Proceedings of the 2005 International Conference on Simulation and Modeling
V. Kachitvichyanukul, U. Purintrapiban, P. Utayopas, eds.

THE PARALLEL DYNAMIC ROUTING SIMULATION BASED ON ANTNET

Kittipong Theerawatsathein

Department of Computer Engineering,
King Mongkut's University of Technology Thonburi,
Bangkok, Thailand

Tiranee Achalakul

Department of Computer Engineering,
King Mongkut's University of Technology Thonburi,
Bangkok, Thailand

ABSTRACT

This paper presents a new approach to parallel computing for the routing simulation of packet switched network. The simulation is based on AntNet. We studied the AntNet adaptive learning technique and redesign its communication methodology. The distributed simulation techniques, called asynchronous parallel discrete event simulation (PDES) is also presented. PDES is used for exchanging the routing and the data packets between nodes. To show the overall performance of the simulation, the network throughput and the average delay time are measured. The experiments on our parallel algorithm were performed on a network of Commercial-off-the-shelf (COTS) workstations, which is a cost-effective solution for parallel modeling. Our parallel model is geared toward practical applications for telecommunication industries.

1 INTRODUCTION

In today's technology, wireless communication network has gained popularity. Not only the technology of network switching and transmission are widely developed, but also the routing algorithms rapidly evolve to support the new network service demands.

The routing scheme in the communication network can be described as follows: Routing services start when a router receives data packets from a source node. A routing table stored at the router is then used to determine a communication path toward the destination. Subsequently, the data packets can be forwarded along the selected path. This process will be repeated until data packets reach the destination. From the previous literatures, the routing protocols can be categorized into, Distance Vector and Link State Protocol [Halabi, 1997]. The Distance Vector protocol exchanges the distance information (hop counts) between routers to search for the shortest path, while the Link State Protocol uses the status and bandwidth information.

The wireless network is a dynamic network topology, where traffic information changes consistently. Thus, traditional routing protocol, which is centralized in nature, is not suitable due to the scalability problem. The centralized model usually generates a lot of information packets causing a large amount of message passing between routers. Path establishment will consume a lot of time in updating the routing table and thus, affect the convergent time. If the routing information is updated too frequently, information used by the routers maybe outdated. The selected routes may then include congested or down paths.

The traditional routing protocols require the entire network routing information for path selection process. The protocols force a lot of overheads on updating these information when the network system changes frequently or increases abruptly in sizes. Moreover, the centralized nature creates the bottleneck problem. To overcome the drawbacks presented, adaptive and distributive routing algorithms are proposed. One of the technique is based on Ant Colony Optimization (ACO) algorithm [Dorigo and Di Caro, 1999]. The ACO algorithm employed the nature of swarm intelligence, which is a distributive multi-agent system. The swarm intelligence such as ant or honey bee can adapt their behaviors smoothly in corresponding to the environment where they live. In other words, they use environment factor that each individual encounters to determine a global solution. Applying the distributive routing scheme, we can avoid the problems presented previously and the overall performance will be improved. The traffic of a dynamic wireless network, can thus be controlled. In this work, we studied the Antnet system and designed a parallel discrete event simulation based on it.

2 RELATED RESEARCH

In recent literatures, the swarm intelligence based techniques are studied and implemented with many problems such as optimization and routing in telecommunication. Its main characteristic is to use interaction between simple agents and their environments to generate global solutions.

In Ant Colony Optimization technique, an artificial ant represents each agent. The main task of ants is collabora-

tively finding the minimum cost path on a graph using pheromone trails as a communication medium. Each ant uses probabilistic values, which are a function of pheromone and heuristic information (distance), to determine the next move. An ant will deposit a pheromone on its selected path that links the present and the next node. Any path with a high concentration of pheromone will be a preferred path for the ants that follow. Consequently, the shortest path will be generated.

The ACO algorithm is proposed for solving hard combinational optimization problems (NP-problem). It is suitable for the static problems (environment factors are stationary), such as the traveling salesman (Ant System (AS) [Dorigo, Maniezzo and Colorni, 1996]). Moreover, The algorithm is often used with the adaptive environment problem, such as the routing problem in telecommunication network, and Quality of services (QOS) on multimedia communication.

In previous literatures, the ACO algorithm, due to its distributive nature, is known to be used for network routing. For example, an adaptive ACO used for analyzing the routing structure of the British Synchronous Digital Hierarchy (SDH) network. The system is called Ant-Based Control (ABC) [Schoonderwoerd, Holland, Bruten and Rothkrantz, 1996.]. The ants in ABC move in only one direction and update the routing table at each passing node. The ants die when they reach the destination node.

