Malaysian Journal of Computer, Vol. 17 No. 2, December 2004, pp. 21-29

AN ADAPTIVE ROUTING ALGORITHM: ENHANCED CONFIDENCE-BASED Q ROUTING ALGORITHM IN NETWORK TRAFFIC

Soon Teck Yap and Mohamed Othman
Department of Communication Technology & Network
Universiti Putra Malaysia
43400 UPM Serdang, Selangor D.E.
email: styap98@hotmail.com
mothman@fsktm.upm.edu.my

ABSTRACT

Confidence-based Q (CQ) Routing Algorithm is an adaptive network routing algorithm. CQ Routing Algorithm evaluates how confidence value (C value) can be used to improve the quality of exploration in Q Routing Algorithm for adaptive packet routing in communication networks. However, the C value incompletely evaluates how closely the Q value represents the current condition of the network in a given length of time, which is measured in term of estimated delivery time for a packet to arrive at its destination. In this paper, an Enhanced Confidence-based Q (ECQ) Routing Algorithm is proposed. The Variable of Decay Constant and Update All Q value approaches are introduced for updating the C values of non-selected Q values. Using these C values would make those non-selected Q values more competitive in order to achieve updated and more reliable values. The quality of exploration in CQ Routing Algorithm would be improved. The performance of ECQ and CQ Routing Algorithms are compared to prove this improvement. ECQ and CQ Routing Algorithms are tested on an irregular 6 x 6 nodes network grid.

Keywords: ***Adaptive routing algorithm, Confidence-based Q Routing, Confidence value***

1.0 INTRODUCTION

A computer network can be defined as a single computer, called a host, with communication circuits, communication equipments, and terminals or two or more computers connected via a communication medium, together with associated communication links, terminals and communication equipments [3]. A computer network can be a simple computerised communication network which links a few hosts to a more complex form of computerised communication network. The information is encapsulated in a standard format and size then transferred from one computer to another as data packets. The process of sending a packet P (*s*, *d*), from its source node *s* to its destination node *d* is referred to as packet routing or network routing [1]. The process of sending a packet usually takes multiple "hops" to transfer the packet from its source to destination node. On its way, the packet spends time queuing to be served at the intermediate nodes and travelling in the transmission medium. Thus the delivery time of a packet, is defined as the total time it takes for a packet to travel from the source node to reach its destination. The delivery time depends on the total time a packet spends between the source node and destined node that includes time spent in the queues of the intermediate nodes and time spent in the transmission medium [8,10].

Normally, a packet has multiple possible routes to reach its destination. The decision of selecting a better route among the multiple available choices is crucial so that the packet reaches its destination early by spending minimum time in the process of packet routing. This is not a simple task. The packet routing has three challenges [4]. First, routing requires information of the coordination between all nodes in a network. Second, the routing system must cope with link and node failures, requiring redirection of traffic. The latest information of network traffic is updated in databases that are maintained by the system. Third, to achieve high performance, the routing algorithm may need to change its routes when some areas within the network become congested or offline. Because the network environment changes from time to time, a routing algorithm used by a routing decision maker to make a routing policy should be able to adjust the routing policy from time to time according to the state of the network.

2.0 CONFIDENCE-BASED Q ROUTING

CQ Routing Algorithm is an adaptive routing algorithm that adapts well by itself by changing its routing policy according to the changes of condition in the network such as changes of network traffic pattern, network traffic load level and network topology. For adaptive routing algorithm, each node maintains its own view of the network. The creation of routing policy in a node is based on local information of the current state of the node itself and the nodes adjacent to it. To make an ideal routing policy, the routing policy should be made by a central controller that monitors the whole network. The central controller makes an effective routing policy to route packets based on the information gathered from all nodes in the network.

But an adaptive routing algorithm does not have a central controller that fully gains information about the status of all nodes and links in the network since this information is impossible to obtain in a large and dynamic network like the Internet. Therefore, without a central controller to monitor nodes in the network, each node makes its own local routing policy and these local routing policies influence one another. As a result, a dynamic and optimised global routing policy is achieved in the network. Unlike the non-adaptive routing algorithm such as the shortest path algorithm, adaptive routing algorithm adapts well by itself as compared to non-adaptive routing algorithm in various conditions of the network.

