Proactive QoS Routing in Ad-Hoc Networks

Ying Ge[1], Thomas Kunz[2], Louise Lamont[1]

[1] Communications Research Center, 3701 Carling Ave, Ottawa, ON K2H 8S2
{ying.ge, louise.lamont@crc.ca}
[2] Dept. of Systems and Computer Engineering, Carleton University, Ottawa, ON K1S 5B6
tkunz@sce.carleton.ca

Abstract. In this paper, we analyze the advantages and disadvantages of the proactive QoS routing in ad-hoc networks. We discuss how to support bandwidth QoS routing in OLSR (Optimized Link State Protocol), a best-effort proactive MANET routing protocol. Using OPNET, we simulate the algorithm, exploring both traditional routing protocol performance metrics and QoS-specific metrics. Our analysis of the simulation results shows that the additional message overhead generated by the proactive QoS routing have a negative impact on the performance of the routing protocol. Given the negative results, we identified research areas that would be worthwhile investigating in order to obtain better performance results.

1 Introduction

QoS routing in Ad-Hoc network is difficult. To support QoS routing, the link state metrics such as delay, bandwidth, jitter, loss rate and error rate in the network should be available and manageable. However, getting and managing such link state information in a MANET is not trivial because the quality of a wireless link changes quite frequently due to mobility and variations in the surroundings. In addition, it is also complex to evaluate the QoS routing performance. Compared to the traditional best-effort routing, QoS routing has two additional overheads – "computational cost" and "protocol overhead" [2]. "Computational cost" comes from the more frequent path selection computations, since besides maintaining the source-destination connection, additional computations are needed to determine paths that satisfy the QoS demands. The additional "protocol overhead" comes from the need to distribute the frequently updated link state information. There is a trade-off between the QoS performance the QoS routing protocol achieves and the additional cost it introduces.

In on-demand QoS protocols such as [3] and [11], a route is found based on specific QoS requirements. However, the unpredictable nature of Ad-Hoc networks and the requirement of quick reaction to QoS demands make the idea of a proactive protocol more suitable. When a request arrives, the control layer can easily check if the pre-computed optimal route can satisfy such a request. Thus, waste of network resources when attempting to discover infeasible routes is avoided. These advantages of the proactive QoS routing motivate us to look into this area. However, similar to a proactive best-effort routing protocol, a proactive QoS routing may introduce "protocol" overhead. Do these additional overhead have a negative effect on the Ad-

Hoc network? If yes, then how much additional overhead does a proactive QoS routing protocol introduce into the network? How does the additional overhead affect the performance of the routing protocol? Can we minimize the costs to achieve better performance? Or should we just give up on proactive QoS routing? The goal of this paper is to investigate the answers to these questions through the performance evaluation of a proactive bandwidth QoS routing algorithm that we have proposed.

In [5], we studied the approach of proactive QoS routing and proposed 3 heuristics that allow OLSR (Optimized Link State Protocol [8]) to pre-compute the best bandwidth route among all the possible routes. That work presents the performance of the heuristics in a static network. In this paper, we implement one QoS OLSR heuristic, which guarantees to find the best bandwidth path in the static network and has comparably low overhead, in OPNET and evaluate the routing algorithm's performance with node movements and data flows, and consequently, analyze the feasibility of proactive routing in MANET.

The rest of the paper is organized as follows: a brief description of OLSR and QoS versions of OLSR is given in Section 2. The detailed implementation of QoS OLSR in OPNET is discussed in Section 3. Section 4 lists the OPNET simulation parameters and discusses the simulation results in OPNET. Section 5 analyses whether proactive QoS routing is practical in an Ad-Hoc network and discusses future work.