Similar to ABC, the Antnet [Di Caro and Dorigo, 1998] system was adapted from ACO. However, the system takes into account the fact that the shortest path is not always optimum. Antnet includes traffic congestion in the routing scheme to ensure optimality making it appropriate for dynamic routing of the wireless technology.

The ACO-based algorithms are mostly compute intensive. In order to improve the performance, parallel techniques were explored. The goal is to reduce the computational workload on each computer by involving a cluster of computers. The first parallel version of adaptive ACO was applying to the traveling salesman problem on the Connection Machine CM-2 as presented in [Bolondi and Bondanza, 1993]. The parallelism was employed at the ant level. The work by Bullnheimer [Bullnheimer, Kotsis and Strauss, 1998] further increased the parallel performance by extending the parallelism to the sub-colony level. The works by Bullnheimer and Stützle [1998] compared the performance and efficiency of the synchronous versus the asynchronous parallel implementation [Antony, Piriyakumar, Louis and Levi, 2002]. The result showed that the asynchronous system yielded a better communication cost.

In our research, we chose to base our work on the AntNet system, due to its performance merits. A parallel simulation is designed. We expect our version of parallel AntNet system to be scalable and practical for real-time simulations.

3 ANTNET: THE SEQUENTIAL SIMULATION

AntNet is an adaptive routing algorithm based on ACO. The algorithm searches for a minimum cost path under stigmergy paradigm using a set of multi-agents, called ants. The ants can move forward and backward. Forward ants explore the traveling route and record the information about selected node and delay in the memory. These information can then be used to update the routing table and the path statistics during the backward trip. Each node is associated with 2 tables: a routing table, and a statistical local traffic table

- A routing table records a relationship between nodes and their outgoing links. Each record is a probabilistic value that refers to the goodness of the route selection. The records are generated from the probability model in the routing policy. The total probability, P_{nd} of any n links being selected on the route toward destination, d, is equal to 1.

$$\sum_{n=1}^{|N_k|} P_{nd} = 1 \qquad (1)$$

  Where N_k is the neighbors of node k

- The statistical local traffic table, $M(\mu_d,\sigma^2_d,W_d)$, stores the mean (μ_d), the variance (σ^2_d) and the observed best trip time (W_d) from a node to the destination, d.

The steps of Antnet adaptive routing can be described as follows:

1. Initializing the probabilistic routing table and interval time for releasing forward ants in each node. Choose a destination node randomly. Then, launches forward ants from each source node.
2. Each forward ant selects the neighboring node with the highest probability of being joined to the destination as its next node. Then the elapsed time needed for traveling between the present node and the selected node is pushed into the ant's memory stack.
3. During a route selection, a forward ant checks for duplicated nodes to prevent traveling in circle. If a cycle is detected, the ant will pop all the data from its stack and restarts routing. If a cycle occurs after half-life of an ant, the ant will be destroyed.
4. When a forward ant reaches a destination node, it will start to travel backward using the information in its stack in opposite direction.
5. Each backward ant updates a record in a routing table at visited nodes using information in its stack. The means and the variance on visited node will also be up-

dated using the trip time, t_{kd}, from node k to destination d as shown in eq 2 and 3.

$$\mu_d = \mu_d + \eta(t_{kd} - \mu_d) \quad (2)$$

$$\sigma^2_d = \sigma^2_d + \eta((t_{kd} - \mu_d)^2 - \sigma^2_d) \quad (3)$$

Where, η is the weighted factor which influences the number of effective samples, $eff \approx 5(1/\eta)$.

The preferred path ranks, r, calculated from reinforcement learning, also have an effect on the probability of the path being selected by a forward ant. The relationship is shown in equation 4.

$$P_{fd} = P_{fd}(1 - r) + r \quad (4)$$

The neighbors of selected node f is updated with equation 5.

$$P_{nd} = P_{nd}(1 - r), \; n \neq f \;, n \in N_k \quad (5)$$

The reinforcement learning, r, can be defined as a function of the trip time and the local traffic as shown in equation 6.

$$r = \frac{c_1(W_{best})}{T} + \frac{c_2(I_{sup} - I_{inf})}{(I_{sup} - I_{inf}) + (T - I_{inf})} \quad (6)$$

W_{best} is the best trip time to destination d. I_{inf} and I_{sup} is the lower and upper bound of the confidence interval. C_1 and C_2 are the weighting coefficients.