The CQ Routing Algorithm improves the quality of route exploration by making an effort to evaluate the reliability of the sole parameter – Q value, in the Q Routing Algorithm to represent the actual state of the network. In CQ Routing Algorithm, a node has first to learn a representation of the state of the network in terms of Q values and these values are used to make control decision. The Q value Q_x (y, d) in node x represents the estimated delivery time for a packet P (s, d), if sent through neighbouring node y to reach its destination node d. A minimum Q_x (y, d) value implies that the estimated delivery time in node x through its neighbouring node y is shorter than other neighbouring nodes and vice versa. So, when the decision maker tries to send a packet through this node (which has minimum Q value), the packet is expected to arrive earlier at its destination.

In the CQ Routing Algorithm, a new parameter called Confidence value (C value) is associated with Q value [11, 12, 13]. In Q Routing Algorithm, it is not known how good this Q value reflects the actual state of the network. Whether or not the Q value has been updated recently is not determined. The purpose of this C value is to represent the reliability of the Q value. If the Q value has just been recently updated then it is a reliable Q value and we are highly confident that this Q value closely reflects the actual state of the network. On the other hand, if the Q value has not been updated for quite some time then it is an unreliable Q value and we have lower or no confidence that this Q value reflects the actual state of the network. The C value of the Q value can be quantified numerically. The C value ranges from zero to one. A C value of one interprets that the associated Q value is highly reliable because it is just recently updated. This highly reliable Q value will be able to reflect the actual state of the network. On the other hand, a C value of nearly zero means that the associated Q value is highly unreliable because it has not been updated for quite some time.

The learning rate is defined as a function of these C values [12]. The C values of Q values that are part of the updates are also used to determine the learning rate [11, 12, 13] in Confidence-based Q Routing Algorithm rather then using an experimental constant learning rate. When a Q value with a low C value is to be updated, it is advisable to keep updating its value. This means that the learning rate should be high. In another situation, if the estimated C value received is high then the learning rate for this Q value should be high too [11, 12]. These show the relationship between the C value and learning rate. The learning rate depends on the current C value and the estimated C value received. Since the CQ Routing Algorithm depends on the reliability of the Q value to adjust its learning rate and update the Q value, it will increase the quality of exploration when more exploration activities occur. The learning rate function, η_f (C^{old}, C^{est}) is chosen based on the following rules [11, 12]. Learning rate should be high if either:

1. Confidence value in the old Q value, C^{old} is low or
2. Confidence value in the new estimate Q value, C^{est} is high.

Since the CQ Routing Algorithm depends on the reliability of the Q value to adjust its learning rate and update the Q value, it will increase the quality of exploration when more exploration activities occur. A simple form of the learning rate function is given as below:

$$\eta_f (C^{old}, C^{est}) = \max (1 - C^{old}, C^{est}); C^{old}, C^{est} \in [0,1] \quad (2.1)$$

Node x sends a packet P (s, d) which is from node s, if sent through neighbouring node y to reach its destination node d. The update function for the selected Q value, which is the minimum Q value includes the update of C and Q values respectively. Node x updates its selected Q value, $Q_x(y, d)$ as follows:

$$\begin{aligned} Q_x(y,d)^{\text{update}} &= Q_x(y,d)^{\text{old}} + \Delta Q; \\ Q_x(y,d)^{\text{update}} &= Q_x(y,d)^{\text{old}} + \eta_f(C^{\text{old}}, C^{\text{est}})\{Q_x(y,d)^{\text{est}} - Q_x(y,d)^{\text{old}}\}; \\ Q_x(y,d)^{\text{update}} &= Q_x(y,d)^{\text{old}} + \max(C^{\text{est}}, 1 - C^{\text{old}}) \\ &\quad \{Q_x(y,d)^{\text{est}} - Q_x(y,d)^{\text{old}}\}; \\ Q_x(y,d)^{\text{update}} &= Q_x(y,d)^{\text{old}} + \max\{C_y(z,d), 1 - C_x(y,d)\} \\ &\quad \{Q_y(z,d) + q_y + \delta - Q_x(y,d)^{\text{old}}\}; \end{aligned} \tag{2.2}$$

