Ant colonies for Adaptive Routing in Packet-switched Communications Networks

Gianni Di Caro and Marco Dorigo

IRIDIA – Université Libre de Bruxelles – Belgium
{gdicaro, mdorigo}@ulb.ac.be

Abstract. In this paper we present AntNet, a novel adaptive approach to routing tables learning in packet-switched communications networks. AntNet is inspired by the stigmergy model of communication observed in ant colonies. We present compelling evidence that AntNet, when measuring performance by standard measures such as network throughput and average packet delay, outperforms the current Internet routing algorithm (OSPF), some old Internet routing algorithms (SPF and distributed adaptive Bellman-Ford), and recently proposed forms of asynchronous online Bellman-Ford (Q-routing and Predictive Q-routing).

1. Introduction

In this paper we consider the problem of adaptive routing in communications networks: we focus on routing for wide area datagram networks with irregular topology, the most remarkable example of such networks being the Internet.

The routing algorithm that we propose in this paper was inspired by previous works on ant colonies and, more generally, by the notion of *stigmergy* [15], that is, the indirect communication taking place among individuals through modifications induced in their environment. Real ants have been shown to be able to find shortest paths using as only information the pheromone trail deposited by other ants [2]. Algorithms that take inspiration from ants' behavior in finding shortest paths have recently been successfully applied to discrete optimization [4, 7, 11, 12, 13, 14, 18].

In ant colony optimization a set of artificial ants collectively solve a combinatorial problem by a cooperative effort. This effort is mediated by indirect communication of information on the problem structure they collect while building solutions. Similarly, in AntNet, the algorithm we propose in this paper, artificial ants collectively solve the routing problem by a cooperative effort in which stigmergy plays a prominent role. Ants adaptively build routing tables and local models of the network status using indirect and non-coordinated communication of information they concurrently collect while exploring the network.

We report on the behavior of AntNet as compared to the following routing algorithms: Open Shortest Path First (OSPF) [17], Shortest Path First (SPF) [16], distributed adaptive Bellman-Ford (BF) [20], Q-routing [5], and PQ-routing [6]. We considered a variety of realistic experimental conditions. In all cases AntNet showed the best performance and the most stable behavior, while among the competitors there was no clear winner.

This work was supported by a Madame Curie Fellowship awarded to Gianni Di Caro (CEC TMR Contract N. ERBFMBICT-961153). Marco Dorigo is a Research Associate with the FNRS.

A.E. Eiben et al. (Eds.): PPSN V, LNCS 1498, pp. 673-682, 1998.

2. Problem Characteristics

The goal of every routing algorithm is to direct traffic from sources to destinations maximizing network performance. The performance measures that usually are taken into account are *throughput* (correctly delivered bits per time unit) and *average packet delay*. The former measures the quantity of service that the network has been able to offer in a certain amount of time, while the latter defines the quality of service produced at the same time.

The general problem of determining an optimal routing algorithm can be stated as a multi-objective optimization problem in a non-stationary stochastic environment. Information propagation delays, and the difficulty to completely characterize the network dynamics under arbitrary traffic patterns, make the general routing problem intrinsically distributed. Routing decisions can only be made on the basis of local and approximate information about the current and the future network states. Additional constraints are posed by the network switching and transmission technology.

3. The Communication Network Model

In the following, a brief description of the features of the considered communication network model is given. In this paper we focus on irregular topology datagram networks with an IP-like (Internet Protocol) network layer and a very simple transport layer. We developed a complete network simulator (in C++) where the instance of the communication network is mapped on a directed weighted graph with N nodes. All the links are viewed as bit pipes characterized by a bandwidth (bits/sec) and a transmission delay (sec), and are accessed following a statistical multiplexing scheme. For this purpose, every routing node holds a buffer space where the incoming and the outgoing packets are stored. All the traveling packets are subdivided in two classes: data and routing packets. All the packets in the same class have the same priority, so they are queued and served only on the basis of a first-in-first-out policy, but routing packets have a higher priority than data packets.