4 ASYNCHRONOUS PARALLEL SIMULATION MODEL

There are two types of packets; data and routing, randomly generated and transmitted during our version of AntNet simulation. The routing packets are used in finding the optimum path, while the data packets are used to simulate the actual packets traveling within the network. With the unpredictable network traffic of both types of packets, our AntNet system is considered a distributive simulation. Thus, the sequential discrete event simulation model does not correspond well to its behavior. A re-design of AntNet based on asynchronous parallel discrete event simulation (PDES) is appropriated. In this section, we proposed a design of such model.

4.1 Partitioning

In a non-stationary routing problem, the problem size has a great effect on the system scalability. In orther words, the execution time of the network simulation depends on the system workloads. Thus, partitioning and distributing workloads are processes of significance in the concurrent version of AntNet. Workload partitioning refers to the process of breaking down both the computation and simulated data into pieces. Task partitioning in our concurrent Antnet can be achieved in two domains: Functional and Data.

4.1.1 Partitioning in functional domain

The structure of an actual network composes of a set of routers. These routers are responsible for switching and routing operations in packets exchanging. In our simulation model, the routers are represented by *logical processes* or LP. In the network topology displayed in Figure 1, each node represents a LP and each link represents a communication path. The LP routes messages and manages message queue. Message routing concerns with two functions: finding an optimal path between any source and destination pair (forward ants), and recording a goodness of route selections in routing tables based on reinforcement learning (backward ants). The queue managing function, on the other hand, deals with the arrival and the departure of both data and routing packets, which are controlled by the event clock. In the distributed simulation, the LPs are mapped independently across a cluster of workstations as shown in Figure 1. At any given time, each LP handles one of the specific tasks described previously. The concurrency can, thus be achieved in the functional domain.

Moreover, the *locality of references* concept is applied to reduce the communication cost. Neighboring nodes in the topology are mapped into the same physical processing unit as they tends to exchange messages (data and routing packets). The single-link clustering method based on agglomerative algorithm [Jain and Dubes, 1948] is used in our work to construct the network topology data structure. Communication overhead can thus be minimized.

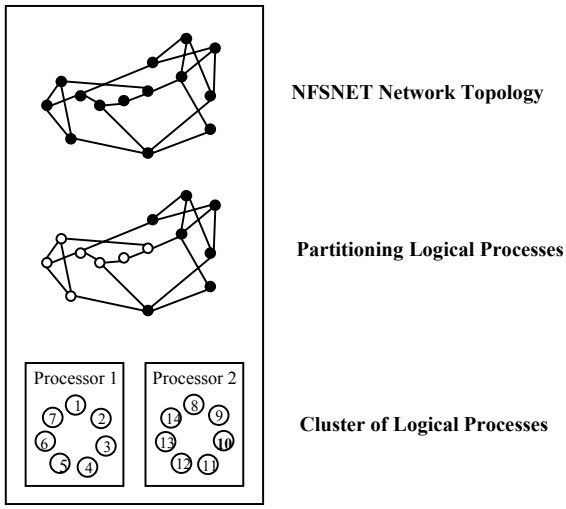

Figure 1: Partitioning network topology to logical processes on multiprocessors architecture.

4.1.2 Partitioning in data domain

The gloal routing table in the simulation model is represented by a *N*x*N* matrix, where *N* is the number of nodes in the network topology. To decompose the tasks in data domain, the concept of local routing is formed. Logical processes are grouped together using the clustering technique and each group conducts its own local simulation. The global routing table can, then, be divided into a set of local tables, where each table only keeps track of the LPs in the group as shown in Figure 2. The local routing table is a matrix of size *L*x*N*, where *L* is the number of communication links of a node.

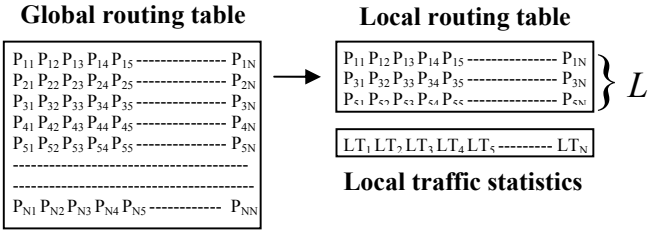

Figure 2: Partitioning a routing table.