By using Equation 2.1, node x updates its C value, $C_x(y, d)$ of the selected Q value as shown in the following equation:

$$\begin{aligned} C_x(y,d)^{\text{update}} &= C_x(y,d)^{\text{old}} + \Delta C; \\ C_x(y,d)^{\text{update}} &= C_x(y,d)^{\text{old}} + \eta_f(C^{\text{est}}, 1 - C^{\text{old}})\{C_y(z,d) - C_x(y,d)^{\text{old}}\}; \\ C_x(y,d)^{\text{update}} &= C_x(y,d)^{\text{old}} + \max\{C_y(z,d), 1 - C_x(y,d)^{\text{old}}\} \\ &\quad \{C_y(z,d) - C_x(y,d)^{\text{old}}\}; \end{aligned} \tag{2.3}$$

The updating function for the non-selected Q value involves the update of Č value of non-selected Q value only. Using a pre-defined decay constant λ, node x updates its $\check{C}_x(y, d)$ as follows:

$$\check{C}_x(y,d)^{\text{update}} = \lambda \cdot \check{C}_x(y,d)^{\text{old}}; \lambda \in (0,1) \tag{2.4}$$

3.0 ENHANCED CONFIDENCE-BASED Q ROUTING ALGORITHM

In CQ Routing Algorithm, the updating rule for a non-selected Q value is updating its C value with a decay constant value – λ. This decay constant value shows how the C value decreases from time to time while the Q value is not selected. Using a single decay constant cannot satisfy a multi-connection environment. This is because of the need to adjust the learning rate in a different multi-connection environment.

3.1 Variable of Decay Constant Approach for Non-Selected Node Connections

Consider the following scenario. A node with four connections to its adjacent nodes, called node-A and a node with two connections to its adjacent nodes, called node-B as illustrated in Fig. 1a and 1b respectively. Node-A has three non-selected Q values (excluding the current selected Q value) while the node B has only one non-selected Q value.

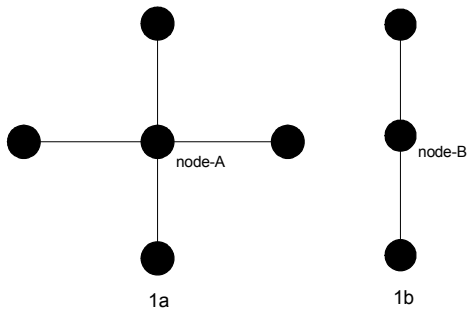

Fig. 1: Above figure shows node-A and node-B with four node connections and two node connections respectively

If the current selection rule changes (current selected Q value is higher than the others) then the decision maker selects a new minimum Q value. Node-A has three candidates for selection while node-B has only one candidate to be selected. Surely for node-B, the non-selected Q value will be selected while node-A will pick the minimum Q value among the non-selected Q values. Node-B may not need to have a high learning rate as compared to node-A since node-B is updating its Q values more frequently compared to node-A. In other words, node-A has to have a higher learning rate according to the first rule of learning rate function because it has lower confidence since it has longer waiting time to update all its Q values.

From the above scenario, we learn that the selected policy will change while the current Q value is higher than other Q values. This is a greedy approach where it tries to exploit the current Q value as much as it can. This will

decrease the chances for the decision maker to quickly detect the swift of change of state in the network because it spends less time exploring other better possibilities.

A simple solution for the above scenario is by introducing the Variable of Decay Constant Approach. In the case of node-A, it has many candidates competing to be selected as compared to node-B. Node-A should maintain a higher learning rate to keep it learning more effective routing policies. In other words, to maintain a higher learning rate, the non-selected connection should update itself with a faster decay constant when it is selected. Thus, the value of the decay constant should decay faster. The decay constant can be associated to the number of non-selected node connections to determine the value of the decay constant. The value of the decay constant should be normalised. This is because for every packet that leaves a node, the fraction of the Q values updated is small, so the decay constant should be normalised to reflect this outcome.

In the following update function, let's assume that node-A sends a packet P (s, d) from the source node s to its destination node d via its neighbouring node y. It has three non-selected connections so the decay constant will be:

$$\lambda^{\{1/(n-1)\}};\ n \geq 2 \tag{3.1}$$

where n is number of node connections.

