Q-Net: A Novel QoS Aware Routing Algorithm for Future Data Networks

Maassoumeh Javadi Baygi, Abdul Rahman B Ramli, Borhanuddin Mohd Ali, Syamsiah Mashohor

Abstract—The expectation of network performance from the early days of ARPANET until now has been changed significantly. Every day, new advancement in technological infrastructure opens the doors for better quality of service and accordingly level of perceived quality of network services have been increased over the time. Nowadays for many applications, late information has no value or even may result in financial or catastrophic loss, on the other hand, demands for some level of guarantee in providing and maintaining quality of service are ever increasing. Based on this history, having a QoS aware routing system which is able to provide today's required level of quality of service in the networks and effectively adapt to the future needs, seems as a key requirement for future Internet. In this work we have extended the traditional AntNet routing system to support QoS with multiple metrics such as bandwidth and delay which is named Q-Net. This novel scalable QoS routing system aims to provide different types of services in the network simultaneously. Each type of service can be provided for a period of time in the network and network nodes do not need to have any previous knowledge about it. When a type of quality of service is requested, Q-Net will allocate required resources for the service and will guarantee QoS requirement of the service, based on target objectives.

Keywords—Quality of Service, Routing, Ant Colony Optimization, Ant-based algorithms.

I. INTRODUCTION

TRANSFERRING messages in reliable fashion with error control and notification about undelivered messages is a common task in many modern communication systems. However, recently the ability to specify timeliness and perceived quality of arriving data has been received too much attention. The underlying concepts of bandwidth, throughput, timeliness (including jitter), reliability, perceived quality and cost are the foundations of what is known as Quality of Service (QoS) [1]. In [1] author divides QoS Characteristics into two categories: Technology-Based QoS Characteristics (which includes Timeliness, Bandwidth and Reliability) and User-Based QoS Characteristics (which includes Criticality, Perceived Quality, Cost and Security). A further important classification of QoS requirements, or more particularly the systems implementing the requirements, is the class of service provided. In [2] author subdivides classes of service (CoS) into five levels: 1)Deterministic guarantee 2)Statistical guarantee 3)Target objectives 4)Best effort 5)No guarantee. Deterministic guarantees will always be met, under all circumstances, while a statistical guarantee allows a percentage of time where the guarantee is not met. The last three levels provide no real guarantee, but offer varying levels of assistance in achieving the desired QoS. A best effort system, like the Internet, would provide the same QoS for all services (i.e. with no real consideration of QoS factors.)*.*

QoS Routing is the missing piece in a full-fledged QoS architecture for the Internet because all current IETF standards are based on traditional routing that is unaware of QoS. If we take the viewpoint that routing consists of a routing algorithm (static) and routing protocol (dynamics), then a QoS Routing algorithm solves the Multi-Constrained (Optimal) Path (MC(O)P) routing problem [3]. A difficulty of the MC(O)P problem is that, it is NPcomplete[4]; which, in turn, has stimulated proposals for heuristic approaches. QoS routing protocol consists of all actions that inform the individual nodes on the network structure with a consistent and timely fashion. QoS routing protocol is the most difficult problem currently hindering the implementation of QoS in the Internet [3].

The main goal of this work is designing a routing system which is able to provide today's required level of quality of service in the networks and effectively adapt to the future needs. We have extended the traditional AntNet routing system to support QoS with multiple metrics such as bandwidth and delay. This novel scalable QoS routing system aims to provide different types of services in the network simultaneously. Each type of service can be provided for a period of time in the network and network nodes do not need to have any previous knowledge about it. When a type of quality of service is requested, Q-Net will allocate required resources for the service and will guarantee QoS requirement of the service, based on target objectives. In section II we will review some related work. In section III we will describe the main concept of proposed QoS aware routing Q-Net, then in section IV we will detail the efficiency, scalability, and extensibility of Q-Net. Section V concludes the paper.

II. RELATED WORKS

In this section we survey ACO approaches in network routing and load-balancing. An artificial ant is typically a simple object which has simple methods for laying and sensing of pheromone, and data structures that record network metrics and the nodes that it passes. When it passes from node to node, an ant simulates laying of pheromone by updating the corresponding entry in the routing table of nodes.