4.2 Event synchronization with rollback

In a distributed simulation environment, activities occur based on an event scheduling and a message passing mechanisms. Logical processes (LPs) in the simulation can operate concurrently and independently. The proper order of the simulation events is maintained by the synchronization of message passing. The header of each message will be stamped with a *scheduling time*. These timestamps will be used to manage simulation event sequence. In our parallel simulation model, the optimistic synchronization method is used. The method does not enforce the *waiting rules* [Chandy and Misra, 1979], i.e. the LPs are allowed to process any message arrived at the node. The causality error can, thus occur. The optimistic method [Jefferson, 1985] then, tries to detect and recover from the error using a rollback mechanism. If the time stamped on the arriving message is prior to the current time, the mechanism will rollback the simulation to the stamped time based on sequence information in a message queue. Note that, the queue is implemented under the M/M/1 queuing model.

4.3 Antnet: the parallel and distributive system

Communications between nodes in our parallel AntNet simulation is based on a manager-worker paradigm. Initially, a manager constructs a global routing table, node identifiers, and a set of communicators for each worker. Then, it decomposes the tasks in both data and functional domains, and assigns a node identifier and local routing table to the workers using the clustering technique. During the simulation, the manager coordinates the operation by processing different types of messages. Table 1 shows the key message types.

Table 1: Message Management

Message type	Manger	Worker	Description
SETLINKQ	Recv	Send	Link queue status
GETLINKQ	Recv and Send	Send and Recv	Link queue status
SETTIME	Recv	Send	Local current event time
GETTIME	Send	Recv	Minimum current event time
FORWARDANT	Recv and Send	Send Recv	Forward ant packet
BACKWARDANT	Recv and Send	Send Recv	Backward ant packet
DATAPKT	Recv and Send	Send Recv	Data packet

The *SETLINKQ* and *GETLINKQ* messages are used by the manager to keep track of workers' traffic. This information is used by the worker in the next node selection process. The manager manages the memory space used in the simulation via the *SETTIME* and the *GETTIME* messages. These messages indicate the recent event time history of each worker and are used to determine the global time. The routing and data packets in the simulation are forwarded in the network topology based on the information in the *FORWARDANT, BACKWARDANT* and *DATAPKT*.

The workers are a group of logical proceses, who conduct the actual simulation by periodically generating a set of discrete events. Each worker maintain a packet queue. In each event, a routing or a data packet is generated. These packets are transported by ants. The routing packets are used to measure the utilization of all communication links in the topology, and are carried by both forward and backward ants during the reinforcement learning period. The throughput and the average delay of data packets indicate a performance of the AntNet routing policy. The workers' task can be described in the following psudo-code fragment.

```
worker() {
1   {routing_table,  traffic_table}  =  Initial-
ize();
2   while (!terminate){
3       if (ant = receive (any_worker))
4           addarrive (ant);
5       send (manager, link_queue_status);
6       if (at_scheduled_time)
7           new_ant = create_ant (route_packet);
8       ant = getevent ();
9       if (ant == arrival)
10          arrive ();
        else {
11          depart ();
```

```
12          if (!destination_node) {
13              compute_trip_time (ant);
14              q = request_link_queue (manager);
15              next_node = select_neighbor (q);
16              if (!detect_cycle (ant));
17                  send (next_node, ant);
                else
18                  destroy (ant);
            }
            else {
19              backward_ant = ant
20              send (prev_node, backward_ant);
            }
        }
    }
}
```

Each worker obtains a partial routing table from a manager and initializes a simulation object (line 1). If an ant arrives from another worker, its timestamp will be checked with the function called, *Addarrive* (line 2 to 4). The worker then sends its link queue status to the manager (line 5), and generate a new ant according to the predefined scheduled time (line 6 to 7). The new ant carries the routing packet. The worker then selects the next event to be processed from both the arrival and the departure event lists (line 8).

If the selected event is equal to arrival, a new ant that carries the data packet is generated using a function called, *Arrive* (line 9 to 10). Otherwise, the selected event/ant is departed from the node. The actions are described in a function, called *Depart* (line 11). After a departure ant is de-queued, it is driven by the routing mechanism. The worker will check whether the current node is the ant's destination. If that is not the case, a trip time will be computed and the next node will be selected from its neighbors based on the link queue status sent by a manager (line 12 to 15). A link queue status indicates traffic at the neighboring nodes by evaluating a number of packets/ants queued at each node. The next node is the node with the least traffic.