2 OLSR and QoS OLSR

The IETF's MANET Working Group has introduced the Optimized Link State Routing (OLSR) protocol for mobile Ad-Hoc networks [8]. The protocol is an optimization of the pure link state algorithm. The key concept used in the protocol is that of multipoint relays (MPRs). The MPR set is selected such that it covers all nodes that are two hops away. A node's knowledge about its neighbors and two-hop neighbors is obtained from HELLO messages – the message each node periodically generates to declare the nodes that it hears. The node N, which is selected as a multipoint relay by its neighbors, periodically generates TC (Topology Control) messages, announcing the information about who has selected it as an MPR. Apart from generating TCs periodically, an MPR node can also originate a TC message as soon as it detects a topology change in the network. A TC message is received and processed by all the neighbors of N, but only the neighbors who are in N's MPR set retransmit it. Using this mechanism, all nodes are informed of a subset of all links – links between the MPR and MPR selectors in the network. So, contrary to the classic link state algorithm, instead of all links, only small subsets of links are declared. For route calculation, each node calculates its routing table using a "shortest hop path algorithm" based on the partial network topology it learned. MPR selection is the key point in OLSR. The smaller the MPR set is, the less overhead the protocol introduces. The proposed heuristic in [8] for MPR selection is to iteratively select a 1-hop neighbor that reaches the maximum number of uncovered 2-hop neighbors as an MPR. If there is a tie, the one with higher degree (more neighbors) is chosen.

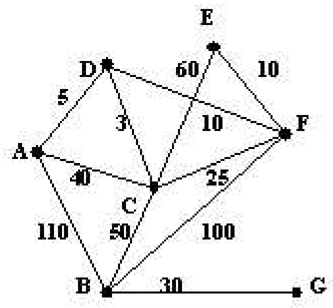


Fig. 1. Simple network. An edge between two nodes indicates that the two nodes connected by this edge are within reach of each other. The edge weight represents the QoS link attribute we are interested in, available bandwidth.

Table 1. Node B's MPR(s), based on Fig. 1

Node	1 Hop Neighbors	2 Hop Neighbors	MPR(s)
B	A, C, F, G	D, E	C

From the perspective of node B, both C and F cover all of node B's 2-hop neighbors. However, C is selected as B's MPR as it has 5 neighbors while F only has 4 (C's degree is higher than F).

OLSR is a routing protocol for best-effort traffic, with emphasis on how to reduce the overhead. So in its MPR selection, the node selects the neighbor that covers the most unreachable 2-hop neighbors as MPR. However, in QoS routing, by such MPR selection mechanism, the "good quality" links may be "hidden" to other nodes in the network. As an example, we will consider the network topology in Fig. 1. In the OLSR MPR selection algorithm, node B will select C as its MPR. So for all the other nodes, they only know that they can reach B via C. Obviously, when D is building its routing table, for destination B, it will select the route D-C-B, whose bottleneck bandwidth is 3, the worst among all the possible routes. Also, when "bandwidth" is the QoS constraint, nodes can no longer use the "shortest hops path" algorithm as proposed in OLSR. Because of these limitations of OLSR in QoS routing, the QoS OLSR version revises it in two aspects: MPR selection and routing table computation.

The decision on how each node selects its MPRs is essential to determining the optimal bandwidth route in the network. In selecting the MPRs, a "good bandwidth" link should not be omitted. Based on this idea, we previously explored three revised MPR selection algorithms [5]. In this paper, we implement the best variant (OLSR_R2) in OPNET to compare its performance with the original OLSR protocol. The idea behind OLSR_R2 is to select the best bandwidth neighbors as MPRs until all the 2-hop neighbors are covered.

Table 2. Node B's MPR(s), using OLSR_R2

Node	1 Hop Neighbors	2 Hop Neighbors	MPR(s)
B	A, C, F, G	D, E	A, F

Among node B’s neighbors, A, C, and F have a connection to its 2-hop neighbors. Among them, link BA has the highest bandwidth. So A is first selected as B’s MPR, and the 2-hop neighbor D is covered. Similarly, F is selected as MPR next and E is covered, so all 2-hop neighbors are covered and the algorithm terminates. This revised OLSR MPR selection algorithm improves the chance that a better bandwidth route is found. However, by using such algorithm, the overhead also increases because the number of MPRs in the network is increased.

Besides the MPR selection method, a node also needs to change the “shortest hops path” algorithm in its routing table computation so as to find the best bandwidth route. We use the “Extended BF” algorithm [6], which computes the best bandwidth paths from a source to any reachable destinations with minimum hop count (shortest-widest path).

With bandwidth constraint as QoS metric, it is reasonable to view the “bandwidth” as available bandwidth. Most probably, the devices in the Ad-Hoc network will be configured with the same wireless card, which means that all nodes in the network have the same maximum bandwidth. So we are only interested in how much of the remaining bandwidth is available for new traffic. However, in real networks, bandwidth computation is a complex issue. Many papers such as [9] discuss how to compute bandwidth in Ad-Hoc networks. Here, we use a rather simple and straightforward approach: measuring how much time a node monitors an idle channel and thus is available to transmit new messages over a link (node’s idle time), which is similar to the approach suggested in [1].