Data packets are fed into the network by applications (i.e., processes sending data packets from origin nodes to destination nodes) whose arrival rate is dictated by a selected probabilistic model. The number of packets to send, their sizes and the intervals between them are assigned according to some defined stochastic process.

At each node, packets are forwarded towards their destination nodes by the local routing component. Decisions about which outgoing link has to be used are made by using the information stored in the node routing table. When link resources are available, they are reserved and the transfer is set up. The time it takes to a packet to move from one node to a neighboring one depends on its size and on the link transmission characteristics. If on a packet's arrival there is not enough buffer space to hold it, the packet is discarded. Packets are also discarded because of expired time to live. No arrival acknowledgment or error notification packets are generated back to the source.

After transmission, a stochastic process generates service times for the newly arrived data packet, that is, the delay between its arrival time and the time when it will be ready to be put in the buffer queue of the selected outgoing link.

We have not implemented a “real” transport layer. That is, we have not implemented mechanisms for a proper management of error, flow, and congestion control. The reason is that we want to check the behavior of our algorithm and of its competi-

tors in conditions which minimize the number of interacting components. Note that the dynamics of adaptive routing and of flow and congestion control components are tightly coupled and that they should therefore be designed to match each other.

4. AntNet: Adaptive Agent-based Routing

As remarked before, the routing problem is a stochastic distributed multi-objective problem. These features make the problem well suited for a multi-agent approach like our AntNet system, composed of two sets of *homogeneous mobile agents* [21], called in the following *forward* and *backward* ants. Agents in each set possess the same structure, but they are differently situated in the environment; that is, they can sense different inputs and they can produce different, independent outputs. In AntNet we retain the core ideas of the ant colony optimization paradigm, but we have translated them to match a distributed, dynamic context, different from combinatorial optimization. Ants communicate in an undirected way according to the stigmergy paradigm, through the information they concurrently read and write in two data structures stored in each network node k:

(i) an array $M_k(\mu_d, \sigma_d^2)$ of data structures defining a simple parametric statistical model of the traffic distribution for all destinations d, as seen by the local node k,

(ii) a routing table, organized as in distance-vector algorithms [20]; the table stores for each pair (d,n) a probability value P_{dn}

$$\sum_{n \in N_k} P_{dn} = 1, \; d \in [1, N], \; N_k = \{neighbors(k)\}$$

which expresses the goodness of choosing n as next node when the destination node is d.

The AntNet algorithm can be informally described as follows.

- At regular intervals, from every network node s, a forward ant $F_{s \to d}$, is launched, with a randomly selected destination node d. Destinations are chosen to match the current traffic patterns.
- Each forward ant selects the next hop node using the information stored in the routing table. The next node is selected, following a random scheme, proportionally to the goodness (probability) of each not still visited neighbor node and to the local queues status. If all neighbors have been already visited a uniform random selection is applied considering all the neighbors.
- The identifier of every visited node k and the time elapsed since its launching time to arrive at this k-th node are pushed onto a memory stack $S_{s \to d}(k)$ carried by the forward ant.
- If a cycle is detected, that is, if an ant is forced to return to an already visited node, the cycle's nodes are popped from the ant's stack and all the memory about them is destroyed.
- When the ant $F_{s \to d}$ reaches the destination node d, it generates a backward ant $B_{d \to s}$, transfers to it all of its memory, and then dies.
- The backward ant makes the same path as that of its corresponding forward ant, but in the opposite direction. At each node k along the path it pops its stack $S_{s \to d}(k)$ to know the next hop node.
- Arriving in a node k coming from a neighbor node f, the backward ant updates M_k and the routing table for all the entries corresponding to every node i on the path $k \to d$, that is the path followed by ant $F_{k \to d}$ starting from the current node k.