Caro and Dorigo's AntNet [5], [6], [7], [8] was first developed for routing in packet switched networks. In contrast to traditional routing algorithms (such as OSPF and RIP) which concern about minimal or shortest path routing, routing in AntNet was conducted to optimize the performance of the entire network. In AntNet, routing has been made by launching forward ants at regular intervals from a source node to a

M. Javadi is with the Institute of Advanced Technology (ITMA) ,University Putra Malaysia(UPM), 43400 UPM SERDANG, Selangor

destination node to explore a possible low cost route and by ants that moves backwards from the destination node to source node to update the pheromone tables at each intermediate node. A forward ant chooses the next hop using a random pattern that takes into account both 1) the probability of choosing next hop, and 2) a heuristic correction factor. Favorable results in terms of higher throughput and lower average delay have been made in AntNet compared to other routing approaches.

In MACO, more than one colony of ants is used to find optimal paths, and each colony of ants deposit different type of pheromone. Although the ants in each colony respond to the pheromone of its own colony, MACO is powered with a mechanism of repulsion [9], [10] that prevents ants from other colonies to choose the same optimal path. Adopting the MACO, it may be possible to reduce the likelihood that all mobile agents establish connections using only the optimal path.

ARS [11] is another agent-based routing system (ARS) with QoS routing. ARS effectively allocates network resources for users requiring real-time communications with its resource management mechanism, where mobile agents gather current link states to test readiness of potential routes maintained in each node. ARS works on a datagram network, users of the network requires two classes of service: datagram and real-time flow. A datagram is a delay-insensitive packet, while the real-time stream is a sequence of delay sensitive packets.

All algorithms mentioned above have scalability problems, since each node must send an ant to all other nodes in the network; this means that the total number of ants that should be sent is $N \times (N-1)$.

For large networks, the volume of traffic generated by the ants would be costly. In addition, for a long-path there is a greater risk of losing ants. Also, the long journey of ants makes the information that they carry, outdated. On the other hand both original versions of AntNet, ABC, [12], and most recent QoS routing algorithms consider only a single metric to characterize routes in a network, such as hop-count, delay, cost, etc, while the use of multiple metrics is needed to better characterize a network and to support a wide range of QoS requirements. ARS supports only two types of service,and none of the other mentioned algorithms support multiple types of services (ToS) in the network. From all mentioned algorithms only ARS uses resource reservation mechanism to guarantee the QoS specification of the session. All of these algorithms have innumerable advantages, however lack above mentioned features. Therefore, we try to take advantage of their strengths and to avoid their weaknesses to propose a potentially powerful QoS algorithm Q-Net.

III. NOVEL Q-NET ROUTING ALGORITHM

Our method is based on the so-called AntNet routing algorithm, an ant-based algorithm developed by G. Di Caro and M. Dorigo, where mobile agents search for short routes in packet switching networks. In AntNet, the information

processed by ants is represented in every node by a routing table and an additional data structure that contains local traffic – delay statistics. In the routing table, every value of P_{ij} represents the probability or appropriateness of choosing j as next node when the destination node is i. These values are calculated according to the information gathered by forward ants and modified by taking into account historical aspects or statistics of local traffic.
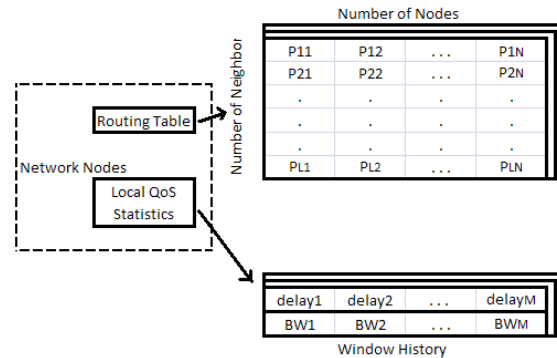