3 QoS OLSR Implementation in OPNET

The Naval Research Laboratory (NRL) of the United States Department of Defense developed the original OLSR model in OPNET. To implement the QoS versions of the OLSR protocol, besides changing the MPR selection mechanism and the routing table calculation, the following revisions are made to develop the QoS OLSR model.

QoS OLSR uses the media idle time to reflect the available bandwidth over a link. Modifying the standard OPNET Wireless LAN model achieves this task. Each OPNET OLSR node connects to the wireless media. The OPNET Wireless LAN simulation model includes a transmitter, and a receiver. If a node is sending packets, its transmitter becomes busy. If there are other nodes beginning transmission within the interference range of the current node, its receiver senses the busy media and sends a media busy signal. As the OPNET Wireless LAN model already defines functionalities to capture changes of the media, the media idle time calculation, using a sliding window over the past 5 seconds, is straightforward.

Also, the QoS OLSR versions need to know the available bandwidth on the neighbor link to select MPRs, and the available bandwidth of the far-away links to compute the routing table. As idle time should be used to calculate the available bandwidth on the links, we revise the format of OLSR Hello and TC messages to include the idle time.

a. Hello message: in addition to the original information such as neighbor address and neighbor link type, a node also includes its own idle time in the Hello messages. Upon receiving a Hello message from its neighbor, a node reads the neighbor idle time, and selects MPRs using the QoS MPR selection algorithm.

b. TC message: the TC message originator not only puts its own idle time in TC messages, but also piggybacks its MPR selectors' idle times, which are obtained from the Hello messages. When a node receives TC messages, it knows the idle time information of both the TC message originator and the MPR selectors, thus gets information about the links and the link bandwidth between the TC message originator and its MPR selectors. In this way, it learns the partial network topology and the bandwidth condition of that partial network, and is ready to calculate the routing table.
Furthermore, QoS OLSR needs to decide when to originate a TC message. In the original OLSR, if a node detects changes in its MPR selector, it generates a new TC message to propagate the changes in the network topology. In QoS OLSR, however, changes in link bandwidth condition must also be propagated for the correct computation of the best bandwidth routes. If an MPR generates a TC message as soon as it detects a bandwidth change over the link between its MPR selector and itself, there will be many messages flooding into the network, causing extremely high overhead. So in our QoS OLSR version, a "threshold" of bandwidth change is defined. If an MPR finds there is "significant bandwidth change", it will generate a new TC message informing the whole network about the change, enabling other nodes to update their routing table reflecting such changes. There is a tradeoff in how to define the "threshold". On one hand, if the "threshold" is low, TC messages will be generated as soon as there is a small percentage change of the bandwidth. That will cause frequent generation of TC messages, introducing high overhead, although more accurate bandwidth information is obtained. On the other hand, if the "threshold" is high, TC messages will not be generated until there is a very large percentage change of the bandwidth. Thus, the overhead is reduced, but the nodes only obtain relatively inaccurate bandwidth information. In the rest of the paper, we will utilize different "threshold" values to compare the network performance, and analyze the performance tradeoffs.

4 OPNET Simulation

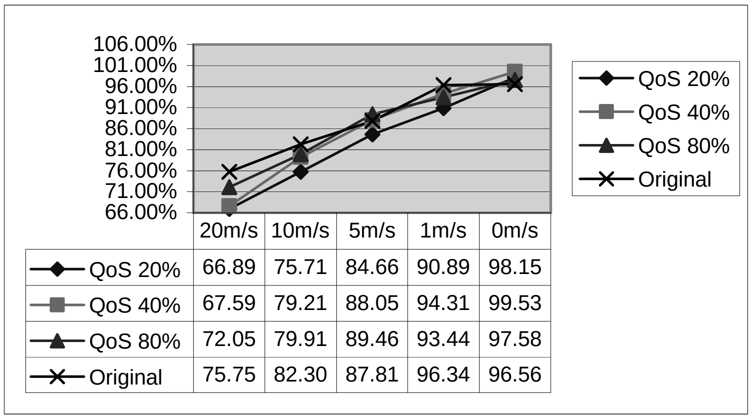
The following environment parameters are defined for OPNET simulations:
Movement Space: 1000m x 1000m flat space
Number of Nodes: 50 nodes
Simulation Time: 900 seconds.
Movement Model: each node randomly selects a destination in the 1000m x 1000m area, moves to that destination at a speed distributed uniformly between 0 and "maximum speed". After it reaches the destination, the node selects another destination and another speed between 0 and "maximum speed", and moves again. In the set of experiments reported here, we use 5 different "maximum speed" values: 20m/s, 10m/s, 5m/s, 1m/s, and 0m/s.
Communication Model: In each simulation, there are 20 communication pairs. Each source sends 64-byte UDP packets at a rate of 4 packets/second. So in total, 80 packets are sent each second.
OPNET Model Parameter: see Table 3.
Routing Protocol: 4 routing protocols – Original OLSR, QoS OLSR with 20% bandwidth updating threshold (20% OLSR), QoS OLSR with 40% bandwidth updating threshold (40% OLSR), and QoS OLSR with 80% bandwidth updating threshold (80%