- The sample means and variances of the model $M_k(\mu_i, \sigma_i^2)$ are updated with the trip times $T_{k \to i}$ stored in the stack memory $S_{s \to d}(k)$.
- The routing table is changed by incrementing the probabilities P_{if} associated with node f and the nodes i, and decreasing (by normalization) the probabilities P_{in} associated with the other neighbor nodes n. Trip times $T_{k \to i}$ experienced by the forward ant $F_{s \to d}$ are used to assign the probability increments.

$T_{k \to d}$ is the only explicit feedback signal we have: it gives an indication about the goodness r of the followed route because it is proportional to its length from a physical point of view (number of hops, transmission capacity of the used links, processing speed of the crossed nodes) and from a traffic congestion point of view[1]. The problem is that $T_{k \to d}$ can only be used as a reinforcement signal. In fact, it cannot be associated with an exact error measure, given that we do not know the optimal trip times, which depend on the net load status. The values stored in the model M_k are used to score the trip times by assigning a goodness measure $r \equiv r(T_{k \to d}, M_k)$, $r \in]0,1]$ (r is such that the smaller $T_{k \to d}$, the higher r). This dimensionless value takes into account an average of the observed values and of their dispersion: $r \propto (1 - W_{k \to d}/T_{k \to d}) + \Delta(\sigma, W)$, where $W_{k \to d}$ is the best trip time experienced over an adaptive time window, and $\Delta(\sigma, W)$ is a correcting term (the rationale behind this choice for r is discussed in [8]); r is used by the current node k as a positive reinforcement for the node f the backward ant $B_{d \to s}$ comes from. The probability P_{df} is increased by the computed reinforcement value r: $P_{df} \leftarrow P_{df} + (1 - P_{df}) \cdot r = P_{df} \cdot (1 - r) + r$. In this way, the probability P_{df} will be increased by a value proportional to the reinforcement received and to the previous value of the node probability (that is, given a same reinforcement, small probability values are increased proportionally more than big probability values).

Probabilities P_{dn} for destination d of the other neighboring nodes n implicitly receive a negative reinforcement by normalization. That is, their values are reduced so that the sum of probabilities will still be 1: $P_{dn} \leftarrow P_{dn} \cdot (1 - r)$.

The transformation from the raw value $T_{k \to d}$ to the definition of the more refined reinforcement r is similar to what happens in Actor-Critic systems [1]: the raw reinforcement signal ($T_{k \to d}$, in our case) is processed by a critic module which is learning a model of the underlying process, and then is fed to the learning system.

It is important to remark that every discovered path receives a positive reinforcement in its selection probability. In this way, not only the (explicit) assigned value r plays a role, but also the (implicit) ant's arrival rate.

An important aspect of the AntNet algorithm is that the routing tables are used in a probabilistic way not only by the ants, but also by the packets. This mechanism allows an efficient distribution of the data packets over all the good paths and has been observed to significantly improve AntNet performance. A node-dependent threshold value avoids the choice of low probability links.

As a last consideration, note the critical role played by ant communication. In fact, each ant is complex enough to solve a single sub-problem but the global routing optimization problem cannot be solved efficiently by a single ant. It is the interaction between ants that determines the emergence of a global effective behavior from the network performance point of view. The key concept in the cooperative aspect lies in

[1] This last aspect is extremely important: forward ants share the same queues as data packets (backward ants do not, they have priority over data to faster propagate the accumulated information), so if they cross a congested area, they will be delayed. This has a double effect: (i) the trip time will grow and then back-propagated probability increments will be small, and (ii) at the same time these increments will be assigned with a bigger delay.

the indirect and non-coordinated way communication among ants happens (stigmergy, [15]). We used stigmergy as a way of recursively transmitting, through the nodes' structures, the information associated with every "experiment" made by each ant.

