Recent Progress in Ant Algorithms for Fixed Telecommunication Networks: A Review

Malgorzata Gadomska, Andrzej Pacut

Institute of Control and Computation Engineering
Warsaw University of Technology
00-665 Warsaw, Poland

Abstract. The aim of this paper is to provide a comprehensive review of Swarm Intelligence based methods for fixed packet-switched telecommunication networks. Such methods, thank to their distributed form and self-organizing dynamics, are very well applicable to problems encountered in a network environment. We believe to show the motivation for using learning methods to solve the routing problem and moreover to create a basis for researchers who act in the field of Swarm Intelligence.

1 Introduction

In the recent years, a rapid growth of telecommunication networks can be observed. The increasing size, changing conditions and complexity of the today's networks implies a need for decentralized and adaptive control mechanisms. Especially, there is a strong need for routing algorithms, as they play a critical role in the network performance. The load of telecommunication networks is neither constant in time (homogeneous) nor uniformly distributed over the whole network (uniform). It is then very important to develop effective routing algorithms which are able to distribute data traffic across multiple paths and quickly adapt to changes in topology and changes in the data traffic distribution.

During the past years, many such adaptive routing algorithms have been proposed. According to the learning techniques used, three major research directions can be distinguished: *Swarm Intelligence*, *Reinforcement Learning* and *Evolutionary Computing*. Swarm Intelligence utilizes the principles of certain self organizing processes observed in Nature to propose solutions to different optimization problems. In Reinforcement Learning techniques, an agent learns on the basis of reinforcements achieved from interactions with an unknown environment. Evolutionary Computing takes the evolution process as a basis for developing algorithms. We concentrate on the Swarm Intelligence approach, mainly on the *ant routing* algorithms. Swarm Intelligence is an evolving field yielding methods for distributed systems optimization. For the purpose of network routing, the ants paradigm has already been proved to be very promising. *Ant algorithms* do not require supervision, and their distributed form makes them well applicable to the routing problem.

The aim of this paper is to survey the most interesting new concepts in Swarm Intelligence based methods for telecommunication networks. We believe to show the motivation

for using learning methods to solve the routing problem and moreover to create a basis for researchers who act in the field of Swarm Intelligence. We will restrict our survey of adaptive routing algorithms to fixed packet-switched networks.

The paper is organized as follows. In Sec. 2 we shortly present the Swarm Intelligence paradigm. We also introduce the Ant Colony Optimization (ACO) metaheuristic and the AntNet algorithm [7, 5] which is the first ant routing algorithm concerning packet-switched networks. Moreover, a classification scheme of the AntNet algorithm's modification is proposed. In the following sections we present the most recent modifications and extensions proposed to AntNet and analyze their impact on the routing efficiency: modifications concerning the learning model in Sec. 3, approaches which improve the algorithms convergence in Sec. 4, different exploration techniques in Sec. 5. A novel Swarm Intelligence approach, namely, the BeeHive algorithm, is introduced in Sec. 6. Section 7 concludes the paper.

2 Swarm Intelligence and Ant Colony Optimization

Swarm Intelligence is related to the emergent collective intelligence of groups of simple autonomous agents. From the Artificial Intelligence perspective, Swarm Intelligence is a self-organization based computational and behavioral paradigm for solving distributed problems. It provides a basis enabling to explore collective (or distributed) problem solving without any centralized control or a provision of a global model. Swarm Intelligence systems are typically made up of a population of simple agents interacting locally with one another and with their environment.

Ant Colony Optimization is a metaheuristic based on the Swarm Intelligence paradigm. It was inspired by observations of real ant colonies and first proposed by Dorigo, Maniezzo, and Colorni in 1991 [6] as a multi-agent approach to the traveling salesman problem (TSP) and the quadratic assignment problem (QAP). Although, by itself, an ant is a simple and unsophisticated insect, collectively, a colony of ants can perform complex tasks such as discovering routes to food sources and building nests. While traveling, ants lay pheromone on the ground. As a result, the concentration of pheromone on the shortest path is reinforced at a higher rate than on the other paths. Ants prefer to travel along paths with higher pheromone concentration, which results in a majority of ants using the shortest path for foraging in a steady state. The communication between ants is indirect through the environment, what is called a stigmergy.