From Equation 2.1, the learning rate function $\eta_f (C^{old}, C^{est})$ for selected Q value (selected connection) is:

$$\eta_f (C^{old}, C^{est}) = \max \{1 - C_A (y, d)^{old}, C_y (z, d)^{est}\};\ z \in N (y) \tag{3.2}$$

where $C_A (y, d)^{old}$ is the current C value and $C_y (z, d)^{est}$ is the estimated C value received from neighbouring node y where $z \in N (y)$.

The update rule for selected Q value is the same as CQ Routing Algorithm, where ΔQ is the difference between current Q value and the new estimated Q value. The $Q_A (y, d)^{old}$ is the current Q value for node-A, $Q_y (z, d)^{est}$ is the minimum Q value received from neighbouring node y where $z \in N (y)$, q_y is queue size of node y and δ is the transmission delay between node-A and node y.

$$\begin{aligned} Q_A (y, d)^{update} &= Q_A (y, d)^{old} + \Delta Q; \\ Q_A (y, d)^{update} &= Q_A (y, d)^{old} + \eta_f (C^{old}, C^{est}) \text{ (new estimated Q value – current Q value);} \\ Q_A (y, d)^{update} &= Q_A (y, d)^{old} + \max \{1 - C_A (y, d)^{old}\}, C_y (z, d)^{est}\} \\ &\quad \{Q_y (z, d)^{est} + q_y + \delta - Q_A (y, d)^{old}\}; \end{aligned} \tag{3.3}$$

The update function for the selected C value is the same as the CQ Routing Algorithm, as follows:

$$\begin{aligned} C_A (y, d)^{update} &= C_A (y, d)^{old} + \Delta C; \\ C_A (y, d)^{update} &= C_A (y, d)^{old} + \max \{1 - C_A (y, d)^{old}, C_y (z, d)^{est}\} \\ &\quad \{C_A (y, d)^{old} - C_y (z, d)^{est}\}; \end{aligned} \tag{3.4}$$

where $C_A (y, d)^{old}$ is the current C value and $C_y (z, d)^{est}$ is the estimated C value received from neighbouring node y where $z \in N (y)$.

The update rule for non-selected C value, a substitute of the decay constant λ with Equation 3.1 is:

$$C_A (\hat{y}, d)^{update} = \lambda^{\{1/(n-1)\}} \cdot C_A (\hat{y}, d)^{old}; \tag{3.5}$$

where $n \geq 2$, $\hat{y} \in N (A)$ and $\hat{y}$ is non-selected neighbouring node.

3.2 Update All Q Value Approach

Those non-selected Q values do not reflect the actual state of the network as the values are derived in the past, after which they passively wait for the chance to be selected and get updated. To improve the reliability of these Q values, certain matters have to be considered. Firstly, we have to define what is an unreliable Q value. The reliability of the Q value is determined by the C value. A C value of one indicates that the Q value is completely reliable since it has just been updated. A value of zero for C value states that the Q value is completely unreliable since it has not been updated for quite some time. In a safety measure, a value of 0.5 could be the margin between these C values and a tolerable measuring value of reliability.

But the C value cannot be used directly in the CQ Routing Algorithm to change the routing policy. This is because the sole measurement of value in CQ Routing Algorithm that changes the routing policy is the Q value. The C value could indirectly participate in determining and changing the routing policy. If the reliability of the Q value is low, it is advisable to not just update it with a faster decay constant but also more frequently before it becomes more unreliable. This could be executed immediately when a decision maker makes a decision by updating all Q values including those Q values that are not selected.