OLSR). All the QoS OLSR algorithms use the OLSR_R2 [5] mechanism to select MPRs, and the "Extended BF" algorithm to calculate the routing table.

Table 3. OPNET Model Parameter

OLSR Parameters	Hello Interval	*0.5s*
	TC Interval	2s
Wireless LAN Parameters	Data Rate	2 Mbps
	Buffer Size	256000 bits
	Retry Limit	7
	Wireless LAN Propagation Range	250 M

The OPNET simulation results are grouped into two sets: Basic Performance and QoS Performance (The data shown in this section are the average value from multiple runs.)
Basic Performance – the basic performance is measured by a set of metrics used to evaluate most routing protocols: "Packet Delivery Ratio" and "End to End Delay".
Packet Delivery Ratio: percentage of packets that successfully reach the receiving nodes each second.
End to End Delay: the average time between a packet being sent and being received
QoS performance –metrics that relate to the bandwidth QoS routing studied in this paper: "Error Rate" and "Bandwidth Difference".
Error Rate: the percentage of times the routing algorithms do not find the optimal bandwidth path.
Bandwidth Difference: the average difference between the optimal bandwidth and the detected non-optimal bandwidth in percentage.



	20m/s	10m/s	5m/s	1m/s	0m/s
◆ QoS 20%	66.89	75.71	84.66	90.89	98.15
■ QoS 40%	67.59	79.21	88.05	94.31	99.53
▲ QoS 80%	72.05	79.91	89.46	93.44	97.58
× Original	75.75	82.30	87.81	96.34	96.56

Fig. 2. Packet Delivery Ratio under different movement patterns

Fig. 2 compares the packet delivery ratio of the 4 algorithms. From high movement to low movement, packet delivery ratio for all algorithms rises continuously. With lower movement, the established links between the nodes have a lower probability of breaking, thus, there are less stale routes in the node routing tables, which results in a higher ratio for correct packet delivery. In low movement scenarios (speed 5m/s, 1m/s and 0m/s), the 4 algorithms achieve similar packet delivery ratio. However, in high movement scenario, the original OLSR protocol has higher packet delivery ratio than the 3 QoS versions, especially with a speed of 20m/s where the performance difference between the QoS versions of OLSR and the original OLSR protocol are statistically significant. There are two main reasons:

a. High Overhead: The original OLSR protocol concentrates on how to reduce the overhead, and minimizes the MPR sets to reduce the TC messages flooding into the network. However, the QoS versions of OLSR select the best bandwidth path, so in their MPR selection mechanism, they select neighbors with high idle time as MPR, resulting in a larger MPR set than the original OLSR protocol. So more TC messages are generated and relayed into the network by QoS OLSR versions. (See Fig. 3)

For all algorithms, there are fewer TC messages sent at lower movement than at higher movement. This is because at lower movement, less TC messages are generated to reflect topology changes. Also, 20% OLSR has the highest number of TC messages generated and relayed, while the original OLSR protocol has the least number of TC messages. Under the same speed, the difference of TC messages sent between the original OLSR protocol and the 3 QoS OLSR versions comes from two aspects:

1. The original OLSR protocol only generates TC messages to reflect topology change, while QoS OLSR versions also need to generate TC messages to reflect bandwidth change; with a lower bandwidth update threshold, more TC messages are generated to reflect bandwidth change, causing the highest overhead in 20% OLSR
2. QoS OLSR versions have larger MPR sets than the original OLSR protocol, so more TC messages are generated and relayed by the larger MPR sets. Among the QoS OLSR algorithms, 20% OLSR may select more MPRs than 40% and 80% OLSR. With the possibly larger MPR set, more TC messages are generated and relayed by 20% OLSR than 40% OLSR and 80% OLSR.