The first ant routing algorithm was proposed by Schoonderwoerd in 1996 [9]. Their ABC algorithm could be applied only to symmetric circuit-switched networks. However, AntNet algorithm introduced by Dorigo and di Caro in 1998 [7] was the first ant-routing algorithm designed for asymmetric packet-switched networks, and the primary objective of the algorithm was to maximize the performance of a complete network. The algorithm implicitly achieves load balancing by probabilistic distribution of packets on multiple paths. The experiments reported in [7, 4, 5] have proved that AntNet outperforms, with respect to the throughput and the delay, other competitors, such as Q-routing [2], PQ-routing [3], Shortest Path First (SPF) and OSPF. In [12] it has been shown that AntNet performs very well also under high and/or time varying load levels (also those changing periodically).

Various modifications of AntNet have been developed within the following years, therefore we will treat this algorithm as a basis, and introduce its modifications and

extensions according to the following classification:

- **Model.** Here we consider modifications of the routing probabilities and the traffic model [18, 4].
- **Convergence.** All techniques that improve the convergence of the ant algorithm are considered here, including an intelligent initialization of the routing tables [1, 14], limitation of the number of ants in the network [1, 14] and introduction of new ant agents [13].
- **Exploration.** The ants must explore the network to find efficient paths. Even when the paths have already been discovered, exploration is still necessary to prevent stagnation and to ensure algorithm's adaptability to changing network conditions [18, 1, 14, 11].
- **Scalability.** Here we consider the algorithms in which a hierarchy is introduced in order to improve a scalability [15].

In the following sections we will present the most interesting modifications in the above categories. Those belonging to more then one category will be placed to the most relevant one.

The presented algorithms are evaluated using various quality indicators. The most common are the mean packet delay and the convergence time, understood as the time necessary for the average packet delay to stabilize. Moreover, the generated routing overhead, which denotes the extra load generated by routing packets, and the ability to avoid congestion are taken into account.

2.1 AntNet

In this section we will describe the basic AntNet algorithm [7]. There are two types of simple agents (ants): the *forward ants* that explore the network in order to find paths, and the *backward ants* that use information collected by the forward ants to improve the routing policies. Every network node k is assigned a *routing table* T_k and a statistical *traffic model* M_k including some local traffic statistics. The *routing table* T_k stores the probabilities $t_k(d, n)$ for each neighbor node n and each destination node d used to determine the probabilistic routing policy. Both the routing tables and the statistical model are calculated iteratively during normal operation of the network.

The *forward ants* $F_{s \to d}$ are launched at regular intervals from randomly selected *source nodes* s to randomly selected destination nodes d. For each node visited k, the forward ant stores its age, i.e. the time elapsed from its launch, and chooses the next node to be visited n according to the probability $p_k(d, n)$. This probability is calculated by modifying the probability $t_k(d, n)$ stored in the *routing table* T_k by a relative length $q_k(n)$ of the output queue in node k, namely

$$p_k(d, n) = \frac{t_k(d, n) + (1 - \ell_k(n))}{1 + (|N_k| - 1)}, \quad n \in N_k \tag{1}$$

where N_k denote the set of neighbors of the node k, $|N|$ denotes the number of elements of a set N and $l_k(n)$ is the relative output queue length, namely

$$\ell_k(n) = \frac{q_k(n)}{\sum_{n' \in N_k} q_k(n')} \tag{2}$$

where $q_k(n)$ denote the output queue lengths. If a cycle is detected, namely, the ant visits a node repeatedly, the nodes in the cycle are removed from the ant's memory, and if the cycle length is greater than half the ant's age, the ant is destroyed.

After reaching the destination node, the forward ant $F_{s\to d}$ creates the *backward ant* $B_{d\to s}$, transfers all the collected knowledge about the visited nodes and the visiting times, to the backward ant and is removed from the system (dies). The backward ant travels back the same path as its parental forward ant, but uses high-priority queues in order to quickly propagate information about the discovered path. In every visited node k it updates the values of the traffic model and the routing probabilities for all the entries corresponding to every node i on its path.

The *traffic model* M_k at node k consists of the estimates $\mu_k(d)$ and $\sigma_k^2(d)$ of the expected value and the variance of the trip time from k to the destination node d, and are updated according to

$$\begin{aligned} \mu_k(d) &\leftarrow \mu_k(d) + \eta\,(o_k(d) - \mu_k(d)) \\ \sigma_k^2(d) &\leftarrow \sigma_k^2(d) + \eta\Big(\big(o_k(d) - \mu_k(d)\big)^2 - \sigma_k^2(d)\Big) \end{aligned} \tag{3}$$

where $o_k(d)$ is the trip time from k to the destination node d as observed by the forward ant $F_{s\to d}$, and η is a parameter. Note that if $\eta = 1/m$ in (3), where m is the update number, we obtain the usual average value.

