 to node 4 is close to 20 ms; Additionally, it estimates that the delay of the path from node 2 to node 6 is 26 ms;
- Node 1 estimates the delays along the link and paths from node 1 to node 2, node 4, and node 6 in the manner explained above.

Whenever a backward ant arrives, the node continues to learn the information about paths or links by estimating the delays. As time passes, the node can confirm the existence of some paths or links, although it does not know exactly what the paths are composed of.

In general, there are many potential paths even in the same source-destination node pair: some of these paths may have longer delays than others and may have longer paths, that is, a greater number of hops. In addition, even in the same path, the delays may be widely dispersed according to traffic conditions of the nodes and links along the path. Accordingly, the delays on these various paths measured by a number of artificial ants may also be widely dispersed.

The AntLBR logic in a node classifies these wide ranges of dispersed delays into a set of similar ones and organizes the virtual subcolony with respect to the classified set. Figure 2 illustrates the framework of AntLBR in node i, and the right side of the figure describes the structure of the subcolony.

We describe the notations used throughout the paper. We use the term X to refer to the general subcolony, X^{ij} to refer to the complete set of organized subcolonies of node i from node i to node j, and X_k^{ij} to refer to the k-th organized subcolony in X^{ij}.

Subcolony X_k^{ij} has two data structures, namely, the delay statistic S_k^{ij} and the pheromone table P_k^{ij}. The estimated statistic information about the delays of the paths from node i to node j explored by a number of artificial ants is $S_k^{ij} \sim (\mu_k, \sigma_k)$. The sample mean and variance estimated over the measured delays are μ_k and σ_k, respectively. The set of pheromone trails $\tau_{kn'}$ is P_k^{ij} ($n' \in N_i$, where N_i is the set of neighbors of node i). The learned desirability that the ant at current node i intends to go through neighbor n' toward node j is indicated by $\tau_{kn'}$ of P_k^{ij}.

Define $I_{\inf_k} = \mu_k - z\left(\sigma_k/\sqrt{w}\right)$ and $I_{\sup_k} = \mu_k + z\left(\sigma_k/\sqrt{w}\right)$, where w is the number of samples to organize X_k^{ij}, $z = 1/\sqrt{1-v}$, and v gives the chosen confidence level. We can then derive the following propositions of the above-mentioned data structures in X_k^{ij}.

Proposition. Suppose that the subcolony X_k^{ij} is sufficiently updated by a number of artificial ants. Then, the following is satisfied.

a) The measured delays of the paths from node i to node j by artificial ants, which are used for organizing X_k^{ij}, are statistically estimated from $I_{\inf_k}$ to $I_{\sup_k}$ with the probability of v.

Suppose that the backward ant strongly deposits the pheromone trails as the measured delay is close to $I_{\inf_k}$.

b) The strongest (highest) pheromone trail among P_k^{ij} heuristically estimates that there exists a path from node i to node j where the end-to-end delay is close to $I_{\inf_k}$.
Therefore, if a flow is transmitted by referring to the highest pheromone trail, then the expected end-to-end delay of the flow is close to $I_{\inf_k}$.

III. AntLBR — Formal Descriptions

We start by describing the definition used to determine the relevance between the measured delay t and subcolony X.

Definition 1. Measure of relevance — the measure of relevance between the measured delay t and subcolony X, which contains statistic $S \sim (\mu, \sigma)$, is defined as

$$r(t, X) = c_1\left(\frac{\mu}{t}\right) + c_2\left(\frac{I_{\sup} - I_{\inf}}{\left(I_{\sup} - I_{\inf}\right) + (t - \mu)}\right),$$

where $c_1 + c_2 = 1$, $I_{\sup} = \mu + z\left(\frac{\sigma}{\sqrt{w}}\right)$, $I_{\inf} = \mu - z\left(\frac{\sigma}{\sqrt{w}}\right)$,

and w is the number of required samples to estimate the statistic S.

Here, $r(t, X)$ is used to evaluate how closely t is related with S in X. As $r(t, X) \rightarrow 1$, t is significantly related with X, and X is said to be *relevant* to t if the below is satisfied:

$$f(t, X) = \left|1 - r(t, X)\right| < \beta,$$

where β is the boundary factor to determine the relevance.

We assume that if X is relevant to t, $r(I_{\sup}, X) \le r(t, X) \le r(I_{\inf}, X)$ almost surely holds because $I_{\sup}$ and $I_{\inf}$ represent two extreme bounds of measured delays with respect to the given confidence level v.

1. Self-Organized Management of Subcolonies

At regular intervals, a number of artificial forward ants depart asynchronously toward a randomly chosen destination node. Suppose that source node s transmits a forward ant toward destination node d. The forward ant explores a path and gathers the status of all nodes along the explored path. Let us assume the gathered information by the forward ant to be $\{I_k, t_k, k=0, 1, \ldots, n\}$, where I_k is the k-th intermediate node ($I_0=s$, $I_n=d$) along the explored path and d_k is the arrival time at I_k. The elapsed delay along the subpath from I_i to I_j is $t_{i\rightarrow j}$ ($t_{i\rightarrow j}=t_j-t_i$). When receiving the forward ant with the gathered information, destination node d generates a backward ant and transfers the information into the backward ant; the backward ant moves backward to source node s [13].