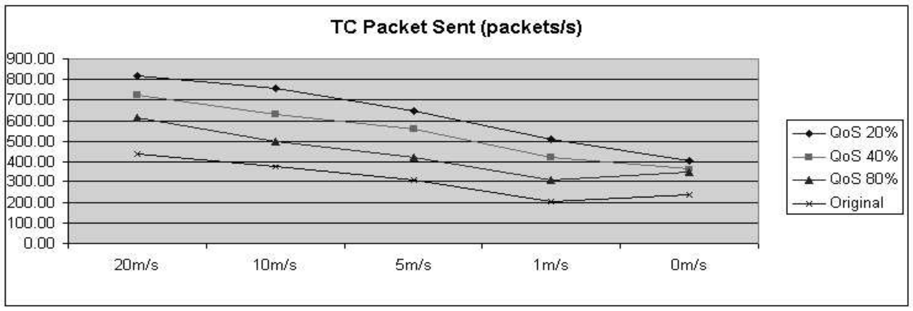
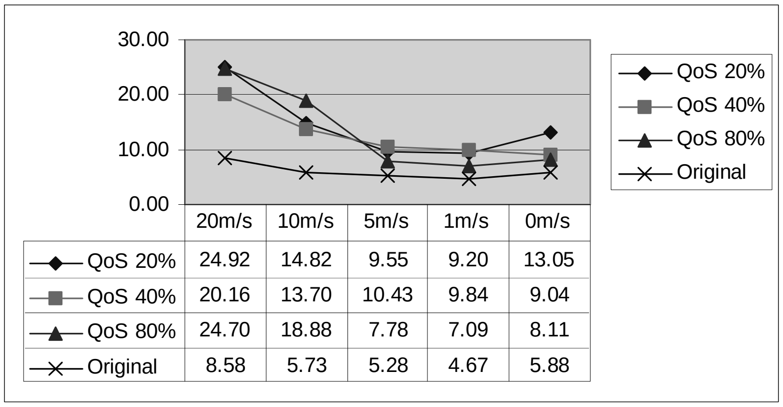


Fig. 3. Average TC message overhead in the network (in packets/s) for the 4 algorithms

With higher overhead introduced into the network, especially for the 20% OLSR at higher movement, the wireless media is more heavily loaded.

b. Incorrect Routing Table: if there are overlapped two hop neighbors covered by multiple MPRs, there is a high probability that TC packets collide at these neighbors, resulting in inaccurate routing tables. This problem happens in all 4 OLSR algorithms. But because of the different MPR selection mechanism, the QoS OLSR algorithms have more overlapped two hop neighbors than the original OLSR protocol, causing more TC message collisions.



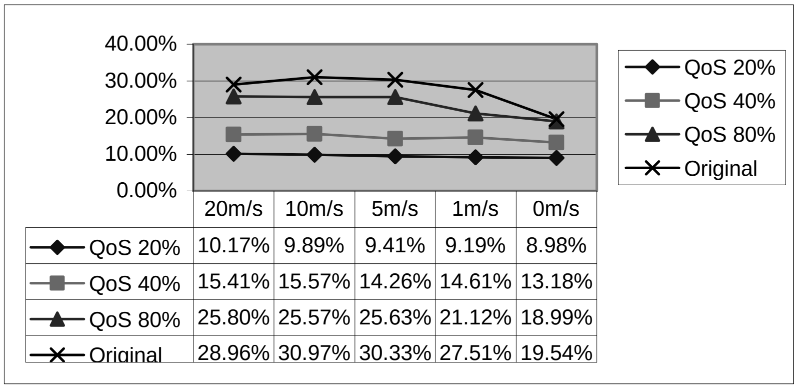
	20m/s	10m/s	5m/s	1m/s	0m/s
◆ QoS 20%	24.92	14.82	9.55	9.20	13.05
■ QoS 40%	20.16	13.70	10.43	9.84	9.04
▲ QoS 80%	24.70	18.88	7.78	7.09	8.11
× Original	8.58	5.73	5.28	4.67	5.88

Fig. 4. End-To-End Delay (ms) of data packets for 4 OLSR algorithms

Fig. 4 shows the End-to-End Delay for each algorithm under each movement pattern. Basically, for all movement patterns, the original OLSR has the lowest delay. Furthermore, in the high movement scenarios, the delay between the QoS versions of OLSR and the original OLSR protocol is statistically different. As the original OLSR has the lowest overhead, its network is the least congested, resulting in the least delay. Also, the original OLSR algorithm always computes the shortest hop path, while the QoS OLSR versions may compute longer paths because they target the best bottleneck bandwidth path, which also affects the end-to-end delay of the data packets.