Fig. 1: Data Structure of a Node

Unlike the original AntNet we have used multiple ant colony idea to support different types of service (ToS) (ToS such as premium, best effort, etc.) in the network simultaneously. Therefore in each node we will use one separate routing table and additional data structure for local traffic for each type of service. In order to reduce memory cost associated with having separate routing tables, routing tables for QoS services are dynamically created when needed and will be destroyed when not used anymore. Furthermore local traffic data structures will contain all required metrics (e.g. Delay, Bandwidth, ...) according to its type of service. In order to use the network resources efficiently, each router will have the best effort routing table initially, then if it receives an ant from other colony it will create a routing table for that colony. This routing table will be deleted and the memory will be freed if the router does not receive any ant from that colony for a specified amount of time. In initialization phase of the network all routers will create ants from best effort class to expedite the initial convergence of the network. Two methods are devised to expedite convergence of the other type of routing tables when these tables are created. 1) Other ant colonies will copy best effort routing table when they are created, 2) Colonies with more strict QoS requirements will copy routing table of colonies with less strict QoS requirements. Every node in the network supports the weighted fair queuing (WFQ) algorithm [13], [14]. The WFQ algorithm evenly divides every link transmission capacity in a network into N pieces. One piece is permanently allocated to best effort type of service, and another piece is permanently allocated to agents, while the other pieces (N- 2 pieces) are allocated by other QoS types of service flow upon request. Best effort type of service can use more share from the link capacity when there are unused pieces of link transmission capacity.

Five types of agents are used in Q-Net, forward explorer

ants (A_{fwd}), backward ants (A_{back}), resource reservation ants (A_{res}), resource releasing agents (A_{rel}) and information agents (A_{info}). Each ant colonies will have their own agents. Two types of ants (forward ants and backward ants) are in a manner similar to the traditional AntNet .

1) Froward Agents: Forward ants are exploration ants that have the goal of discovering new routes and constantly evaluate the state of existing routes, analyzing the delay and the available or residual bandwidth.

2) Backward Agents: At each node along the return path, backward ants update local traffic data structures in every node of the QoS data model and the routing table.

3) Reserve Agents: When QoS session is about to establish a source node s launches a reserve agent toward the destination d to reserve resources along the feasible path from source to destination.

4) Release Agents: When QoS session ends or upon unsuccessful reservation, a release agent is launched to release reserved resources.

5) Info Agents: Info Agents are used by QoS management system and will be explained later.

For each class of service, an independent forward ant, A_{fwd} that examines the network load for that traffic class, is activated at regular time intervals t from each server s towards a destination client d. The destination nodes are locally selected according to the probability Pd, which is related to the traffic patterns generated by each server related to each ant colony. Pd represents the probability of activating an ant from the node s to the node d, and therefore it adapts the exploratory activity of forward ants according to the current traffic distribution for each traffic class. In order to minimize the routing overhead each server only launches ants from the class of service that it provides. Each forward ant at each node will save the current time. Best effort agents will save the estimation of time which a normal packet will wait to exit from the interface of selected next hop. When forward explorer ant reaches the destination node, it will launch a backward ant, transfer its entire stack to this ant and will die. Each backward ant A_{back} will go exactly through the hops that forward ant visited previously, When each backward ant arrives at a node k from a neighboring node f, the backward ant A_{back} updates local traffic data structures of the node.

A. Selecting Next Hop

Every best effort data packet P that is in a transit node k, selects the next hop node n among all feasible neighboring nodes, omitting last visited node. The neighbor node n is selected according to probability Pn, which is slightly the modified version of the probabilities in the routing table of best effort ant colony. Every QoS data packet P that belongs to a QoS session and is in a transit node k, will be routed through the reserved resources of that specific QoS session. Every forward ant A_{fwd} that is in a transit node k, selects the next hop node n among all feasible neighboring nodes, omitting last visited node. The neighbor node n is selected according to the probability Pn, which is related to the probabilities in the