$\{I_k, t_k, k = 0, 1, \cdots, n\}$: the gathered information by ants.
I_k : the k-th intermediate node along the explored path
(I_0 = source node, I_n = destination node).
t_k : the arrival time at I_k.
Require: $\{I_k, t_k, k = 0, 1, \cdots, n\}$
1: **for** $j = i+1$ to $j = n$ **do**
2: $t_{i\rightarrow j} = t_j - t_i$ // the elapsed delay from I_i to I_j.
3: $X_*^{ij} \leftarrow$ ***selectSubColony*** $\left(t_{i\rightarrow j}, X^{ij}\right)$
4: **if** X_*^{ij} does not exists, then
5: ***makeNewSubColony*** $\left(t_{i\rightarrow j}, X^{ij}\right)$
6: **else**
7: ***updateDataStructures*** $\left(t_{i\rightarrow j}, X_*^{ij}\right)$
8: **end if**
9: **end for**

Fig. 3. Pseudocode for management of subcolonies.

Operation Concept — Figure 3 illustrates the pseudocode for the management of the organized subcolonies when the backward ant arrives at I_i. When receiving the backward ant with the set of information $\{I_k, t_k\}$, I_i selects the subcolony that has a significant relevance to $t_{i\rightarrow j}$ (lines 2 and 3). If a relevant subcolony does not exist (line 4), I_i begins the process of generating a new subcolony (line 5). Otherwise, I_i updates the data structures of the selected subcolony (lines 6 and 7). All pheromone trails in the subcolonies experience the evaporation process, and subcolonies hence delete themselves if not used during some of the periods. Formal descriptions of each process are provided below.

selectSubColony — Let us define Ψ^{ij} as the set of relevant subcolonies of I_i, that is, $f(t_{i\rightarrow j}\ X_k^{ij}) < \beta$ holds with regard to $X_k^{ij} \subset \Psi^{ij}$. Then, I_i selects the subcolony to be updated as follows:

$$X_*^{ij} = \arg\min_{X_k^{ij} \subset \Psi^{ij}} f\left(t_{i\rightarrow j}, X_k^{ij}\right).$$

This equation implies $0 \le f\left(t_{i\rightarrow j}, X_*^{ij}\right) \le f\left(t_{i\rightarrow j}, X_k^{ij}\right) \le \beta$,

where $X_k^{ij} \subset \Psi^{ij}$ and $X_k^{ij} \ne X_*^{ij}$; $f(\bullet) \rightarrow 0$ is equivalent to $r(\bullet) \rightarrow 1$. Accordingly, $t_{i\rightarrow j}$ is significantly relevant to X_*^{ij}.

makeNewSubColony — If X_*^{ij} does not exist (that is, $\Psi^{ij} = \phi$), I_i decides to make a new subcolony and stores the delay information $t_{i\rightarrow j}$ and the identifier of visited neighbor node I_{i+1} into the memory of I_i. If receiving several backward ants having similar but differential delays in the near future, I_i

makes a new subcolony X^{ij}_{new}, which is organized by the above stored information.

Based on the definition of "measure of relevance" and the process of making a new subcolony, we can draw Corollary 1.

Corollary 1a. If the value of β (β<1) is small, a node is forced to generate more subcolonies.

Corollary 1b. The more frequently the network environment changes, the more subcolonies are generated.

Proof. A new subcolony is generated if $|1-r(\bullet)|>\beta$ holds, that is, is not relevant. We can consider the following.

1a) β — It directly affects the probability of satisfying $|1-r(\bullet)|>\beta$. As β approaches zero, $r(\bullet)\to 1$ should be held. Accordingly, as seen in Definition 1, the only information that is significantly related with the existing subcolonies is considered relevant. If a relevant subcolony does not exist, the process of making a new subcolony proceeds.

1b) Consider a single path. In a static network environment, almost the same information is reflected for all delays; hence, a single subcolony may be generated. However, as the network environment begins to change, the gathered information may be more widely dispersed according to the traffic conditions. Accordingly, even in the same source-destination pair, more subcolonies related to the dispersed information are generated. □

updateDataStrucutures — If X^{ij}_{*} is successfully selected, I_i updates S^{ij}_{*} in X^{ij}_{*} as follows:

$$\mu_* \leftarrow \mu_* + \delta\left(t_{i\to j} - \mu_*\right),$$

$$\sigma_*^2 \leftarrow \sigma_*^2 + \delta\left(\left(t_{i\to j} - \mu_*\right)^2 - \sigma_*^2\right),$$

where the factor δ weighs the number of most recent samples that will actually affect the average [13].

The pheromone trails $\tau_{(*)n'}$ indicate that the ant at I_i has learned to and intends to go through neighbor node n' ($n'=I_i+1$) toward I_j. When receiving the backward ant via its neighbor n', I_i updates $\tau_{(*)n'}$ with

$$\tau_{(*)n'} \leftarrow \tau_{(*)n'} + r\left(t_{i\to j}, X^{ij}_{*}\right).$$

Note that as the delays measured by the artificial ants approach I_{inf_*}, more pheromone trails are accumulated.

deleteSubColony — At regular intervals, all pheromone trails experience the evaporation process as follows:

$$\tau \leftarrow (1-\rho)\times\tau,\ \ \rho\in(0,1].$$

It favors the obliteration of poor choices made in the past.

Pheromone evaporation also provides the criteria for deleting unused subcolonies. If the value of $\sum\tau$ of P in X is below the specific threshold, X deletes itself. Because the decrease of $\sum\tau$ implies that X is no longer relevant to the measured delay, all pheromone trails in P of X experience the evaporation without the deposit.

We can draw Corollary 2, as follows.

Corollary 2. The total number of managements made by one artificial ant amounts to $\sum_{i=1}^{n}(n-i)$, where n is the number of hops along the path explored by the ant.