The C value represents the reliability of the Q value. If the C value equals to one, it has two possibilities: it has previously been updated or the packet will reach its destination in the next step (Q value = δ, C value = 1) [12]. When the C value equals to one, we assume that this numeric Q value reflects 100% of the actual state of the network. When the C value reduces, for instance to 0.95, we can assume that the Q value reflects approximately 95% of the actual state of the network. So, the Q value is about ± 5% disparate from the actual state. From the above assumptions, a C value of 0.95 for estimated Q value therefore reflects that the actual state is ±5% from the current Q value. It is better to test a value 5% lower instead of 5% higher. The additional 5% will make the Q value larger and less likely to get updated. It could be argued that when the C value equals to zero, then the Q value will be zero. This does not really reflect the actual situation. In fact, this is true but it is a positive errand number for the Q value. Such is the case because the decision maker will choose this errand number in a shorter time and the error will be corrected (at the same time updating this random Q value). Using this approach, the non-selected Q values are more competitive for selection and more exploration will occur. These non-selected Q values will be updated as follows:

$$Q_A(\hat{y}, d)^{\text{update}} = Q_A(\hat{y}, d)^{\text{old}} - \{(1 - C_A(\hat{y}, d)^{\text{update}})\, Q_A(\hat{y}, d)^{\text{old}}\} \quad (3.6)$$

where $\hat{y} \in N(A)$ and $\hat{y}$ is non-selected neighbouring node

4.0 SIMULATION DESIGN AND SETUP

The network used for experiment is an irregular 6 x 6 nodes network grid [8, 11, 12, 13]. The network is basically divided into two parts (clusters), left cluster from node-1 to node-18 and right cluster from node-19 to node-36. Both clusters are connected through route R1 and R2 as shown in Fig. 2. The number of connections a node has to its neighbouring nodes varies among each node in the network. Basically, each node is connected by at least two bi-directional links to its neighbouring nodes and at most four bi-directional links to its neighbouring nodes.

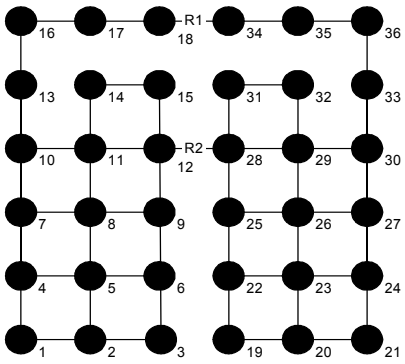

Fig. 2: Irregular 6 x 6 nodes network. Left cluster (node-1 to node-18) are connected to right cluster (node-19 to node-36) through route R1 (node18 & 34) and route R2 (node12 & 28)

In each **simulation time step** (sts), packets are randomly generated at a random node with a random destination. These packets are stored in the queue of the source nodes. The amount of packets injected into the network depends on the value of the network traffic load. Below is the formula for calculating the network traffic load:

$$\text{Network traffic load} = n \bullet Nnodes; \quad (4.1)$$

where n is the average number of packets each node will generate, and $Nnodes$ is the total number of nodes in the network.

In the simulation setup, the maximum number of packets allowed in the network, Max_{pkt} is the summation of the half queue size for all nodes in the network and all nodes are identical. The Max_{pkt} is a suitable experimental value to simulate high load without blockage on route R1 and R2. The equation for Max_{pkt} is shown as follows:

$$\text{Max}_{\text{pkt}} = 0.5 \bullet Nnodes \bullet MQ_{\text{size}}; \quad (4.2)$$

where MQ_{size} is the size of the queue in a node.

The decay constant λ for both CQ Routing Algorithm and ECQ Routing Algorithm is set to 0.95 [12]. The λ for ECQ Routing Algorithm also depends on the number of connections a node has. All initial values for C and Q are set to zero. The waiting time in the queue is set to one unit of time and the travelling time between two nodes is set to eight units of time. The assumption is that the travelling time is usually longer than the waiting time in a node. These two experimental values are showing the relative time required to complete the action. For example, in one

sts, each unit of time a packet is waiting in the queue in one unit of time, the value of one is added to the total delivery time of the packet, and for each transmission going through the communication link eight units of time is added to the total delivery time of the packet.

In one sts, each node sends a packet to its neighbouring node if its queue is not empty. Then, all packets arrive in its queue and the first packet in the queue is processed first by the node. Next, it sends a feedback to the sender node. If this packet is not destined to this node, then the packet is sent to its best neighbouring node else it is removed from the network. Wait and update C_y (z, d) value, q_y and Q_y (z, d) value in the feedback from the neighbouring node y and thus updating its Q_x (y, d) value. For non-selected Q values, the CQ Routing Algorithm updates its C values and ECQ Routing Algorithm updates both its C and Q values.