routing table of each ant colony. As in traditional AntNet, if a cycle is detected, the nodes in this cycle are removed from the ant's memory stack. If the cycle is too long, the ant is destroyed to disable ants that are carrying old and invalid values about the network state. Each backward ant A_{back} will go exactly through hops that forward ant has visited previously. Every reserve ant A_{res} that is in a transit node k, selects the next hop node n among all feasible neighboring nodes which satisfies the type of service constraints of QoS session, omitting last visited node. The neighbor node n is selected according to the probability Pn, which is related to the probabilities in the routing table of each ant colony. If a cycle is detected or if all the neighboring nodes do not satisfy the type of service constraints of QoS session, reserve agent will die and a release agent will be launched. Each release ant A_{rel} will go exactly through the hops that the reserve ant has visited previously. Every info agent that is in a transit node k, selects the next hop node n among all feasible neighboring nodes, omitting last visited node. The neighbor node n is selected according to the probability Pn, which is slightly the modified version of the probabilities in the routing table of agent's ant colony.

B. Updating Routing Tables

When each backward ant arrives at a node k from a neighboring node f, the backward ant A_{back} updates local traffic data structures of the node. The local QoS statistics or traffic data are structures with information about the route from node k to each node i of the network. According to the type of service of each ant colony, local traffic data structure could have following information.

1) Delay structure contains the delay averages and their respective degree of trust (e.g. trust intervals). Using these values, the probability Pd of obtaining the requested delay conditions for the different classes of service is calculated (each backward ant can only reflect its experience about the network status for its own class).

2) Bandwidth structure contains the minimum values of remaining detected bandwidth (Available bandwidth) for each type of service (hop by hop), and its respective degree of trust. With these values, the probability P_{bw} of obtaining the requested bandwidth conditions for the different types of service is calculated.

At each node K a routing table similar to vector–distance algorithm (as used in traditional AntNet) is constructed for each type of service. In this table, a probability value P_{ij} is associated with the selection of node j as the next hop, for the requested type of service, when the destination node is i. The probability weights are subject to the following constraint:

$$\sum_{j=1}^{N_k} P_{ij} = 1 \qquad (1)$$

where $i \in [1,N]$ and $N_k = \{$ neighbor node of $k\}$. Probability P_{ij} is a function that depends on the probabilities of obtaining a requested QoS condition. The probability P_{ij} is incremented by a value proportional to received reinforcement

r (according to accomplishment of certain conditions of reinforcement and inspired by the original reinforcement used in AntNet) and to the previous value of the probability in its corresponding routing table.

$$P_{ij} \leftarrow P_{ij} + (1 - P_{ij}) * r \qquad (2)$$

The probability of other interfaces for the same destination decrease as follow.

$$P_{ij} \leftarrow P_{ij} - P_{ij} * r \qquad (3)$$

The reinforcement value r is proportional to reinforcement value associated with each QoS condition as follow

$$r = r_d \cdot g_d(t) * r_{bw} \cdot g_{bw}(t) \qquad (4)$$

C. QoS Management

For each class of service that network provides, we will run different ant colony. When a client requests specified type of service the QoS management will use respective routing table to provide the service. When a Server wants to initiate a QoS session, it tries to provide the requested class of service by reserving a line for the client. If this process was successful, server starts to send the requested data.

When a QoS session is about to start the server reserves the line by sending reserve agent A_{res} toward the client. Reserve agent walks through the network toward the client and at each router tells the router to reserve the line if such resource is available. If reservation is successful reserve agent launches a information agent A_{info} toward the server to inform about reservation success, on the other hand if the reservation is not successful the resource reservation agent launches a resource release agent A_{rel} toward the server to free up all the resources which are reserved by the resource reservation agent A_{res}.

If the reservation is successful server starts to send data packets, and at the end of session server will send a resource release agent A_{rel} to free up all resources. Upon unsuccessful attempt to reserve a line, server will mark this session as unsuccessful one. At the middle of a session, if a line drops accidentally, the adjacent nodes will launch release agents toward the server and the client to release the reserved resources, and the session will be marked as uncompleted one. If a router does not receive any packet for a reserved QoS session for a period of time, it will conclude that the session is failed somehow and the server is unable to inform the router about it, therefore the router will free up all resources which were reserved for that session.