Proof. The above-mentioned management processes are performed on all nodes along the path taken by the backward ant; each node I_i (i=0, 1,…, n–1) iteratively updates the data structures of X^{ij}_{k} for all j (j=i+1,…, n). After finishing all update processes, I_i transmits the backward ant to node I_{i-1}; I_{i-1} also performs the above-mentioned update processes. Accordingly, the number of management decisions made by an artificial ant amounts to $\sum_{i=1}^{n}(n-i)$. □

In this manner, by using a number of artificial ants, the generation, updating, and deletion of the subcolonies occurs automatically to promptly adapt to the ever-changing network status without any supervised control, that is, the subcolonies are self-organized.

2. Uses of Self-Organized Subcolonies

This subsection provides formal descriptions focusing on how to achieve load balancing while accommodating the delay-guaranteed services by making use of subcolonies.

Suppose that for each node $i, j \in V$ (V is the vertex set) and k=1, 2,…, Q^{ij} (Q^{ij} is the number of organized subcolonies in node i for destination j), $S^{ij}_{k} \sim (\mu_k, \sigma_k)$ and P^{ij}_{k} in subcolony X^{ij}_{k} are sufficiently updated by a number of artificial ants. We start by shedding light on the properties of the subcolony in Lemma 1.

Lemma 1. If $\tau_{kn'}\neq 0$ holds for a specific neighbor $n'\in N_i$, where N_i is the set of neighbors and node i transmits an incoming flow through n' by referring to $\tau_{kn'}$ toward destination j, then the following holds.

Lemma 1a. The estimated elapsed delay of the flow along the path from node i to node j is not greater than $I_{\text{sup}_k} = \left(\mu_k + z\dfrac{\sigma_k}{\sqrt{w}}\right)$.

Suppose that $\tau_{kn'}$ is the highest pheromone trail among P^{ij}_{k}.

Lemma 1b. Then, the estimated elapsed delay along the path to be taken by the flow is close to I_{inf_k}.

Proof. The existence of X^{ij}_{k} implies that $\sum_{n'\in N_i}\tau_{kn'}\neq 0$ (if $\sum(\bullet)$=0, X^{ij}_{k} should be deleted); $\sum(\bullet)\neq 0$ implies that a number of artificial ants, which organize to X^{ij}_{k}, explore arbitrary paths from node i to node j with relevant delays $t_{i\to j}$ and deposit the pheromone trails.

a) Because $t_{i\to j}$ is relevant to X^{ij}_{k}, $r\left(I_{\text{sup}_k}, X^{ij}_{k}\right)\leq r\left(t_{i\to j}, X^{ij}_{k}\right)\leq r\left(I_{\text{inf}_k}, X^{ij}_{k}\right)$ holds, and $t_{i\to j}\leq I_{\text{sup}_k}$ is

Require: $F_{s\to d}$ with d_{req} arrives at node i.
1: **while** node i $\neq$ destination node d **do**
2: **if** node i = source node s **then**
3: ***callAdmissionControl*** (d_{req}, X^{id})
4: **end if**
5: $d_m = d_{\text{req}} - d_{s\to i}$
6: X_*^{id} = ***selectSubColonyForFlow*** $\left(d_m, X^{id}\right)$
7: neighbor = ***selectNeighborNode*** $\left(P_*^{id}, X_*^{id}\right)$
8: ***sendNeighbor*** (node i, neighbor)
9: node i $\leftarrow$ neighbor
10: **end while**

Fig. 4. Pseudocode to find path for flow $F_{s\to d}$ with delay requirement d_{req}.

hence satisfied. This denotes that all artificial ants that organize X_k^{ij} explore the paths from node i to node j with delays of less than $I_{\sup_k}$. Because the path chosen by referring to $\tau_{kn'}$ is one of the paths taken by these ants, the delay of the path should be less than $I_{\sup_k}$.

b) Here, $\tau_{kn'}$ can gain more pheromone trails as higher values of $r(\bullet)$ are obtained. To obtain a higher $r(\bullet)$, as shown in Definition 1, the delays measured by the ants should be close to $I_{\inf_k}$ (if the delays are less than $I_{\inf_k}$, the delays are no longer relevant to X_k^{ij}). Accordingly, if $\tau_{kn'}$ is the highest pheromone trail, the estimated delay of the incoming flow is close to $I_{\inf_k}$. □

Lemma 1 forms basic properties provided by the subcolony. Suppose that a new flow $F_{s\to d}$ from source node s to destination node d with delay requirement d_{req} arrives. The AntLBR algorithm finds the path, which guarantees the requested delay of the flow using the basic properties of the subcolony.

Operation Concept — Figure 4 shows the pseudocode to be performed on a node to choose a neighbor for $F_{s\to d}$. First, $F_{s\to d}$ arrives at node i; if node i is source node s, it performs an admission control for $F_{s\to d}$ (lines 2 through 4). If $F_{s\to d}$ is accepted, node i calculates the remaining delay requirement d_m (d_m=$d_{\text{req}} - d_{s\to i}$, where $d_{s\to i}$ is the experienced delay of $F_{s\to d}$ from source node s to node i and $d_{s\to s}$=0) (line 5). To discriminate the delay $t_{i\to j}$, which is gathered by the artificial ant, we use $d_{i\to j}$ for the experienced delay of the flow. Node i then chooses the subcolony X_*^{id}, which mostly likely guarantees d_m (line 6) and transmits $F_{s\to d}$ through the neighbor by referring to the highest pheromone trail among P_*^{id} of X_*^{id} (lines 7 and 8). These processes (lines 5 through 9) repeat until $F_{s\to d}$ arrives at destination node d.