In the routing table T_k, the probability corresponding to the neighbor f chosen at node k by the forward ant $F_{s\to d}$ is increased

$$t_k(d,f) \leftarrow t_k(d,f) + r\,(1 - t_k(d,f)) \tag{4}$$

with the corresponding decrease of the remaining neighbor probabilities

$$t_k(d,n) \leftarrow t_k(d,n) - r\,t_k(d,n), \quad n \neq f, n \in N_k \tag{5}$$

to make their sum equal to one. The *reinforcement parameter* r depends on the ant trip time and on the local *traffic model* M_k, namely

$$r = c_1 \frac{W_k(d)}{o_k(d)} + c_2 \frac{I_k^{hi}(d) - I_k^{lo}(d)}{\big(I_k^{hi}(d) - I_k^{lo}(d)\big) + \big(o_k(d) - I_k^{lo}(d)\big)} \tag{6}$$

where $o_k(d)$ is the observed trip time, constants c_1 and c_2 control the effect of the last $o_k(d)$, and $W_k(d)$ is the ant's shortest trip time for the given destination at the last observation period. $[I_k^{lo}(d), I_k^{hi}(d)]$ denotes the confidence interval of the mean trip time. All the details and justifications can be found in [7].

3 Modifications of the learning model

In this section we present modifications of the model used in AntNet proposed in order to achieve a better performance.

3.1 AntNet-FA

The authors of AntNet proposed a variant of AntNet, known as AntNet-CO or AntNet-FA [4]. Its basic behavior is identical to AntNet except that in AntNet-FA both

forward and backward ants make use of high-priority queues. Moreover, the forward ants do not carry any information about their experienced trip times. The backward ants update the routing tables in the visited nodes using estimates of forward ants trip times computed at each node k on the base of the depletion dynamics of local links l and the actual queue sizes. On the base of this estimates, a statistical *traffic model* M_k is built for each node k. The estimated trip time $o_k(n)$ from any node k to a neighbor node n is calculated as:

$$o_k(n) = d_k(n) + (\ell_k(n) + s_a)/B_k(n) \tag{7}$$

where $d_k(n)$ is the links propagation delay, $\ell_k(n)$ is the size of the output queue to neighbor n (in bytes), s_a is the ant's size and $B_d(n)$ is the link bandwidth.

Through this modification, forward ants quickly discover paths that are later evaluated by backward ants using locally estimated trip times. As reported in [4], AntNet-FA showed always the best performance, as compared both to traditional algorithms (OSPF, SPF) and learning based algorithms (Q-Routing, PQ-Routing, basic AntNet). The advantages of AntNet-FA increased with the size of the test networks. In a 150-node topology, the delays obtained when using AntNet-FA where about four times lower considering the 90th percentile, then in the case of basic AntNet.

We believe that the most significant advantages of AntNet-FA are that: (i) the forward ants use priority queues which accelerates the propagation of information about good paths in the network, (ii) the estimated trip times are gathered later and are more up-to-date then in the case of AntNet, (iii) the statistical traffic model is also more reliable.

3.2 Ant Swarm-based Routing

The Adaptive Swarm-based Routing (ASR) was proposed in 2004 for packet-switched networks by Yong, Guang-Zhou and Fan-Jun [18]. Only the main differences from AntNet will be described, as the basic concept is very similar.

A different *traffic model* M_k is introduced. For every destination node d and neighbor node n, the last two estimates, $D_k(d,n)$ and $D_k^-(d,n)$, of the trip time from k via n to the destination node d are stored. The backward ant $B_{d\to s}$ updates the values of the model

$$D_k(d,n) \leftarrow D_k(d,n) + \eta\left((1-\alpha)\Delta D_k(d,n) + \alpha\Delta D_k^-(d,n)\right) \tag{8}$$

where

$$\begin{aligned} \Delta D_k(d,n) &= d_k(d,n) - D_k(d,n) \\ \Delta D_k^-(d,n) &= D_k(d,n) - D_k^-(d,n) \end{aligned} \tag{9}$$

$d_k(d,n)$ is the forward ants trip time from node k to node d via node n, η is the learning rate and α is the momentum parameter.