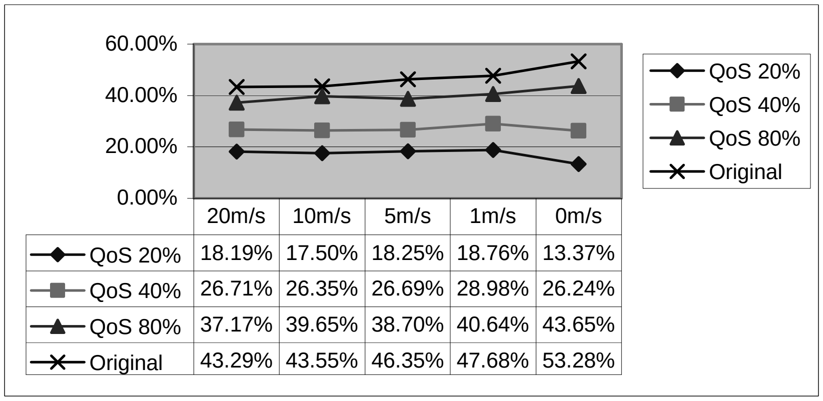
For the three QoS OLSR algorithms, we can see that at higher movement speed (20m/s and 10m/s), the 80% threshold QoS OLSR has a higher delay, while at lower movement speed (5m/s, 1m/s and 0m/s), its delay is close to the original OLSR. To analyze this phenomenon, recall that the 80% threshold QoS OLSR has the most inaccurate bandwidth information of the network, which means that the routing algorithm may select a route that is still relatively congested. At higher movement, all the QoS OLSR algorithms have higher overhead because of the frequent updates due to topology change (see Fig. 3), causing the network to be congested. Working on the already congested networks, 20% QoS OLSR and 40% QoS OLSR do a better job in directing the traffic to the less congested routes, resulting in the lower packet delay. However, at lower movement speed, there are much less topology updates, so the more frequently sent bandwidth update messages in 20% and 40% OLSR tend to make the network busy, resulting in a larger delay than the 80% OLSR.

Fig. 5 and Fig. 6 show the “Average Difference” and “Error Rate” among the 4 algorithms under different movement patterns. All QoS OLSR outperform the original OLSR in both the “Error Rate” and “Bandwidth Difference”. Among the QoS OLSR algorithms, 20% OLSR updates the bandwidth condition most frequently, introducing the highest overhead, but gets the most accurate bandwidth information. So the routes it calculates are closest to the optimal routes. The 40% and 80% OLSR, however, update bandwidth information less frequently, introducing less overhead, but their QoS performances are not as good as that of 20% OLSR.



	20m/s	10m/s	5m/s	1m/s	0m/s
◆ QoS 20%	10.17%	9.89%	9.41%	9.19%	8.98%
■ QoS 40%	15.41%	15.57%	14.26%	14.61%	13.18%
▲ QoS 80%	25.80%	25.57%	25.63%	21.12%	18.99%
✕ Original	28.96%	30.97%	30.33%	27.51%	19.54%

Fig. 5. Comparison of Average Bandwidth Difference



	20m/s	10m/s	5m/s	1m/s	0m/s
◆ QoS 20%	18.19%	17.50%	18.25%	18.76%	13.37%
■ QoS 40%	26.71%	26.35%	26.69%	28.98%	26.24%
▲ QoS 80%	37.17%	39.65%	38.70%	40.64%	43.65%
✕ Original	43.29%	43.55%	46.35%	47.68%	53.28%

Fig. 6. Error Rate

The results for “Bandwidth Difference” and “Error Rate” of each algorithm are calculated based on its own network conditions – the bandwidth difference between the routes the routing algorithm calculated and the optimal paths in the network in which the routing algorithm works. However, because the QoS OLSR versions introduce more overhead than the original OLSR protocol, the networks in which the QoS OLSR versions work may have worse overall available bandwidth than a network that runs the original OLSR algorithm. So one may question if the QoS OLSR versions really improve the route bandwidth condition. To explore this question, for each scenario and OLSR version, the average available bandwidth over both the optimal routes and the paths found by the routing algorithms are computed. In the following, as available bandwidth is directly related to idle time in percentage, we report available bandwidth as percentage of idle time.