Before describing each step, we additionally define a new function $g(d_m, X)$ as below:

$$g\left(d_m, X\right) = c_1\left(\frac{I_{\sup}}{d_m}\right) + c_2\left(\frac{I_{\sup} - I_{\inf}}{\left(I_{\sup} - I_{\inf}\right) + \left(d_m - I_{\sup}\right)}\right),$$

where all variables are the same as those in $r(t, X)$. Theorem 1 provides the condition to guarantee d_{req} of $F_{s\to d}$.

Theorem 1 (Delay Guarantee). Let us assume that $\exists X_k^{sd} \in X^{sd}$ satisfies $g(d_{\text{req}}, X_k^{sd}) \le 1$. If $F_{s\to d}$ is routed over the neighbor by referring to the highest pheromone trails in P_k^{sd} of X_k^{sd}, then d_{req} of $F_{s\to d}$ is statistically guaranteed.

Proof. To achieve $g(d_{\text{req}}, X_k^{sd}) \le 1$, $d_{\text{req}} \ge I_{\sup_k}$ should be satisfied. Note that $I_{\sup_k}$ denotes the expected worst case end-to-end delay when an incoming flow is serviced by X_k^{sd} (Lemma 1a). Lemma 1b indicates that if $F_{s\to d}$ is routed to the neighbor chosen by referring to the highest pheromone trail, its estimated end-to-end delay is close to $I_{\inf_k}$. Accordingly, d_{req} of $F_{s\to d}$ is statistically guaranteed ($d_{\text{req}} \ge I_{\sup_k} \ge I_{\inf_k}$). The subcolony X_k^{sd}, which satisfies Theorem 1, is said to be a feasible solution to guarantee the d_{req} of $F_{s\to d}$. □

On the basis of the aforementioned theorem, we provide the formal description of each step listed in Fig. 4.

callAdmissionControl — Theorem 1 can provide the call admission control condition of $F_{s\to d}$. Suppose that $F_{s\to d}$ arrives at source node s. If $g(d_{\text{req}}, X_k^{sd}) \le 1$ does not hold for all subcolonies X_k^{sd}, k=1, 2,…, Q^{sd}, where Q^{sd} is the number of organized subcolonies, source node s rejects $F_{s\to d}$. This result is obvious because this case implies that none of the paths explored by artificial ants guarantee d_{req} of $F_{s\to d}$.

selectSubColonyForFlow — Let us define Φ^{id} as the set of subcolonies in node i, which are feasible solutions to guarantee d_m (d_m=d_{req}–$d_{s\to i}$). That is, $g(d_m, X_k^{id}) \le 1$ holds with regard to $X_k^{id} \subset \Phi^{id}$. For the accepted flow, node i determines the subcolony as follows:

$$X_*^{id} = \arg\min_{X_k^{id} \subset \Phi^{id}} g\left(d_m, X_k^{id}\right).$$

Refer to the equation of the $g(\bullet)$ function. The above equation implies that $I_{\sup_*} \le I_{\sup_k} \le d_m$ holds for all $S_k^{ij} \sim \left(\mu_k, \sigma_k\right)$ of X_k^{id}, where $X_k^{id} \subset \Phi^{id}$ and $X_k^{id} \ne X_*^{id}$. Lemma 1a indicates that $I_{\sup_k}$ is the estimated worst case delay when serviced by X_k^{id}. $I_{\sup_*} \le I_{\sup_k}$ implicitly indicates that the ***selectSubColonyForFlow*** function selects the subcolony among Φ^{id} that is most likely to have the least delay.

selectNeighborNode — If successfully selecting X_*^{id}, node i chooses the highest pheromone trail $\tau_{(*)n'}$ among P_*^{id} of X_*^{id} and forwards the flow to neighbor n', which is referred to by $\tau_{(*)n'}$. As shown in Lemma 1b, if the flow is forwarded by $\tau_{(*)n'}$, then its estimated end-to-end delay is close to $I_{\inf_k} \le I_{\sup_k} \le d_m$. That is, d_m is statistically guaranteed.

In this manner, $F_{s\to d}$ is transmitted until it reaches destination node d.

AntLBR chooses paths for incoming flows in a hop-by-hop manner. As described in [17], in a hop-by-hop routing, it is crucial to construct a loop-free path. Theorem 2 guarantees that the constructed path by the pseudocode in Fig. 4 will be loop-free. Let $p^*_{s\to d}$ be the path for $F_{s\to d}$, which is constructed by concatenating all chosen nodes. That is, $p^*_{s\to d}=\{I_0\ldots, I_i,\ldots, I_n\}$, where I_i is the i-th visited node (I_0=source s, I_n=destination d). The end-to-end delay of path p is defined as $D(p)$. Then, Lemma 2 always holds.

Lemma 2. If $\hat{p}_{i\to d}$ is the subpath from I_i to d of $p^*_{s\to d}$, $D(\hat{p}_{i\to d}) > D(\hat{p}_{j\to d})$ holds for any $i<j$.

Proof. Refer to the equation of ***selectSubColonyForFlow*** function in Fig. 4. The node chooses the subcolony minimizing $g(\bullet)$. That is, packets traverse along the least delay path. Since $D(\hat{p}_{i\to d}) = D(\hat{p}_{i\to j}) + D(\hat{p}_{j\to d})$ and $D(\bullet)>0$, Lemma 2 holds. □

Theorem 2 (Loop Prevention). The path $p^*_{s\to d}$ chosen by the pseudocode in Fig. 4 is a loop-free path.