D. Simulation and Results

The proposed algorithm is implemented in a C++. Two networks are modeled to check the algorithm performance. The first one is a simple network which its geometry is shown in Fig. 2. The second one is NFSNET network. We have assumed that the properties of these networks are as listed in Table I.
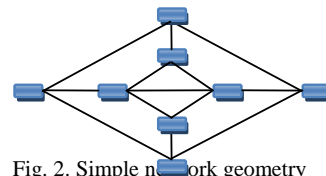

Fig. 2. Simple network geometry

TABLE I
PROPERTIES OF TWO SIMULATED NETWORKS

TABLE II
PROVIDED CLASS OF SERVICE IN TWO NETWORKS

| Class | Simple Net | | NFSNET | |
	Min-Delay	Min-Bw	Min-Delay	Min-Bw
C1: (Best Ef	No-limit	No-limit	No-limit	No-limit
C2:	4 ms	0.5 Mb/s	60 ms	0.5 Mb/s
C3:	3 ms	1.5 Mb/s	30 ms	1.5 Mb/s

In each simulation we have assumed three classes of service as listed in Table II. Fig. 3 shows number of successful sessions after 5 minutes from start of simulation for different rates of request arrival (λ) in simple net. Fig. 4 shows the same results in NFSNET. The square marked line is for the case where just two classes of service C1 and C2 are provided in each networks; the triangle marked line is for the case where all three classes of service are provided. Diamond marked line indicates total number of requests.
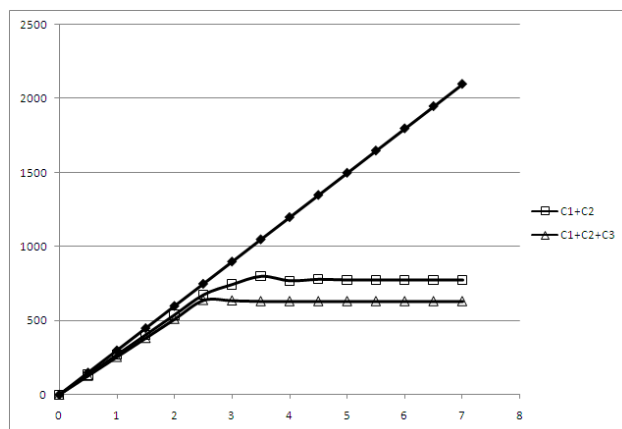

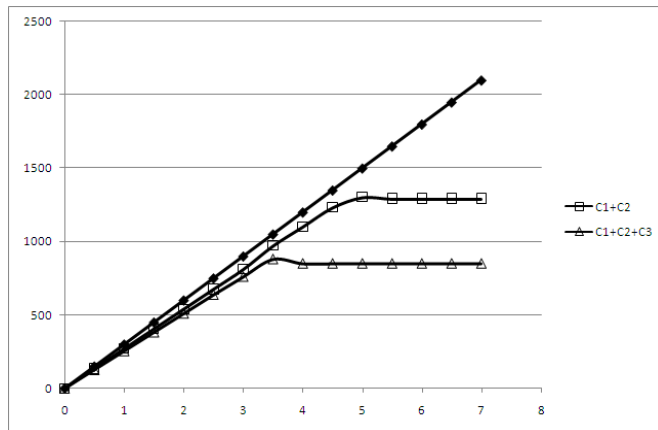Fig. 3: Number of successful requests vs. rate of request arrival λ (request/sec) in simple Net

Fig. 4: Number of successful requests vs. rate of request arrival λ (request/sec) in NFSNET

Fig. 5 shows percentage of used resources for different request arrival rates for simple Net. As can be seen in the figure, when request arrival rate increases, the percentage of used resources increases too, until it reaches a 50% of available bandwidth, Then from this point further, increment of request arrival rate will not change resource usage, significantly. Fig. 6 shows the same results for NFSNET.
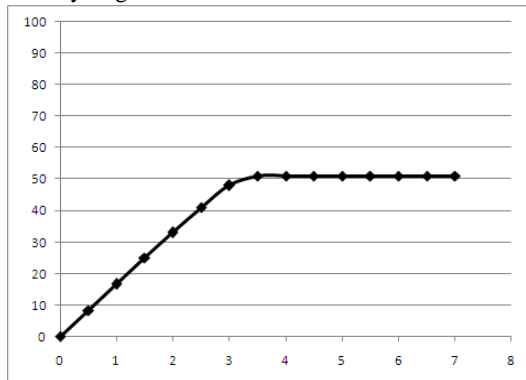