The second significant difference is the way of updating the routing probabilities stored in the routing table T_k at every node k. After updating the network's traffic model M_k, the routing probabilities are computed as

$$t_k(d,n) = \frac{(D_k(d,n))^{-\beta}}{\sum_{n'\in N_k}(D_k(d,n'))^{-\beta}} \tag{10}$$

where $\beta \geq 1$ is a *non-linearity parameter*. Note that this method corresponds to Boltzmann exploration with the quality equivalent to $\ln D_k(d,n)$, since we can write

$$t_k(d,n) = \frac{e^{-\beta \ln D_k(d,n)}}{\sum_{n' \in N_k} e^{-\beta \ln D_k(d,n')}} \tag{11}$$

Finally, the ant agent realizes the ϵ-*greedy* policy, namely, it chooses a random neighbor node with a probability ϵ and with the probability $1-\epsilon$ it selects the node according to the routing probabilities $t_k(d,n)$. This approach prevents the probabilities of the shortest paths from becoming very close to 1, which would reduce the exploration, i.e. the capability of the algorithm to adapt to the environmental changes.

In order to speed up the learning process, the routing probabilities are initialized in such a way that a higher initial probability is given to the neighbor which happens to be the destination node.

In [18, 12, 8] it was shown that the ASR finds efficient routing policies much faster then AntNet and under high loads it assures lower mean packet delays. Moreover it assures faster adaptation to various load level changes, what has been investigated in [12]. Experiments reported in [8] also proved that contrary to the common belief, also TCP in the transport layer still enables the adaptive algorithms to extend the range of load levels under which they can find efficient routing policies. Under TCP, the ASR outperformed AntNet in terms of both the mean packet delay and the convergence range.

4 Improving the convergence

In this section we will describe the methods proposed to improve the convergence of AntNet. One of the ideas is to introduce intelligent initialization of the routing tables, as it has been done in ASR (Sec. 3.2). The same initialization method is also used in [1].

In [1], it is also suggested to constrain the number of ants inside the network, to remain lower than four times the number of links. This restriction may reduce the routing overhead and the possible congestion, but it also places a restriction on the frequency of launching ants which may lead to a reduction in the adaptiveness of the routing algorithm. A different scheme of controlling the number of ants in the network is used in [14, 12, 8] where the ratio of the number of forward ants and data packets is controlled.

In order to reduce the overhead and out-of-date information it is proposed in [14] to delete from the network the ants older then 2 times the number of nodes. This modification reduced the mean packet delay, because in the original AntNet some packets travel with a very high number of hops.

4.1 Helping ants

In [13] the authors introduced the *helping ants* into AntNet in order to increase the cooperation among neighboring nodes, thereby reducing the algorithm's convergence time. The concept of helping ants is inspired by the fact that insect's coordination via the pheromones relies on three main dynamic processes [14]: aggregation of successive deposits by different ants, evaporation of pheromone to discard obsolete information, and propagation of pheromone from one location to other nearby locations to share

information. Ant Colony Optimization uses aggregation and evaporation, but does not perform pheromone propagation.

The helping ants imitate the mechanism of information sharing among nearby ants by pheromone propagation in their natural colonies. The modified AntNet has three kinds of agents: forward, backward and helping ants. Similar to AntNet, each forward ant $F_{s\to d}$ explores the network and collects information while moving from a source mode s toward a destination d. After arriving at its destination, the forward ant generates a backward ant $B_{d\to s}$ and is removed from the system. The backward ant travels back along the same path as the forward ant and updates the routing tables and statistical traffic models on its way. The statistics computed in the traffic model and the updating methods are the same as in the original AntNet.

When the backward ant $B_{d\to s}$ reaches its destination node s and the trip time $o_s(d)$ is good, i.e. $o_s(d) < \mu_s(d)$, s creates helping ants and sends them to its neighbors. When a helping ant reaches the corresponding neighboring node i, it computes $o_i(s)$, as $o_i(s) = o_s(i)$. Then, under the assumption that the queuing time is negligible, the new trip time $o_i(d)$ through the neighbor s is estimated as: $o_i(d) = o_i(s) + o_s(d)$.