Before an ant is sent to the next node, its memory will be searched for a circular path (line 16). The ant will be forwarded if and only if a cycle is not detected. It will be destroyed, otherwise (line 17 to 18). If the node is the ant's destination, the ant will reverse its traveling direction and becomes a backward ant (line 19 to 20). The information which is stored in a backward ant's memory is used in reinforcement process. Thus, a backward ant does not need to find the next node. It can just use the information in its memory to trace backward toward the source.

Addarrive: manipulates the arrival ants from another worker in the simulation. Addarrive uses a synchronous mechanism to handle an arrival ant with a timestamp smaller than some previously serviced messages. If the timestamp breaks the increasing order, the roll-back mechanism is employed. If needed, an *anti-message* (message cancellation) will be sent to the required destinations.

Arrive: generates a new arrival ant. The worker will check whether its status is *busy* or *idle*. If the status is busy, an arrival ant will be kept at the end of the worker's queue, where it awaits a service. Otherwise, the worker's status will be updated and the ant will be serviced.

Depart: generates a departure ant. The worker's queue is checked. In a case of an empty queue, the worker status is changed to idle and a new departure ant is generated. Otherwise, a current event at the front of the queue is used to generate a departure ant. The delay time, which is a partial trip time used in a reinforcement process, is also calculated in the process.

5 EXPERIMENTAL RESULTS

In our experiments, we use the NFSNET or the National Science Foundation topology. The NFSNET composes of 14 nodes and the communication bandwidth are 1.5 Mbps. The propagation delay ranges from 4 to 20 msec. The network topology is shown below.

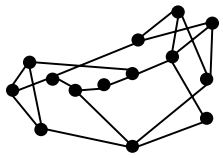

Figure 3: NSFNET topology

Other experimental data setting uses, the packets' size of 4096 bits. The forward ant is generated every 1 sec. The mean of packet inter-arrival time is 0.3 and the data packets are generated at the same rate for all nodes (uniform passion distribution).

The overall performance of the routing mechanism in a communication network is measured by the *throughput* and the *average packet delay*. The throughput measures the quantitative services that the network is able to offer in a certain amount of time, while the packet delay defines the quality of service produced.

Figure 4 and 5 show the network performance of the parallel simulation running on a heterogeneous network of PCs. From the result, notice that, the difference between 1-, 2-, 3-, and 4-node environment are minimal indicating that our parallel AntNet generate the correct results.

Figure 6 shows the parallel scalability analysis. From the plot, as more computers were added, the average processing time decreased rapidly. Applying parallel computing model, we are able to achieve the convergent point (same throughput and average delay) much faster. Our modified version of AntNet divided the simulation in both data and functional domains. It is fast and efficient and, thus is appropriate for use in the network routing simulation.