Proof. We can prove it by contradiction. Suppose that $p^*_{s\to d}=\{I_0,\ldots, I_i,\ldots, I_j, I_k,\ldots, I_n\}$ is not a loop-free path to be concatenated by all chosen nodes and both I_i and I_k denote the same node, that is, a loop is formed along $\hat{p}_{i\to k}$. By Lemma 2, $D(\hat{p}_{i\to d}) > D(\hat{p}_{j\to d}) > D(\hat{p}_{k\to d}) = D(\hat{p}_{i\to d})$ holds, and it contradicts Lemma 2. Therefore, $p^*_{s\to d}$ is a loop-free path. □

By making use of the self-organized subcolonies, AntLBR provides load balancing for the network, as shown below.

Theorem 3 (Load Balancing). AntLBR naturally provides load balancing without any supervised control.

Proof. The load balancing performance depends heavily on 1) how many alternate paths exist and 2) how well the traffic distributes on these alternate paths.

1) AntLBR determines the path for an incoming flow in a hop-by-hop manner. Each node chooses the neighbor for the flow by referring to the organized subcolonies. For example, let $p^*_{s\to d}=\{I_0,\ldots, I_i,\ldots, I_n\}$ be the constructed path for the flow and Q^{id} be the number of organized subcolonies of I_i from I_i toward destination node d. Then, the number of possible candidate paths is up to $\prod_{I_i\in p^*_{s\to d}} Q^{id}$; $p^*_{s\to d}$ is chosen as the best path among these candidate paths. To achieve a desirable performance for load balancing, it is beneficial to have more subcolonies. Corollary 1 describes that AntLBR has many subcolonies. In particular, as shown in Corollary 1b, AntLBR generates more subcolonies when the network environments change frequently. There is a strong indication that AntLBR effectively resolves the problem of congested traffic by autonomously generating more subcolonies.

2) Suppose that X^{ij}_k is relevant to $t_{i\to j}$, which is gathered by an artificial ant, and $\tau_{kn'}$ in P^{ij}_k of X^{ij}_k is the highest pheromone trail. Therefore, incoming flows, which request delay requirements that are feasible solutions to guarantee X^{ij}_k, are transmitted along the link $l_{in'}$, referred to by $\tau_{kn'}$. As time passes, $l_{in'}$ becomes congested; the congestion of $l_{in'}$ causes $t_{i\to j}$ to increase, and $t_{i\to j}$ is hence no longer relevant to X^{ij}_k ($t_{i\to j}$ may be used to update another relevant subcolony). It implies that $\tau_{kn'}$ of X^{ij}_k does not gain the pheromone deposits. Accordingly, as time passes, incoming flows choose the link (not $l_{in'}$) that has become marked as having the highest pheromone trail. In this manner, when the congestion occurs, the traffic is distributed on other links. □

On the basis of 1) and 2), AntLBR provides the load balancing effect in an efficient and autonomous manner.

3. Self-Organizing Networking Properties

In this subsection, we exhibit the self-organizing properties achieved by AntLBR. The basic self-organizing properties are the continuous self-learning of the network status and the self-adjusting to incorrect measures, as shown below.

Self-learning is the capability to continuously learn the changing network status. Because the pheromone trails are accumulated whenever the backward ant arrives, the accumulated pheromone trails include not only the information from previous ants but also instantaneous information from the current ants. Accordingly, AntLBR can learn and adapt to the changing network status more accurately.

Self-adjusting is the capability to stand against the inaccurate information. In AntLBR, the ant replaces the inaccurate information with a small amount of pheromone trails; the point to be considered is that the substituted pheromone trails are considerably small compared to the accumulated pheromone trails. Since the accumulated pheromone trails determine the routing path, the impact of information that is inaccurate can be mitigated.

Based on these properties, AntLBR provides the following self-organizing networking properties.

Theorem 4 (Self-Organization). AntLBR provides the following networking properties in a very flexible way:

a) AntLBR can naturally resist multiple network failures;

b) AntLBR can promptly adapt to changing environments;

c) AntLBR can be scaled for larger networks, incurring additional network overhead.

Proof. AntLBR provides the following self-organizing properties by making use of the aforementioned networking properties, that is, self-learning and self-adjusting.

a) Network Robustness — Network failures cause the artificial ants present during the failures to be lost; hence, the paths in which the failures occur do not gain pheromone deposits. Only the paths free from failure gain the deposits. Therefore, the flow naturally chooses a failure-free path with strong pheromone concentrations.

b) Network Flexibility — It follows from the algorithms that make, update, and delete the subcolonies, as described in subsection III.1.

c) Network Scalability — As the network grows, the number of hops of a path also grows; as the hops increase, the total number of management decisions made by one artificial ant also grows (by Corollary 2, it amounts to $\sum_{i=1}^{n}(n-i)$, where n is the number of hops of the path). Accordingly, with only minimal additional network overhead, AntLBR can be extended for larger networks. □

IV. AntLBR — Performance Evaluations

AntLBR can achieve a desirable performance in terms of not only load balancing but also a high degree of self-organization. In this section, we evaluate these characteristics of AntLBR using the ns-2 simulator [18].

Here, a 14-node NSFNet topology is considered, as illustrated in Fig. 5, where the rate of the link is assumed to be 100 Mbps. The propagation delay between nodes is proportional to the actual distance shown in Fig. 5. To capture the ever-changing network characteristics, we try to introduce a randomness to the network: During the simulation time, we generate tens of thousands of flows; The source node and destination node of each flow are randomly chosen; The interarrival time, holding time, and rate of the flow are exponentially distributed; The size and interarrival time of the packets in the flow are also exponentially distributed. The delay requirements of the flows are taken as independent and uniformly distributed in the range of 50 ms to 150 ms. The parameters for AntLBR used in simulations are described in Table 1.