To calculate the average available bandwidth on the routes the routing algorithms find, first we obtain the average optimal route bandwidth (see Table 4.).

The results shown are consistent with our former analysis: The lower the movement speed, the less the overhead all the OLSR algorithms introduce into the network. So from speed 20m/s to 0m/s, the optimal bandwidth conditions for all the OLSR algorithms rise continuously. The original OLSR algorithm has the least overhead, so the network that runs the original OLSR algorithm always has the best bandwidth condition. Compared with 80% OLSR, 40% OLSR evenly directs traffic throughout the network, so under high movement (speed 20m/s, and 10m/s) where the wireless media are rather busy, 40% OLSR has better optimal bandwidth routes than that of the 80% OLSR, although it has more overhead than 80% OLSR. Under low movement (speed 5m/s, 1m/s, and 0m/s), the added overhead of 40% OLSR has a negative effect on the network bandwidth condition, thus the 40% OLSR has less optimal bandwidth than 80% OLSR. As the 20% OLSR has the highest overhead, its optimal bandwidth routes have the lowest available bandwidth.

From the results, we can also see that because the original OLSR has the lowest overhead, it provides the network with the best bandwidth condition – its best bandwidth paths have the highest bottleneck bandwidth among all the OLSR versions. However, as the original OLSR does not make efforts to find these optimal bandwidth paths, the actual path it finds may have a lower bandwidth than the paths the QoS OLSR versions find. In the following, we compare and analyze the actual bandwidth on the path the 4 versions of OLSR calculate.

Table 4. Avaliable bandwidth on the optimal paths (measured as idle time)

Algorithm	20m/s	10m/s	5m/s	1m/s	0m/s
QoS 20%	77.68%	80.93%	82.29%	84.69%	89.73%
QoS 40%	82.23%	84.92%	86.29%	87.46%	90.17%
QoS 80%	78.17%	84.27%	87.17%	90.08%	92.34%
Original	87.07%	87.28%	90.63%	91.14%	93.08%

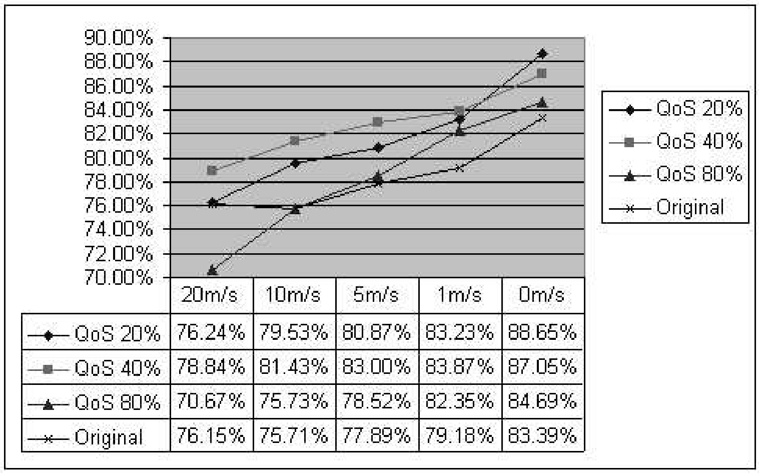
The actual average available bandwidth the routing algorithms calculate

= the available bandwidth on the optimal paths x ((1- “Bandwidth Difference”) x “Error Rate”) + (1- “Error Rate”))

= the available bandwidth on the optimal paths x (1- “Bandwidth Difference” x “Error Rate”)

Using the “Bandwidth Difference” and “Error Rate” values, the result for actual average available bandwidth the routing algorithms calculated is shown (see Fig. 7). We can see that although the QoS OLSR versions introduce more overhead, the routes they compute still have higher available bandwidth than the routes in a network running the original OLSR. In movement patterns with maximum speed 20m/s, 10m/s, 5m/s, and 1m/s, among all the OLSR algorithms, the 40% OLSR always computes the route with the best available bandwidth, as it has less overhead than 20% OLSR and more accurate bandwidth information than 80% OLSR. In the fixed network case, because of few topology updates, all the algorithms have low overhead. Thus, 20% OLSR finds the routes with highest bandwidth, for it has the most accurate bandwidth information.