5. Routing Algorithms Used for Comparison

The following algorithms, belonging to the various possible combinations of static and adaptive, distance vector and link state classes [20], have been implemented and used to run comparisons. **OSPF** (static, link state) is our implementation of the official Internet routing algorithm [17] (since we did not consider failure conditions the algorithm reduces to static shortest path routing). **SPF** (adaptive, link state) is the prototype of link-state algorithms with dynamic metric for link costs evaluations. A similar algorithm was implemented in the second version of ARPANET [16]. We implemented it with state-of-the-art flooding algorithms and link cost metrics [19]. Link costs are evaluated over moving windows using a link usage metric based on the fraction of time the link has been used during the last observation window. This metric was the most effective among the several we considered. **BF** (adaptive, distance-vector) is an adaptive implementation of the distributed Bellman-Ford algorithm with dynamic metrics [3]. Link costs are evaluated as in SPF above. **Q-R** (adaptive, distance-vector) is the Q-routing algorithm as proposed in [5]. This is an online asynchronous version of the Bellman-Ford algorithm. **PQ-R** (adaptive, distance-vector) is the Predictive Q-routing algorithm [6], an extension of Q-routing.

6. Experimental Settings

We have selected a limited set of classes of tunable components and for each of them we have made realistic choices.

Topology and physical properties of the net. In our experiments we used two networks: NSFNET and an irregular 6 x 6 grid. NSFNET is a real net, that is, the old (1987) T1 US backbone, while the 6 x 6 grid was proposed in [5]. NSFNET is composed of 14 nodes and 21 bi-directional links, and the 6 x 6 grid has 36 nodes and 50 bi-directional links. The topology and the propagation delays of NSFNET are those used in [8], while for the 6 x 6 grid see [5]. Links have a bandwidth of 1.5 Mbit/s in NSFNET, and 10 Mbit/s in the 6 x 6 grid net. All nets have null link and node fault probabilities, local buffers of 1 Gbit, and packets maximum time to live set to 15 sec.

Traffic patterns. Traffic is defined in terms of open sessions between a pair of active applications situated on different nodes. We considered three basic spatial and temporal traffic pattern distributions:

- Uniform Poisson (UP): for each node is defined an identical Poisson process for sessions arrival, that is, inter arrival times are negative exponential distributed.
- Hot Spots (HS): some nodes behave as hot spots, concentrating a high rate of input/output traffic. Sessions are opened from the hot spots to all the other nodes.
- Temporary Hot Spot (TMPHS): a temporary sudden increase in traffic load is generated switching on some hot spots for a limited period of time.

All the experiments have been realized considering various compositions of the above main patterns. For all the session types, packets sizes, packets inter arrival times and the total number of generated bits follow a negative exponential distribution.

Metrics for performance evaluation. We used two standard performance metrics: *throughput* (delivered bits/sec), and *data packets delay* (sec). For data packets delay we use either the average value over a moving time window, or the empirical distribution that takes into account the intrinsic variability of packet delays.

Routing algorithms parameters. For each algorithm the routing packets size and elaboration time are reported in Table 1. The other main parameters are the following. In AntNet, the generation interval of ants is set to 0.3 (sec), the exploration probability is set to 0.05, and the ant processing time is set to 3 ms. In OSPF, SPF, and BF, the length of the time interval between consecutive routing information broadcasting and the length of the time window to average link costs are the same, and they are set to 0.8 or 3 seconds, depending on the experiment. In Q-R and PQ-R the transmission of routing information is data-driven.

Table 1. Routing packets characteristics for the implemented algorithms. N_h is the incremental number of hops done by the forward ant, N_n is the number of neighbors of node n, and N is the number of network nodes.

	Ant Net	SPF & OSPF	BF	Q-R & PQ-R
packet size (byte)	$24+8\cdot N_h$	$64+8\cdot N_n$	$24+12\cdot N$	12
packet elaboration time (msec)	3	6	2	3

7. Results

The goal of a routing algorithm is to route all the generated traffic, without losses, while keeping packets delay as low as possible (i.e., it should operate the network far from saturation conditions). Moreover, packet losses would require retransmission (this is managed by the congestion control layer, which is not implemented in our simulator) with a further increase in traffic. Therefore, when observing the results presented in the following of this section, the first performance comparison will be done on throughput, and a fair comparison on packet delays can only be done for those algorithms which have a similar throughput.