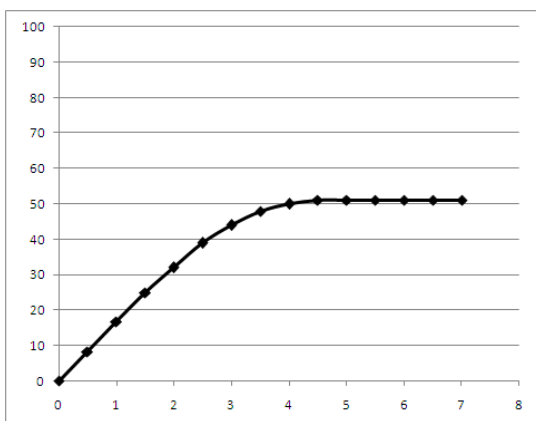Fig. 5: percentage of used resources vs. rate of request arrival λ (req/sec) in simple Net

Fig. 6: percentage of used resources vs. rate of request arrival λ (req/sec) in NFSNET

This paper shows that Q-Net effectively increases different types of services which could be provided in the network, virtually there is no limitation on the number of simultaneous types of service which could be provided. We have simulated two networks to check the scalability and performance of our routing algorithm. Resource utilization, throughput and ratio of successful sessions to unsuccessful ones for different request arrival rates are considered too. The simulation results show that performance of Q-Net is competent with other well-known routing algorithms.

REFERENCES

[1] D. Chalmers, M. Sloman: Survey of Quality of Service in Mobile Computing Environments, - IEEE Communications surveys, (1999).
[2] G. Bochmann,   A. Hafid: Some principles for quality of service management, Distrib. Syst. Engng. l4, 16-27 (1997).
[3] X. Masip-Bruin et al: Research challenges in QoS routing, Computer Communications. 29, 563–581 (2006).
[4] Z. Wang, J. Crowcroft: Quality of service routing for supporting multimedia applications, IEEEJSAC. 14, 7 (1996).
[5] G. D. Caro, M. Dorigo: AntNet: Distributed stigmergetic control for communications networks, J. Artif. Intell. Res. 9, 317-365 (1998).
[6] G. D. Caro, M. Dorigo: AntNet: A Mobile Agents Approach to Adaptive Routing, Univ. Libre de Bruxelles, Brussels, Belgium, Tech. Rep. IRIDIA/9712 (1997).
[7]   G. D. Caro, M. Dorigo: Mobile agents for adaptive routing, In Proc. 31st Hawaii Int. Conf. Systems Sciences, Kohala Coast, HI, Jan. ,pp. 74-83 (1998).
[8] G. D. Caro, M. Dorigo: An adaptive multi-agent routing algorithm inspired by ants behavior, In Proc. 5th Annual Australasian Conf. Parallel Real-Time Systems 261-272 (1998).
[9] N. Varela, M. C. Sinclair: Ant colony optimization for virtual-wavelength- path routing and wavelength allocation, In Proc. Congress Evolutionary Computation, Washington, DC, pp. 1809-1816 (1999).
[10] S. Fenet, S. Hassas: An ant based system for dynamic multiple criteria balancing, In Proc. 1st Int.Workshop Ants Systems, Brussels, Belgium (1998).
[11] K. Oida, M. Sekido: ARS: an efficient agent-based routing system for QoS guarantees, Computer Communications. 23, 1437–1447 (2000).
[12] K. M. Sim, W. H. Sun: Multiple Ant Colony Optimization for Load Balancing, In Proc. 4th Int. Conf. Intelligent Data Engineering Automated Learning, Hong Kong, vol. 2690, pp. 467-471 Springer,Heidelberg (2003).
[13] M. Dorigo, G. D. Caro: Ant Algorithms for Discrete Optimization. Artificial Life, MIT Press 137-172 (1999)
[14] T. Stutzle, H. H. Hoos: MAX-MIN Ant System. Future Generation Computer System, 889-914 (2000).