Results from the experiments are collected at a constant simulation time interval (sti) where one sti consists of 20 sts and each experiment runs for 5,000 sts. Two values are collected from the experiment, Average Packet Delivery Time (APDT) and Average Number of Packets Delivered (ANPD). These values are collected from 50 sets of experiments.

5.0 EXPERIMENTAL RESULTS

In the experiments, the network traffic load and network topology are constant throughout the experiment. The value n of network traffic load is set to 0.4 for simulation of high network traffic load and 0.1 for simulation of low network traffic load. The CQ Routing Algorithm is comparable to ECQ Routing Algorithm. The graph patterns from both algorithms are found relatively similar since they share most of the common characteristics.

At low network traffic load, the average packet delivery times (Fig. 3) shows that ECQ Routing Algorithm performs far better than CQ Routing while the average number of packet delivered (Fig. 4) is slightly lower than CQ Routing Algorithm. The maximum APDT value recorded for ECQ Routing Algorithm is 748 while for CQ Routing Algorithm is 2,012. The ANPD value for ECQ Routing Algorithm is 0.9 packets while for CQ Routing Algorithm is 1.0 packet.

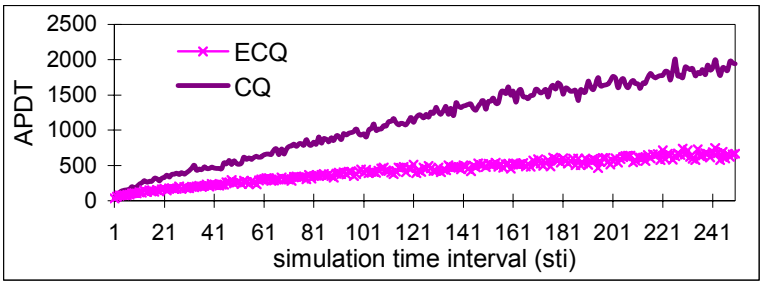

Fig. 3: Average packet delivery time at a low load. Comparison of average packet delivery time in 5000 simulation time steps for ECQ Routing and CQ Routing

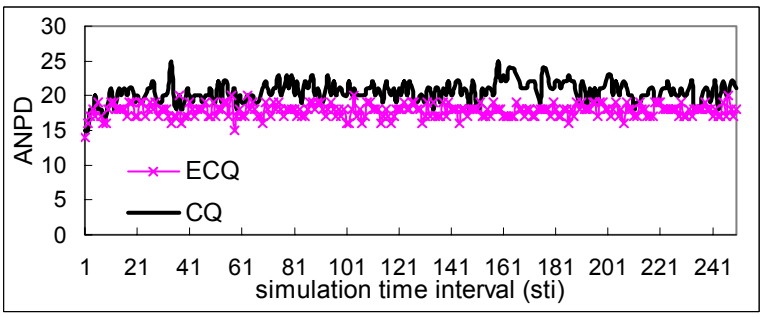

Fig. 4: Average number of packet delivered at a low load. Comparison of average number of packet delivered in 5000 simulation time steps for ECQ Routing and CQ Routing

At high network traffic load, the average packet delivery time (Fig. 5) shows that ECQ Routing Algorithm perform slightly better than CQ Routing Algorithm while the performance of the average number of packet delivered (Fig. 6) is almost the same level with CQ Routing Algorithm. The maximum APDT value recorded for ECQ Routing Algorithm is 2,171 while for CQ Routing Algorithm is 2,444. The ANPD value for ECQ Routing Algorithm is 1.4 packets while for CQ Routing Algorithm are 1.3 packets.

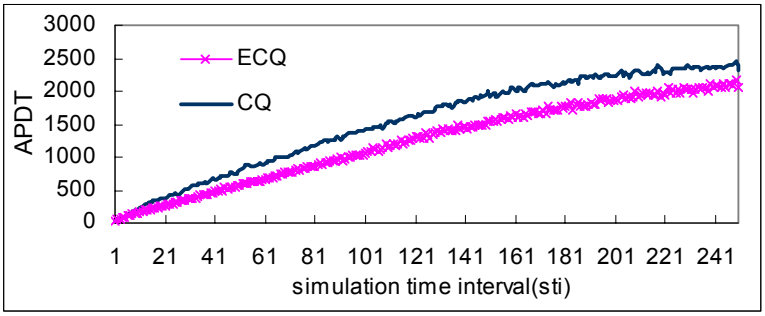

Fig. 5: Average packet delivery time at a high load. Comparison of average packet delivery time in 5000 simulation time steps for ECQ Routing and CQ Routing