To minimize the performance variations owing to the ever-changing state of the network, we run the simulations several times and average them in a 95% confidence interval.

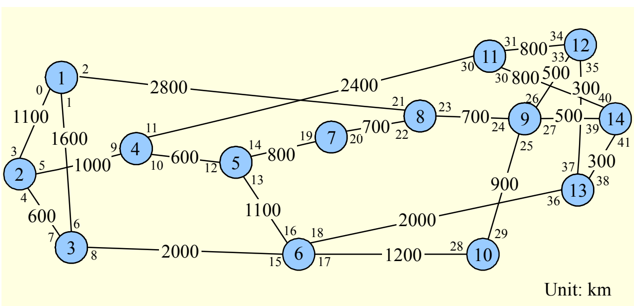

Fig. 5. Simulated network.

1. Performance Evaluation of AntLBR

To show the performance of AntLBR, we compare it with the well-known least delay path routing (LPR) and multipath routing (MPR). Note that AntLBR is proposed to achieve load balancing while accommodating the delay-sensitive services. For this reason, so as to compare the performance under the same condition, previously used routing schemes must be modified. Because these schemes should guarantee the delay requirements, they use probing packets to investigate whether the delay requirement of an incoming flow can be guaranteed. Additionally, in the case of MPR, all packets belonging to the same flow are routed over the same path to avoid a packet resequencing problem.

LPR — The source sends a probing packet for an incoming flow along the least delay path. If the probing packet with an ACK returns, the flow is permissible and transmitted along the path.

MPR — The source simultaneously sends the probing packet along predefined multiple paths. If one of the probing packets with ACK is received, the flow is permissible and transmitted along the path taken by the first received probing packet.

We also want to compare the performance of the AntLBR algorithm with two-phase routing. However, we cannot change two-phase routing to be capable of accommodating delay-sensitive services due to the following reason. As described in [9], the end-to-end delay of two-phase routing might be about twice that of shortest path routing. That is, the two-phase routing scheme may not be suitable for delay-sensitive services. However, if we insist on making a modified version of two-phase routing for delay-sensitive services, two-phase routing may be implemented by using multipath routing, which is modified in this paper. However, due to the delay constraint,

Table 1. Parameters for AntLBR used in simulations.

Parameter	Symbol	Value
Launching rate of artificial ants	Δt	30 ms, 50 ms, 100 ms
First weight factor in $r(\bullet)$	c_1	0.7
Second weight factor in $r(\bullet)$	c_2	0.3
Boundary to determine relevance	β	0.2
Confidence level in $r(\bullet)$	υ	0.96
Pheromone evaporation period	t_p	0.5 s
Pheromone evaporation rate	ρ	0.2
Weight of exponential mean coefficient	δ	0.005

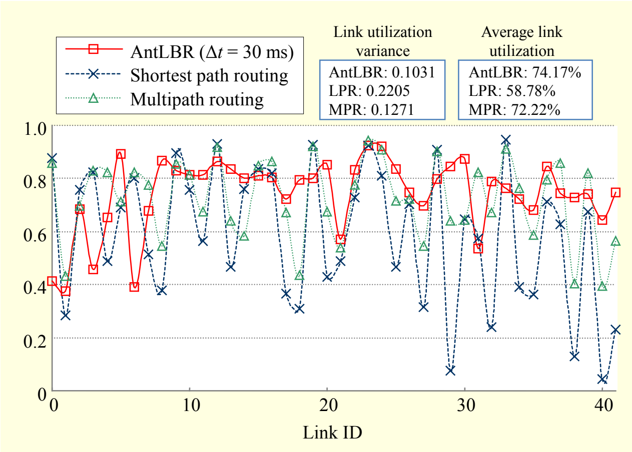

Fig. 6. Link utilizations for load balancing routing algorithms. Link IDs on x-axis are designated by numbers identifying links in Fig. 5.

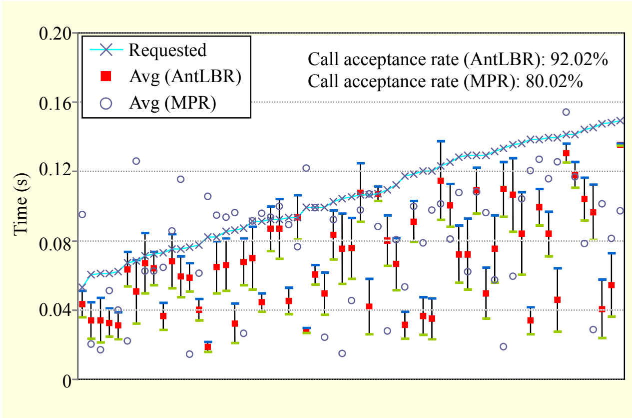

Fig. 7. End-to-end delays of flows in source-destination node pair.

the performance quality may be lower.

Figure 6 describes the utilization of all links in terms of the above-mentioned routing schemes. If all links have similar utilization, we can say that the traffic is well balanced in the network. The red line indicates the utilization achieved by AntLBR. We can easily know that AntLBR has evenly distributed link utilizations compared to other load balancing routings. To describe it simply, we indicate the standard deviation of all link utilizations. Note that having a smaller one is equivalent to sustaining a more well-balanced network. Additionally, as the network is well balanced, more flows can be accepted, which makes it possible to achieve a high throughput.

As shown in Fig. 6, LPR has the lowest utilization because LPR simply rejects the flow if the least delay path is not sufficient to accommodate the delay requirement of an incoming flow. The under-utilization of LPR is greatly enhanced by giving an opportunity to choose one or more alternate paths (MPR). AntLBR achieves the highest link utilizations owing to the rapidity of its self-adaptability: it always tries to adapt the subcolony best fit for the current changing environments; hence, it effectively resolves the congested traffic.