Experiments reported in this section compare AntNet with the previously described routing algorithms. All experiments are averaged over 10 trials. Parameters values for traffic characteristics are given in the figures' captions with the following meaning: NHS is the number of hot spot nodes, MSIA is the mean of the sessions inter arrival time distribution, MPIA-UP and MPIA-HS are the means of the packet inter arrival time distributions for the UP and HS sessions respectively. In all the experiments the mean of the packet size distribution is set to 4096 bit, and the mean of the total number of bits produced by each session is set to 2 Mb.

The results obtained on the NSFNET for (i) a uniform Poisson traffic load (UP) distribution, (ii) hot spots superimposed to a uniform Poisson traffic load (UPHS), and (iii) temporary hot spots superimposed to a light UP load, are shown respectively in figures 1, 2, and 3. The uniform Poisson traffic was chosen to be "heavy", that is, we set the values of the traffic patterns parameters to values that caused the network to reach a state very close to saturation. The reason to do this is that it is only under heavy load conditions that differences among competing algorithms can be appreciated in a meaningful way. In fact, when the traffic load is low, almost all the algorithms perform similarly. On the other hand, if the traffic load is too high, then a reasonable assumption is that it is a temporary situation. If it is not, structural changes to the network characteristics, like adding new and faster connection lines, rather than

improvements of the routing algorithm, are in order. In both figures 1 and 2, the bigger, outer graph shows the throughput, while the smaller, inner graph shows the empirical distribution of packet delays. From these two figures we can extract the following information: (i) all the algorithms, with the exception of OSPF and PQ-R, can successfully route the totality of the generated throughput, and (ii) AntNet is the only algorithm capable of maintaining the packet delay of more than 90% of the packets below 0.5 sec.

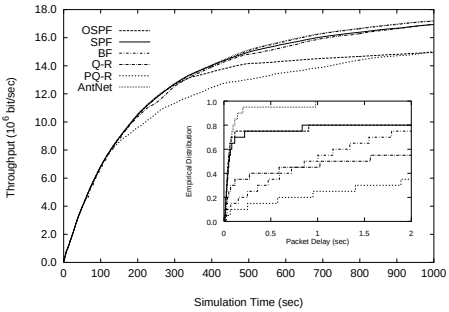

Fig. 1. NSFNET: A comparison of AntNet with five competing algorithms for a heavy uniform Poisson traffic (UP). Average over 10 trials. MSIA=1.5, MPIA-UP=0.2.

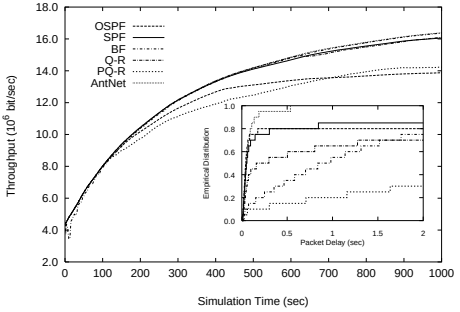

Fig. 2. NSFNET: A comparison of AntNet with five competing algorithms for hot spots superimposed to a heavy uniform Poisson traffic (UPHS). Average over 10 trials. NHS=4, MSIA=2.0, MPIA-UP=0.3, MPIA-HS=0.05.

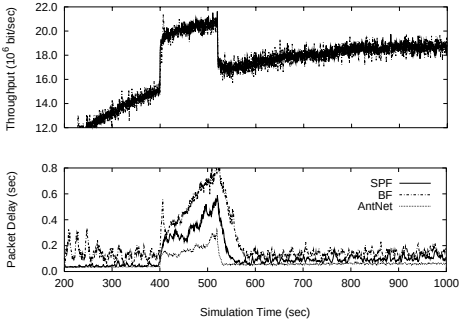

Fig. 3. NSFNET: A comparison of AntNet with SPF and BF (the behavior of the other algorithms was much worse): after 400 sec some hot spots are superimposed to a light UP load for 120 sec (TMPHS-UP). Reported throughput and packet delay values are averages over a moving time window of 0.5 sec. Average over 10 trials. NHS=4, MSIA=1.5, MPIA-UP=0.5, MPIA-HS=0.05.