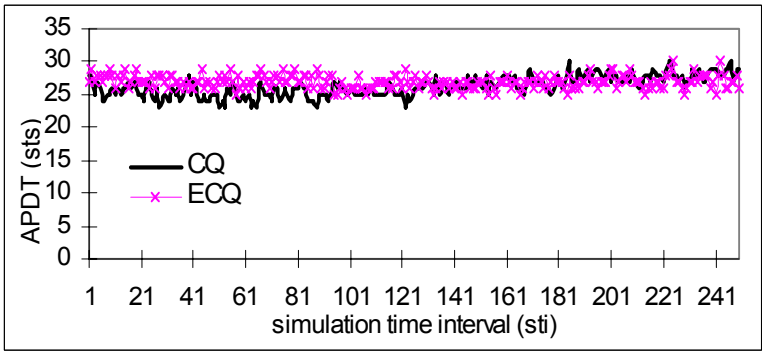

Fig. 6: Average number of packet delivered at a high load. Comparison of average number of packet delivered in 5000 simulation time steps for ECQ Routing and CQ Routing

Overall, the ECQ Routing Algorithm outperforms the CQ Routing Algorithm. The reason both routing algorithms have increased APDT is that the number of packets injected into the network is far higher than the packets removed from the network and this is more noticeable in high network traffic load. Although in each simulation time step, the maximum possible number of removed packets is equivalent to the number of nodes in the network, *Nnodes* (36 nodes) but the actual number of packets removed is far lower then the average packet injected into the simulation.

6.0 CONCLUSION

ECQ Routing Algorithm evidently has a better adaptation ability compared to the CQ Routing Algorithm in these experiments. The ECQ Routing Algorithm has a better adaptation ability because of its lower APDT and at the same time it has a higher number of ANPD compared to the CQ Routing Algorithm. Although at low network traffic load, ECQ Routing Algorithm has a slightly lower ANPD compared to CQ Routing Algorithm but ECQ Routing Algorithm is still better than CQ Routing Algorithm since it lowered its APDT much more than the CQ Routing Algorithm. A conclusion from the results of the experiments is, ECQ Routing Algorithm is better then CQ Routing Algorithm in terms of the quality of exploration. The introduction of Variable of Decay Constant and Update All Q Value approaches enables multi-connection nodes to continuously learn in changing environments. As they learn more about their environment, they can adapt better to that environment.

7.0 DISCUSSION AND FUTURE WORK

Future researches should focus on the dynamic network environments. The observation of the changes in network properties in the system will let us have a better understanding about the efficiency of a particular routing algorithm in adapting to its environment. The properties of the dynamic environment can include the on-line and off-line linkage between two nodes, different lengths and types of transmission mediums, different network topologies and different processing power of the nodes.

For the stationary network environment, certain modifications of the network properties are worth investigating. In the present study, the value of the network traffic load (n) is set to 0.4. This value can be changed to simulate different sets of network traffic load levels for future experiments. The value $n = 0.4$ can be considered as a high network traffic load since the simulator can inject an average of 14.4 packets to the system while the packet removal from the system is averaged at less than 2 packets. To compare different performances in different network traffic load levels such as low, medium and very high levels of network traffic load, the n can be set to 0.1, 0.2 and 1.0 respectively.

An enhancement to the Update All Q Value approach is by applying the Memory-based Reinforcement Learning approach from Predictive Q Routing [2]. The decay constant is replaced by a memory-based variable that keeps the best experiences learned. It is used to predict the network traffic patterns and update those non-selected Q values. By doing so, these updated non-selected Q values are more precisely reflected to the actual condition of the network.