Figure 7 illustrates the end-to-end delays of generated flows in a specific source-destination node pair. The x-axis and y-axis denote the IDs of the generated flows and their experienced end-to-end delays, respectively. The requested delay of a flow is indicated with an "X." Each experienced delay of the flow is indicated with a red opaque square in terms of AntLBR and with the outline of a circle in terms of MPR. In the case of AntLBR, we also plot the upper and lower bounds of the delays in a 95% confidence level. For convenience, the end-to-end delays are sorted in ascending order of the requested delay requirements.

Figure 7 describes the performance in terms of how well each routing technique guarantees the requested delay requirements of incoming flows. As seen in the figure, in the case of MPR routing, delay violations frequently occur. At the time when a probing packet is transmitted, the path chosen by the probing packet can be capable of guaranteeing the delay requirement of the flow. However, the real experienced delay of the flow is not so. It is natural because the chosen path for the flow may rapidly change into another state due to the variability of the traffic. However, even in that case, AntLBR tries to sustain the delay performance to a certain degree.

To exemplify, suppose that a specific link is chosen for a flow with a delay requirement and another new incoming flow starts to use this specific link. The arrival of the new flow may cause a violation of the requirement in the delay-sensitive flow. A number of artificial ants may learn this violation of the flow and let the flow take another path that is capable of guaranteeing the delay requirement.

Delay violations also occur in AntLBR. They only occur when the launching rate of artificial ants is too small to learn the current changing network status. In this case, the violation intermittently occurs. This is the reason why we plot the upper and lower bounds of delays of flows in AntLBR.

The average end-to-end delays tend to increase in AntLBR when the delay requirements increase, which differs from the results found in previous research, wherein the end-to-end delays remained constant irrespective of the delay requirements. The reason that the end-to-end delays tend to increase in AntLBR is that AntLBR tries to evenly distribute the congested traffic in the network, hence accepting more flows. As seen in the call acceptance rate of the figure, AntLBR accepts over 10% more flows than MPR. Accordingly, the number of experienced average end-to-end delays of the flows increases but does not violate the delay requirements.

2. Performance Evaluation of Self-Organization Capabilities

This subsection shows how well AntLBR resolves the congested traffic caused by the sudden input of highly bursty traffic. Note that MPR and two-phase routing are static routing schemes in the sense that they do not change the routing paths, regardless of the variety of the traffic, without the aid of such other techniques as MPLS-TE; hence, they do not effectively handle unexpected congestion.

Through a case study, we illustrate how well AntLBR resolves the congested traffic caused by the sudden arrival of highly bursty traffic. To show the performance, additional bursty traffic is intentionally injected into the path from node 4 to node 9 during the period of 200 s to 207 s. The rate of the bursty traffic is 20 Mbps, and it hence occupies 20% of the link capacity. Figure 8 illustrates the utilizations of link (4:11), link (5:6), and link (5:7) with respect to the simulation time.

Bursty traffic generally causes some specific links to experience congestion. If AntLBR is flexible to the changing environments, it may solve the congestion by evenly distributing the bursty traffic on adjacent links. The efficiency is determined by how fast it solves the congestion. Figure 8 shows that AntLBR can solve the congestion. At time 200 s, it transmits the bursty traffic through the link between node 4 and node 11. The rapid increase of traffic carried on the link increases the network delay; hence, the deposited pheromone trails rapidly decrease. The decrease of pheromone trails forces the bursty traffic to be transmitted on other links. In Fig. 8, at time 201 s, the bursty traffic is distributed on the link between node 5 and node 7. In this manner, at time 202 s, the traffic is also distributed on the link between node 5 and node 6.

A sudden arrival of bursty traffic transforms a relatively static environment into a frequently changing environment, which results in the generation of more subcolonies. The increase in subcolonies enables AntLBR to efficiently resolve the congested traffic by allowing the traffic to be autonomously and evenly distributed on more available paths.

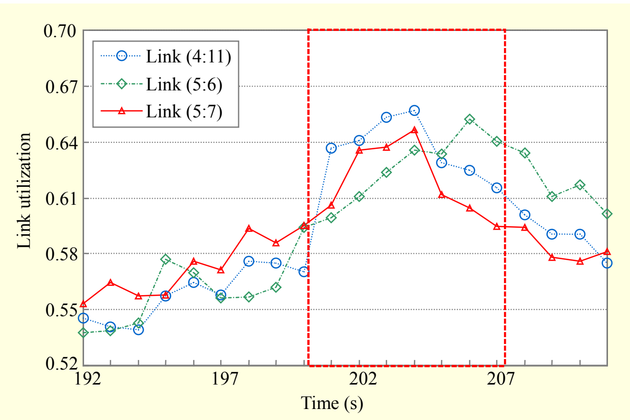

Fig. 8. Utilizations of link (4:11), link (5:6), and link (5:7) with respect to simulation time. Additional traffic (20 Mbps) from node 4 to node 9 is intentionally generated from 200 s to 207 s.

V. Conclusion

Although load balancing routing methods have matured considerably over the decades, there are few works on the type of ever-changing network in which the traffic is extremely dynamic and the network topologies frequently change. In this paper, we proposed a bio-inspired load balancing routing for such a network. The proposed AntLBR promptly adapts to the changing network status by making use of a number of artificial ants. The adapted AntLBR greatly enhances the load balancing effect and maximizes the network throughput while accommodating the delay requirements of emerging services. Simulation results show that AntLBR not only evenly distributes the traffic in the network but also effectively resolves the congestion caused by the sudden arrival of bursty traffic.