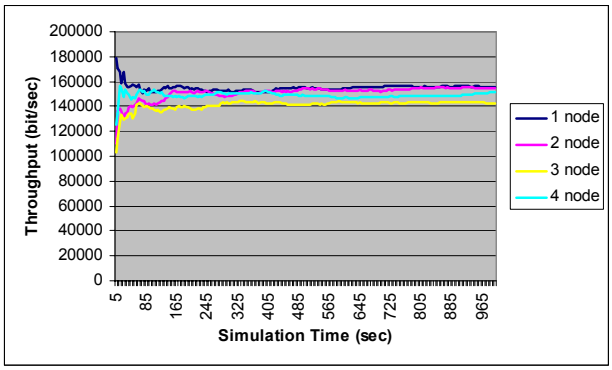

Figure 4: Throughput with MPIA 0.3

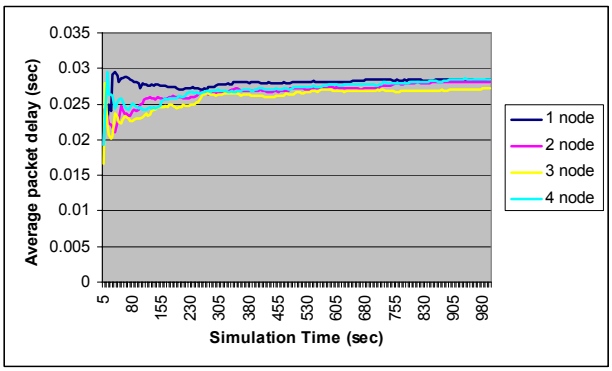

Figure 5: Average packet delay

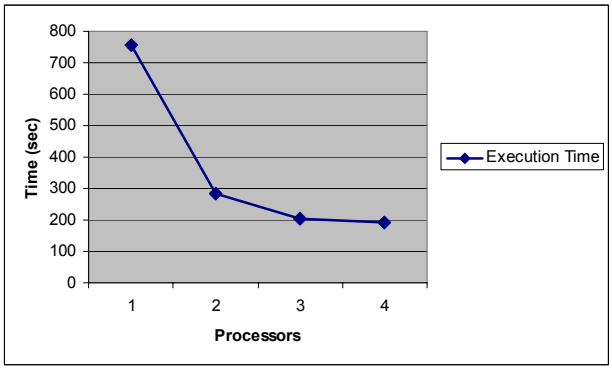

Figure 6: The Heterogeneous Scalability Plot

6 CONCLUSION

This research presents a parallel computing framework for the AntNet routing simulation. We redesigned AntNet communication and synchronization methodology based on PDES. To show the overall performance of the simulation, the network throughput and the average delay time are measured on a network of Commercial-off-the-shelf (COTS) workstations. Our experiments offer similar result to other AntNet system in terms of throughput and average delay. However, the parallel algorithm can cut down processing time greatly. From the results, the algorithm scale extremely well, and thus can be considered for use with real-time network simulation in the future.

7 REFERENCES

Antony, D., Piriyakumar, Louis and Levi, P. 2002. A new approach to exploiting parallelism in ant colony optimization, *International Symposium on Micromechatronics and Human Science*, IEEE: 237-243.

Bolondi, M. and Bondanza, M. 1993. Parallelizzazione di un algoritmo per larisoluzione del problema del commesso viaggiatore. Master's thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy.

Bullnheimer, B., Kotsis, G. and Strauss, C. 1998. Parallelization strategies for the Ant System. In R. De Leone, A. Murli, P. Pardalos, and G. Toraldo, editors, *High Performance Algorithms and Software in Nonlinear Optimization*, volume 24 of Applied Optimization, Kluwer Academic publishers, Dordrecht, NL: 87-100.

Chandy, K.M. and Misra, J. 1979. Distributed simulation: A case study in design and verification of distributed programs, *IEEE Trans. Software Eng.*, Vol. SE-5, no. 5, pp. 440-452.

Di Caro, G. and Dorigo, M. 1998. AntNet: Distributed Stigmergetic Control for Communications Networks, *Tech.Rep. IRIDIA/98-01*, Université Libre de Bruxelles, Belgium.

Dorigo, M. and Di Caro, M. 1999. The Ant Colony Optimization meta-heuristic In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization.* McGraw Hill, London, UK: 11-32..

Dorigo, M., Maniezzo, V. and Colorni, A. 1996. The Ant System: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics – Part B*: 26(1):29-41.

Jain, Anil K. and Dubes, Richard C. 1948. *Algorithms for Clustering Data*, Prentice Hall: 58-73

Jefferson, D. 1985. Virtual time, *ACM Trans. Prog. Lang. Sys.*, vol. 7, No. 3.

Halabi, B. 1997. Interdomain Routing Basics, *Internet routing architectures.* Cisco Press: 89-96.

Schoonderwoerd, R., Holland, O., Bruten, J. and Rothkrantz, L. 1996. Ant-based load balancing in telecommunications networks, Adaptive Behavior: 5(2): 169-207.

Stützle, T. 1998. Parallelization strategies for ant colony optimization, In Agostion E. Eiben, Thomas Bäck, Marc Schoenauer, and Hans-Paul Schwefel, editors, *Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature*, volume 1498 of Lecture Notes in Computer Science, Springer Verlag, Berlin, Germany: 722-731.

AUTHOR BIOGRAPHIES

KITTIPONG THEERAWATSATHEIN is a master student of Department of Computer Engineering, King Mongkut's University of Technology Thonburi, Thailand. He received a B.S. degree in computer science from Silpakorn University, Thailand, in 2000. He has worked for Office of Information Technology, Chulalongkorn University, as a system analyst. His research interests include parallel and distributed computing and network simulation. His email address is <tymsum@hotmail.com>

TIRANEE ACHALAKUL received her B.S. degree in computer engineering from King Mongkut's Institute of Technology Ladkrabang, Thailand, in 1994 and completed her PhD in computer engineering from Syracuse University, New York, in 2000. After finishing her Ph.D., she joined the Johnson and Johnson Vision Care, Inc., as a system developer in Roanoke, Virginia. She is currently holding a faculty position at the Department of Computer Engineering, King Mongkut's University of Technology Thonburi, Thailand. Her research interests include parallel and distributed computing, image processing and fusion, and collaborative learning and their applications. Her email address is <tiranee@cpe.kmutt.ac.th>