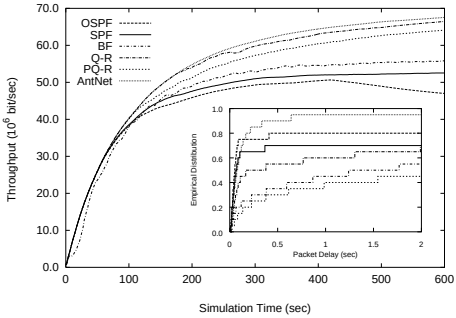

Fig. 4. 6 x 6 irregular grid net: A comparison of AntNet with five competing algorithms for a heavy uniform Poisson traffic (UP). Average over 10 trials. MSIA=1.0, MPIA-UP=0.1.

In Fig. 3 we investigate the answer of the algorithms to a sudden increase of traffic load. During the whole simulation the network is given a light UP load distribution; at simulation time t=400 the hot spots are switched on, to be subsequently switched off at time t=520. The figure shows only AntNet, SPF, and BF since the other algorithms' performance was so much worse that their graphs were out of scale. The upper graph in Fig. 3 shows the instantaneous throughput averaged over a window of 0.5 sec. It is clear that there are no significant differences among AntNet, SPF, and BF: their

graphs are practically superimposed on the whole simulation time. The three algorithms increase their throughput when the hot spots are switched on, and, once the hot spots are switched off, they quickly forget the transitory situation. The lower graph (Fig. 3) shows the instantaneous packet delay averaged over a window of 0.5 sec. Here we chose not to report the empirical distribution because we wanted to highlight the answer in time of algorithms. The graph confirms the superiority of AntNet over the other algorithms also in this case: after time t=400 the average packet delay greatly increases for all the algorithms, except for AntNet which is able to maintain it well below 0.4 sec.

As a last experiment, in Figure 4 we present results obtained on the irregular 6 x 6 grid network. Once again, AntNet offers the best throughput, although differences with Q-R and PQ-R are not statistically significant. BF, SPF, and OSPF lose up to 20% of the packets. The inner graph shows for the empirical distribution of packet delays a similar pattern to that of experiments with NSFNET: AntNet is the best algorithm, followed, in this case, by OSPF and SPF (that, however, had a worse throughput performance).

8. Discussion

The results presented are rather sharp: AntNet performs better, both in terms of throughput and of average delay, than both classic and recently proposed algorithms[2]. Among the competitors there is not a clear winner.

Concerning network resources utilization, in Table 2 we report, for each algorithm, the routing overhead expressed as the ratio between generated routing traffic and total available bandwidth. Even if the AntNet overhead is higher than that of some of its competitors, it must be considered that (i) the relative weight of the routing packets on the net resources is negligible, and (ii) this slightly higher network resources consumption is compensated by the much higher performance it provides.

Table 2. Routing overhead for experimental conditions considered in the paper expressed as the ratio between the generated routing traffic by each algorithm and total available network bandwidth (note that the ratios are scaled by a factor of 10^{-3}).