Adaptive routing methods are widely studied by many researchers all around the world. There are other routing researches focusing on total latency such as Selfish Routing [9] and Self Adaptive Routing [6]. There are other studies on different performance metrics as well such as delay, error rate, financial cost, etc. An example of these studies is adaptive QoS Routing [5]. Each required Type of Services (ToS) could be associated with a particular metric according to which IP traffic can be differentially routed for a guaranteed QoS. A further study in adaptive QoS Routing is by using prediction methods in order to obtain a guaranteed QoS in ad hoc networks called link performance prediction strategy [7].

REFERENCES

[1] R. E. Bellman, "On a Routing Problem". *Quarterly of Applied Mathematics*, Vol. 16, 1958, pp. 87-90.

[2] Choi, S. P. M. and Yeung, D. Y., "Predictive Q Routing: A Memory-Based Reinforcement Learning Approach to Adaptive Traffic Control". *Advances in Neural Information Processing Systems*, Vol. 8, 1996, pp. 945-951.

[3] David A. Stamper, *Business Data Communications*. Forth edition, CA, The Benjamin/Cummings Publishing Co. Inc., 1994.

[4] Dimitri Bertsekas, Robert Gallager, *Data Networks*. Second edition, NJ, Prentice Hall, 1992.

[5] Dominique Snyers, Martin Heusse and Philip Buckle, "Adaptive Agent-Driven Routing and QoS in Communication Networks". *Nortel internal Report,* 1999.

[6] Haiyong Xie, Lili Qiu, Yang Richard Yang and Yin Zhang, "On Self Adaptive Routing in Dynamic Environments – An Evaluation and Design Using a Simple, Probabilistic Scheme", in *Proceedings of 12th International Conference on Network Protocols (ICNP 2004)*, 2004, pp. 12–23.

[7] HongXia Sun and Herman D. Hughes, "Adaptive QoS Routing Based on Prediction of Local Performance in Ad Hoc Networks". *WCNC 2003 - IEEE Wireless Communications and Networking Conference*. Vol. 4, no. 1, 2003, pp. 1191-1195.

[8] Justin A. Boyan, and Michael L. Littman. "Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach". A*dvances in Neural Information Processing Systems*, Vol. 6, 1993, pp. 671-678.

[9] Lili Qiu, Yang Richard Yang, Yin Zhang and Scott Shenker, "On Selfish Routing in Internet-Like Environments", in *Proc. ACM SIGCOMM 2003*, 2003, pp. 151–162.

[10] Michael Littman and Justin Boyan, "A Distributed Reinforcement Learning Scheme for Network Routing", in *Proceeding of International Workshop on Applications of Neural Networks to Telecommunications*, 1993, pp. 45-51.

[11] Shailesh Kumar, and Risto Miikkulainen, "Confidence-Based Q-Routing: An On-Line Adaptive Network Routing Algorithm", in *Smart Engineering Systems: Neural Networks, Fuzzy Logic, Data Mining, and Evolutionary Programming*, Vol. 8, 1998, pp. 147-152.

[12] Shailesh Kumar, "Confidence-based Dual Reinforcement Q-Routing: An On-line Adaptive Network Routing Algorithm". *Tech. Report AI98-267, Master's Thesis*, The University of Texas at Austin, Austin, US, 1998.

[13] Shailesh Kumar and Risto Miikkulainen, "Confidence Based Dual Reinforcement Q-Routing: An Adaptive On-Line Routing Algorithm". *Sixteenth International Joint Conference on Artificial Intelligence*, 1999, pp. 231–238.

BIOGRAPHY

Soon Teck Yap was born in Kuala Lumpur, Malaysia on 13th October 1976. He acquired his primary and secondary school education in Kuala Lumpur. He obtained his Bachelor of Computer Science from Universiti Putra Malaysia, Serdang, Malaysia, in May 2000.

Mohamed Othman is Head of Department of Communication Technology and Network, University Putra Malaysia. He received his PhD from the National University of Malaysia with distinction (Best PhD Thesis in 2000 awarded by Sime Darby Malaysia and Malaysian Mathematical Society). In 2002 and 2003, he received gold medal award for Research and development Exhibition. His main research interests are in the fields of parallel and distributed algorithms, high speed computer network, Network Management (security and traffic monitoring) and scientific computing. He already published more than sixty journal papers. He is also an associate researcher and coordinator of High Speed Machine at Institute of Mathematical Science (INSPEM), Universiti Putra Malaysia.