If the trip time is good, i.e. $o_i(d) < \mu_i(d)$, the helping ant updates the routing probabilities in the node i, according to the same scheme as in the original AntNet (equations 4, 5, 6).

According to [13], the proposed modification significantly reduces the mean delay and network overhead without an increase of the loss rate or jitter. Simulations using the NSFNet, which is the old USA backbone (1987), show that helping ants, in comparison to the original AntNet, decreased the mean packet delay at a significant factor. Moreover, reduction of the forward ant generation rate by 50% in the modified Antnet, makes the behavior of both algorithms approximately equal, while the overhead in modified AntNet is much lower. Experiments also indicate that the performance of the modified AntNet significantly improves when the paths are long or there exists few alternative paths. The algorithm's behavior was also tested under changing network conditions, such as a link going down and coming up again. In such situations AntNet with helping ants reduces the packet delay by reducing the convergence time.

In our opinion, this modification is worth noticing, however, we believe that the estimated trip time $o_i(s)$ should not be based on the real trip time of the helping ant traveling from s to d, because the ant travels this path in the opposite way. The estimate of the trip time $o_i(s)$ should rather be calculated on the base of the known link delay from d to s and the present size of the output queue.

5 Introducing alternative exploration techniques

If many ants would choose a non-optimal path at the beginning of the learning process, other ants would follow these paths and reinforce them. If no exploration techniques were applied, the ants would stagnate travelling along a non-optimal paths. Even if they did find an optimal path, lack of exploration would prevent them from adapting to changing network conditions. It is therefore very important to use exploration techniques. Here we present the main exploration techniques used in ant algorithms, namely, the pheromone control strategies. A more detailed description can be found in [17].

In the Ant Colony Optimization algorithms exploration is assured by pheromone evaporation [6]. The value of pheromone μ_{ij} on all links (i, j) is decreased by a factor

$\alpha < 1$, namely $\mu_{ij} \leftarrow \alpha\mu_{ij}$. This reduces an influence of the past experience. If a congestion occurs on some frequently used path and few ants will succeed returning from that path, the probability of using this path will decrease and the other paths will be discovered. The second method to mitigate stagnation is *ant aging*, motivated by a decrease with ant's age.

In AntNet [7, 4], the reinforcement (the amount of pheromone) of a path depends on its quality evaluated on the base of the traffic model, the trip time of the current forward ant and the best trip time experienced by any ant within the specified time window (6). According to [7, 4], such policy results in a better performance and convergence.

The authors of ASR [18] use a method corresponding to Boltzmann's exploration, with the quality of a path inversely proportional to the logarithm of the estimated trip time. Additionally, with a probability ϵ the ant chooses a random neighbor and with the probability $1 - \epsilon$ chooses a node according to the routing probabilities. This assures the capability of the algorithm to adapt to the environmental changes (Sec. 3.2). A similar technique is also proposed in [1].

An original approach to use the simulated annealing principles to assure exploration in an ant algorithm was introduced in [11] and will be described closely in the following section.

5.1 Annealed ant algorithm

In [11], the authors propose an annealed ant system approach to network routing. The original ant algorithm is modified by adding an annealed strategy with a cooling schedule to calculate the routing probabilities, which are then used to choose the ant's next hop by using Roulette wheel selection. The major strength of the proposed annealed ant approach is that the routing probabilities follow the annealed strategy with a cooling schedule.

Simulated annealing is a stochastic relaxation strategy used successfully to resolve complex optimization problems. This technique allows the search to escape a local minimum, because there is a nonzero probability to move temporarily toward a worse state in order to move away from local extreme. In order to converge to a near global minimum in the annealing process, a feasible *cooling schedule* is required. In [11] the following scheme is used, previously proposed in [10]:

$$T(t) = \frac{1}{\mu + 1} \left(\mu + \tanh(w^t)\right) T(t-1) \tag{12}$$

where w is a constant close to 1 and μ is also a constant. It is shown in [10] that the proposed scheme results in a faster decrement speed than an exponential cooling scheme, $T(t) = \alpha T(t-1)$, $\alpha < 1$.