	AntNet	OSPF	SPF	BF	Q-R	PQ-R
NSFNET UP (10^{-3})	1.70	<0.10	0.40	0.69	8.6	10.0
NSFNET UPHS (10^{-3})	1.70	<0.10	0.47	0.69	8.1	10.0
6x6(10^{-3})	2.30	<0.10	0.16	0.24	8.0	9.4

Differences among algorithms performances can be understood on the basis of the different degree of adaptivity and of speed with which the different algorithms respond to changing traffic conditions. The very low performance of OSPF can be explained by both the lack of use of an adaptive metric (which all the other methods use), and by the fact that we set link costs only on the basis of a shortest path computation. Differently, on real networks (on the Internet, for example) these are set by network administrators who use additional heuristic knowledge about traffic patterns. To explain why AntNet performs better than theothers is slightly more tricky. We identified the following main reasons: (i) the use of local versus global information, and (ii) the different routing table update frequencies, which are discussed in the following.

[2] Experiments similar to those presented in this paper have been run on other network topologies with increasing number of nodes and different traffic patterns obtaining similar results (see [8, 9, 10]).

The use of local versus global information. BF, Q-R and PQ-R work with local estimates of distances to destinations. These estimates are updated by using strictly local information: the traffic situation on outgoing links and the distance estimates maintained by neighbor nodes. Differently, AntNet samples the network and redistributes the global information ants collect: backward ants redistribute the global information relative to the paths sampled by the corresponding forward ants to all the nodes they visited. SPF maintains a global representation of the whole network in each node, which is updated by periodic flooding of local link costs information. If one of this cost information is badly estimated (as it is often the case when dynamic metrics are used), the wrong estimate propagates to all the local representations of the network. Here it is used to calculate shortest paths to build the new routing tables. The result is that a single erroneous estimate will negatively affect all the routing tables. From this point of view, AntNet is more robust: an incorrect update will affect only entries relative to the ant destination in those routing tables belonging to the ant path.
Routing table update frequency. In BF and SPF the broadcast frequency of routing information plays a critical role, particularly so for BF which has only a local representation of the network status. This frequency is unfortunately problem dependent, and there is no easy way to make it adaptive, while, at the same time, avoiding large oscillations. In Q-R and PQ-R, routing tables updating is data driven: only those Q-values belonging to pairs (i,j) of nodes visited by packets are updated. Although this is a reasonable strategy, given that the exploration of new routes could cause undesired delays to data packets, this causes delays in discovering new good routes, and is a great handicap in a domain where good routes change all the time. In AntNet, we experimentally observed the robustness to changes in the ants' generation rate. For a wide range of values of the generation rate, the more the ants generated, the better the algorithm works, until the traffic induced by ants ceases to be negligible with respect to the data traffic.

9. Related Work

AntNet is not the only algorithm based on the ant colony metaphor that has been applied to routing. Schoonderwoerd et al. [18] have considered the routing problem in connection-oriented communications networks. Their approach is different from ours in many respects. First, their communication network was modeled after a very specific type of telephone network: (i) they considered links carrying an infinite number of full-duplex, fixed bandwidth channels, (ii) their nodes are just reconfigurable switches with limited connectivity (that is, there is no necessity for queue management in the nodes). Second, their ants did not share transmission channels with data packets (i.e., they used a "virtual" network). These assumptions are strongly reflected in their algorithm structure: since we use a realistic communications network simulator which models a general data network, it is impossible to re-implement and compare their algorithm with ours.

10. Conclusions

In this paper we proposed AntNet, a novel algorithm for routing in communications networks inspired by previous work on artificial ants colonies in combinatorial optimization. We compared AntNet to a set of state-of-the-art algorithms using a realistic network simulator using the T1-NSFNET network and an irregular 6 x 6 grid network

as benchmark problems. In all the experiments we ran, AntNet had the best distribution of packet delays, and was among the best algorithms as far as throughput was concerned. More, AntNet showed a robust behavior under the different traffic conditions and the ability to reach a stable behavior very quickly. AntNet had also, as well as OSPF, SPF and BF, a negligible impact on the use of network bandwidth.