References

[1] A.C. Weaver and B.B. Morrison, “Social Networking,” *IEEE Comput.*, vol. 41, Feb. 2008, pp. 97-100.

[2] ITU-T Recommendation Y.2012, “Functional Requirements and Architecture of the NGN,” Apr. 2000.

[3] A. Medina et al., “Traffic Matrix Estimation: Existing Techniques and New Directions,” *ACM SIGCOMM*, Aug. 2002.

[4] M. Kodialam et al., “Traffic-Oblivious Routing for Guaranteed Bandwidth Performance,” *IEEE Commun. Mag.*, vol. 45, no. 4, Apr. 2007, pp. 46-51.

[5] M. Kodialam et al., “Oblivious Routing of Highly Variable Traffic in Service Overlays and IP Backbones,” *IEEE/ACM Trans. Netw.*, vol. 17, no. 2, Apr. 2009, pp. 459-472.

[6] M. Antic and A. Smiljanic, “Routing with Load Balancing: Increasing the Guaranteed Node Traffics,” *IEEE Commun. Lett.*, vol. 13, no. 6, June 2009, pp. 450-452.

[7] M. Antic et al., “Two Phase Load Balanced Routing Using OSPF,” *IEEE J. Sel. Areas Commun.*, vol. 28, no. 1, Jan. 2010, pp. 51-59.

[8] D. Applegate and E. Cohen, “Making Intra-domain Routing Robust to Changing and Uncertain Traffic Demands: Understanding Fundamental Tradeoffs,” *ACM SIGCOMM*, Aug. 2003.

[9] M. Kodialam et al., “Advances in Oblivious Routing of Internet Traffic,” *Performance Modeling and Engineering*, Z. Liu and C.H. Xia, Eds., NY: Springer, 2008, Chapter 5, pp. 125-146.

[10] E. Oki and A. Iwaki, “Fine Two-Phase Routing with Traffic

Matrix," *IEEE ICCCN*, Aug. 2009.

[11] C. Prehofer and C. Bettstetter, "Self-Organization in Communication Networks: Principles and Design Paradigms," *IEEE Commun. Mag.*, vol. 43, July 2005, pp. 78-85.

[12] E. Bonabeau et al., "Inspiration for Optimization from Social Insect Behaviour," *Nature*, vol. 406, July 2000, pp. 39-42.

[13] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, Cambridge, MA: The MIT Press, 2004.

[14] Y.M. Kim et al., "Ant Colony Optimization Based Self-Organizing QoS Framework in IP Networks," *IEEE Commun. Lett.*, vol. 14, Dec. 2010, pp. 1074-1076.

[15] Y.M. Kim et al., "Ant Colony Based Self-Adaptive Energy Saving Routing for Energy Efficient Internet," *Elsevier Comput. Netw.* vol. 56, no. 7, July 2012, pp. 2343-2354.

[16] J.H. Ahn and S.H. Kang, "NoC-Based SoC Test Scheduling Using Ant Colony Optimization," *ETRI J.*, vol. 30, no. 1, Feb. 2008, pp. 129-140.

[17] L. Zhenjiang et al., "Algebra and Algorithms for QoS Path Computation and Hop-by-Hop Routing in the Internet," *IEEE/ACM Trans. Netw.*, vol. 10, no. 4, Aug. 2002, pp. 459-472.

[18] The Network Simulator — ns-2. http://www.isi.edu/nsnam/ns/

Young-Min Kim received his MS and PhD in information and communication from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Rep. of Korea, in 2006 and 2011, respectively. In 2011, he joined ETRI, Daejeon, Rep. of Korea. His research mainly focuses on energy saving routing, network architecture, network protocols, and QoS framework in telecommunication networks. He is a member of the IEEE.

Hak Suh Kim received his MS from the Department of Computer Science, Chungbuk National University, Cheongju, Rep. of Korea, in 1999. Since 2000, he has been a senior researcher at ETRI, Daejeon, Rep. of Korea. Also, he is working toward his PhD at Chungbuk National University. His research mainly focuses on network architecture, network protocols, QoS, and traffic engineering.

Boo-Geum Jung received her BS from Busan National University, Busan, Rep. of Korea, in 1986 and her MS from Sookmyung Women's University, Seoul, Rep. of Korea in 1991 in computer science. In 1986, she joined ETRI and worked on the development of the TDX digital switching system, ATM switching systems, and QoS router systems. She is currently working on virtual private cloud networking technology in the Computing Network Research Department. Her main areas of research include network OSs and security.

Hea-Sook Park received her PhD in computer science from the Department of Computer Engineering, Chungnam National University, Daejeon, Rep. of Korea, in 2005. From 1994, she has been a principal member of the research engineering staff of ETRI, Daejeon, Rep. of Korea. She is currently developing flow-based gateway and router technology in the Future Communication Research Department. Her main areas of research include network QoS and traffic engineering.

Hong-Shik Park received his BS from Seoul National University, Seoul, Rep. of Korea, in 1977 and his MS and PhD from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Rep. of Korea, in electrical engineering in 1986 and 1995, respectively. In 1977, he joined ETRI and worked on the development of the TDX digital switching system family, including TDX-1, TDX-1A, TDX-1B, TDX-10, and ATM switching systems. In 1998, he moved to the Information and Communications University, Daejeon, Rep. of Korea, as a member of the faculty. Currently, he is a professor in the Department of Electrical and Electronics Engineering, KAIST. His research interests are network architecture, network protocols, and performance analysis of telecommunication systems. He is a member of the IEEE, IEEK, and KICS of the Rep. of Korea.