The routing probabilities of choosing neighbor node n when being at node k and traveling to destination d, are calculated according to the Boltzmann probabilities, namely

$$t_k(d,n) = \frac{e^{-E_k(d,n)/T}}{\sum_{n' \in N_k} e^{-E_k(d,n)/T}} \tag{13}$$

where

$$E_k(d,n) = -\tau_k(d,n)^\alpha \mu_k(d,n)^\beta \tag{14}$$

$\tau_k(d,n)$ is the pheromone intensity on path (k,n), which is inversely proportional to the ants trip time from k to d via n, $\mu_k(d,n)$ is the *visibility* of node n from k and is inversely proportional to the Euclidean length of the path (k,n), α and β are constants.

In numerical experiments there is no other traffic in the network, apart from ants, therefore the situation is static. Two learning models are introduced: the *Concentrated Model* in which all ants are initially concentrated in the source node and their task is to find an efficient path to the destination node, and the *Distributed Model*, where all ants randomly select the start nodes with at least one ant in the source node.

The authors show that the Concentrated Model assures better performance than the Distributed Model and is able to find the efficient path faster. Moreover, the proposed annealed system in the Concentrated Model found the optimal solution of the shortest-path problem in all experiments.

The authors haven't tested their approach in real network conditions, however we believe that adding a cooling strategy to ant routing algorithms may be useful. Such strategy would enable the algorithm to test more solutions at the initial stage of the learning process, and therefore a near-optimal solution could be found. The cooling process should be stopped at a degree assuring some exploration, even when the learning process is completed, in order to preserve the algorithm's ability to adapt to environmental changes. Moreover, introducing the annealing strategy in the initial faze of learning would cause the ant routing algorithms to be more independent of initial conditions.

6 A novel Swarm Intelligence approach: the BeeHive algorithm

The *BeeHive* algorithm, proposed in [15], is a novel approach to adaptive routing, based on the foraging principles of a honey bee colony. Contrary to the ant routing approach, in BeeHive a hierarchy is introduced to the routing scheme. The algorithm consists of two types of agents: *short distance* and *long distance agents*. The main task of both is to explore the network and to evaluate the quality of the discovered paths. However, the short distance agents have constrained travel range, namely they are allowed to make only up to 7 hops from their source node. The long distance agents collect and propagate information in the entire network. Such hierarchical model helps to minimize the routing overhead and causes the algorithm to be better scalable.

The BeeHive network is subdivided into foraging zones, defined as the sets of nodes around a given node, from which short distance agents can reach the given node. The foraging zones may overlap. Moreover, the network is viewed as a cluster of non-overlapping foraging regions in which a node belongs to just one region only. Details of the foraging regions formation process can be found in [15]. Each foraging region has a representative node and its role is to launch long distance bee agents. Each node maintains the routing information for all nodes within its foraging zone, and for representative nodes of the foraging regions. If the destination of a packet does not lie within the foraging zone of a node, then it is forwarded along a path leading to the representative node of the foraging region containing the destination node.

During the start-up phase, the network is divided into foraging regions by the short distance agents. During the normal phase, each non-representative node periodically sends short distance agents to explore the network and the representative nodes launch long distance agents. Each agent collects and carries path information while traveling. In each visited node, the agent leaves the estimate of the time to reach its source node from

this node over the incoming link. All bee agents use priority queues in order to quickly propagate information about efficient routes in the network. Each node must maintain three routing tables: for reaching the nodes within its foraging zone, for reaching the representative nodes of foraging regions, and the routing table providing mappings of all possible destinations to foraging regions.

The next hop for a data packet is selected according to a stochastic policy:

$$t_k(d,f) = \frac{\frac{1}{o_k(d,f)+\ell_k(d,f)}}{\sum_{n\in N_k}\left(\frac{1}{(o_k(d,n)+\ell_k(d,n)}\right)} \tag{15}$$

where $o_k(d,f)$ and $\ell_k(d,f)$ are the propagation and queuing delays, respectively, estimated by the bee agents, for reaching destination d via neighbor f of node k and N_k is the set of node's k neighbors.

The BeeHive algorithm is able to deliver the same or better performance than AntNet, but with significantly smaller routing tables, processing complexity and control overhead. We believe that the BeeHive algorithm may help to increase the scalability of adaptive routing algorithms, without loosing their good performance.

7 Conclusions

We made a comprehensive survey of ant algorithms for fixed telecommunication networks. The AntNet algorithm has been chosen as a base ant routing algorithm for packet-switched networks and the most interesting and promising approaches were described in the context of AntNet's modifications. We proposed a classification scheme of AntNet modifications and extensions, and the impact of each modification on the routing efficiency was reported. We also describe a novel approach to network routing, namely the BeeHive algorithm.