References

1. Barto A.G., Sutton R.S., & Anderson C.W. 1983. Neuronlike Adaptive Elements that Can Solve Difficult Learning Control Problems. *IEEE Tr. on Syst., Man, and Cyb*. 13: 834–846.
2. Beckers R., Deneubourg J.L., & Goss S. 1992. Trails and U-turns in the Selection of the Shortest Path by the Ant Lasius Niger. *J. of Theor. Biology* 159:397–415.
3. Bertsekas D. & Gallager R. 1992. *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall.
4. Bonabeau E., Dorigo M., & Theraulaz T. (in press). *From Natural to Artificial Swarm Intelligence*. Oxford University Press.
5. Boyan J. A. & Littman M. L. 1994. Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach. In *Proc. of NIPS-6*, San Francisco, CA: Morgan Kaufmann, 671–678.
6. Choi S. P. M. & Yeung D.-Y. 1996. Predictive Q-Routing: A Memory-based Reinforcement Learning Approach to Adaptive Traffic Control. In *Proc. of NIPS-8*, Cambridge, MA: The MIT Press, 945–910.
7. Costa D. &, Hertz A. 1997. Ants Can Colour Graphs. *J. of the Oper. Res. Soc*. 48:295-305.
8. Di Caro G. & Dorigo M. 1998. AntNet: Distributed Stigmergetic Control for Communications Networks. *Tech.Rep. IRIDIA/98-01*, Université Libre de Bruxelles, Belgium. To appear in *Journal of Artificial Intelligence Research (JAIR)*.
9. Di Caro G. & M. Dorigo 1998. An adaptive multi-agent routing algorithm inspired by ants behavior. *Proceedings of PART98 – Fifth Annual Australasian Conference on Parallel and Real-Time Systems*, September 28–29, 1998, University of Adelaide, Australia, in press.
10. Di Caro G. & M. Dorigo 1998. Distributed Adaptive Routing by Artificial Ant Colonies. *PDCS'98 – 1998 International Conference on Parallel and Distributed Computing and Systems*, October 28–31, 1998, Las Vegas, Nevada.
11. Dorigo M. 1992. *Ottimizzazione, Apprendimento Automatico, ed Algoritmi Basati su Metafora Naturale* (Optimization, Learning and Natural Algorithms). Ph.D.Thesis, Politecnico di Milano, Italy (in Italian), pp.140.
12. Dorigo M. & Gambardella L. M. 1997. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Trans. on Evol. Comp.* 1(1): 53–66.
13. Dorigo M., Maniezzo V., & Colorni A. 1991. Positive Feedback as a Search Strategy. *Tech. Rep. No. 91-016*, Dip. di Elettronica, Politecnico di Milano, Italy.
14. Dorigo M., Maniezzo V., & Colorni A. 1996. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Trans. on Syst., Man, and Cybern.–Part B* 26(2): 29–41.
15. Grassé P. P. 1959. La reconstruction du nid et les coordinations inter-individuelles chez *Bellicositermes natalensis et Cubitermes sp*. La théorie de la stigmergie: Essai d'interprétation des termites constructeurs. *Insect Sociaux* 6: 41–83.
16. McQuillan J. M., Richer I., & Rosen E. C. 1980. The New Routing Algorithm for the ARPANET. *IEEE Trans. on Commuications* 28:711–719.
17. Moy J. 1994. OSPF Version 2. Request for Comments (RFC) 1583, Network Work Group.
18. Schoonderwoerd R., Holland O., Bruten J., & Rothkrantz L. 1996. Ant-based Load Balancing in Telecommunications Networks. *Adaptive Behavior* 5(2):169–207.
19. Shankar A. U., Alaettinoglu C., Dussa-Zieger K., & Matta I. 1992. Performance Comparison of Routing Protocols under Dynamic and Static File Transfer Connections. *ACM SIGCOMM Computer Communication Review* 22(5): 39–52.
20. Steenstrup M. E. (ed.) 1995. *Routing in Communications Networks*. Prentice-Hall.
21. Stone P. & Veloso M. 1996. Multiagent Systems: A Survey from a Machine Learning Perspective. *Tech. Rep. CMU-CS-97-193*, Carnegie Mellon University, PA.