A variety of algorithms described in this review prove that Swarm Intelligence methods can be successfully applied to solve the routing problem in telecommunication networks.

Acknowledgments

This work has been supported by Warsaw University of Technology Research Program grant.

Bibliography

[1] B. Baran, R. Sosa, "A new approach for AntNet routing", *Ninth International Conference on Computer Communications and Networks*, Las Vegas, NV, USA, pp. 303-308, 2000.

[2] J. A. Boyan, M. L. Littman, "Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach", *Advances in Neural Information Processing Systems*, volume 6, pp. 671-678, Morgan Kaufmann Publishers, Inc., 1994.

[3] P. Choi, D. Yeung, "Predictive q-routing: a memory-based reinforcement learning approach to adaptive traffic control", *Advances in Neural Information Processing Systems 8*, pp. 945-951, 1996.

[4] G. Di Caro, M. Dorigo, "Two ant colony algorithms for best-effort routing in datagram networks", *Proceedings of the Tenth IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS98)*, IASTED/ACTA Press, Anheim, pp. 541-546, 1998.

[5] G. Di Caro, M. Dorigo, "Ant Colonies for Adaptive Routing in Packet-Switched Communications Networks", *Proceedings of PPSN V - Fifth International Conference on Parallel Problem Solving from Nature*, Amsterdam, Holland, LCNS 1498, Springer-Verlag, pp. 671-683, 1998.

[6] M. Dorigo, V. Maniezzo, A. Colorni, "Positive feedback as a search strategy", *Technical Report 91-016*, Politecnico di Milano, Dipartimento di Elettronica, 1991.

[7] M. Dorigo, G. Di Caro, "AntNet: Distributed Stigmergetic Control for Communications Networks", *Journal of Artificial Intelligence Research 9*, pp. 317-365, 1998.

[8] M. Gadomska, A. Pacut, "Performance of Ant Routing Algorithms When Using TCP", *Applications of Evolutionary Computing*, LNCS 4448, Springer-Verlag, pp. 1-10, 2007.

[9] R. Schoonderwoerd, O. Holland, J. Bruten, L. Rothkrantz, "Ant-based load balancing in telecommunications networks", *Adaptive Behavior, 5(2)*, pp. 169-207, 1996.

[10] Jzau-Sheng Lin, "Image vector quantization using an annealed Hopfield neural network", *Optical Engineering*, Vol. 38, pp. 599-604, 1999.

[11] Shao-Han Liu, Jzau-Sheng Lin, Zi-Sheng Lin, "A shortest-path network problem using an annealed ant system algorithm", *Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science (ICIS05)*, Washington, DC, USA, pp. 245-250, 2005.

[12] A. Pacut, M. Gadomska, A. Igielski, "Ant-Routing vs. Q-Routing in Telecommunication Networks", *Proceedings of the 20-th ECMS Conference*, Bonn, Germany, pp. 67-72, 2006.

[13] Azadeh Soltani, M.-R. Akbarzadeh-T, M. Naghibzadeh, "Helping ants for adaptive network routing", *Journal of the Franklin Institute*, pp. 389-403, 2006.

[14] F. Tekiner, F. Ghassemlooya, S. Al-khayattb, "The Antnet Routing Algorithm - Improved Version", *Proceedings of the Communication Systems, Networks and Digital Signal Processing Conference*, Newcastle, UK, pp. 416-419, 2004.

[15] H. F. Wedde, M. Farooq, Y. Zhang, "BeeHive: An effcient fault-tolerant routing algorithm inspired by honey bee behavior", *In Ant Colony Optimization and Swarm Intelligence*, LNCS 3172, Springer-Verlag, pp. 83-94, 2004.

[16] H. F. Wedde, M. Farooq, "A performance evaluation framework for nature inspired routing algorithms", *Applications of Evolutionary Computing*, LNCS 3449, Springer-Verlag, pp. 136-146, 2005.

[17] H. F. Wedde, M. Farooq, "A comprehensive review of nature inspired routing algorithms for fixed telecommunication networks", *Journal of Systems Architecture*, pp. 461-484, 2006.

[18] Lu Yong, Zhao Guang-zhou, Su Fan-jun, "Adaptive swarm-based routing in communication networks", *Journal of Zhejiang Univ. SCIENCE, 5(7)*, pp. 867-872, 2004.