Based on these results, we conclude that the QoS OLSR versions do achieve bandwidth improvement over the original OLSR algorithm.



	20m/s	10m/s	5m/s	1m/s	0m/s
◆ QoS 20%	76.24%	79.53%	80.87%	83.23%	88.65%
■ QoS 40%	78.84%	81.43%	83.00%	83.87%	87.05%
▲ QoS 80%	70.67%	75.73%	78.52%	82.35%	84.69%
× Original	76.15%	75.71%	77.89%	79.18%	83.39%

Fig. 7. Average available bandwidth (in idle time) on the routes of the 4 OLSR algorithms

5 Analysis of QoS Routing and Future Work

As mentioned in Section 1, there is a trade-off between the QoS performance that the QoS routing protocol achieves and the additional cost it introduces. The QoS OLSR versions we study in this paper confirm this – QoS OLSR algorithms do enhance the network QoS performance. However, in order to achieve this improvement, additional "protocol overhead" is also introduced, which degrades the performance of these QoS routing protocols, especially with respect to "Packet Delivery Ratio" and "End-to-End Delay" in high mobility cases. Does this then imply that we should abandon proactive QoS routing and switch to on-demand QoS routing because of the cost? Not necessarily:

- We do not know if on-demand routing algorithms have the same overhead problems. [3] discusses the performance of the "ticket-based probing" algorithm in a delay-constrained environment, calculating what percentage of routes that the algorithm finds meet the delay request. But it fails to analyze other aspects of the routing algorithm, such as control overhead, packet delivery ratio etc. [11] tests the CEDAR algorithm using bandwidth as the QoS parameter, giving a detailed performance evaluation. However, [11] does not experiment with node movement. Nor does it run the simulation in a real shared-channel environment, and the impact of channel interference and packet collision are not considered.
- Many proposed proactive QoS routing algorithm such as [10] and [7] just present a basic idea, without performance evaluation. So it is not clear whether the negative effect on the routing performance caused by the additional routing overhead is a common problem to proactive QoS routing.

Based on the above analysis, proactive QoS routing is still worth studying. As the added overhead is the main cost that affects the QoS routing algorithm's performance, the future work on QoS routing in Ad-Hoc networks may be focused on how to reduce the overhead. Our future work plans include the following:

- TC packet collisions at the 2-hop neighbors cause the problem of stale routing tables. To avoid this problem, we can add some jitter mechanism into the OLSR protocol – when an MPR receives a TC message, it waits for a random delay time before it relays that TC message, instead of relaying it immediately.
- Compared to the data packet load, the additional overhead the QoS OLSR versions introduce use a large amount of link bandwidth. This overhead is relatively independent of the nominal link bandwidth. We plan to explore whether the use of 802.11b, with up to 11 Mbps data rate, reduces the added network load and the resulting negative effect on the delivery ratio and delay.

Acknowledgements: This work is funded by the Defense Research and Development Canada (DRDC) and benefited from various discussions with members of the CRC RNS mobile networking group.

References

1. G. S. Ahn, A. T. Campbell, A. Veres and L. H. Sun, "SWAN: Service Differentiation in Stateless Wireless Ad-Hoc Networks", IEEE Computer and Communications Societies, 2002, pages 457-466, June 2002
2. G. Apostolopoulos, R. Guerin, S. Kamat, and S. K. Tripathi, "Quality of Service Based Routing: A Performance Perspective", Association for Computing Machinery's Special Interest Group on Data Communication '98, pages 17-28, September 1998
3. S. Chen and K. Nahrsted, "Distributed Quality-of-Service Routing in Ad-Hoc Networks", IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, August 1999, pages 1488-1505
4. S. Chen and K. Nahrstedt, "An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions", IEEE Network Magazine, Vol.12, No.6, pages 64-79, November 1998
5. Y. Ge, T. Kunz and L. Lamont, "Quality of Service Routing in Ad-Hoc Networks Using OLSR", Proceeding of the 36th Hawaii International Conference on System Sciences (HICSS-36), Hawaii, USA, January 2003, ISBN 0-7695-1874-5, IEEE 2003.
6. R. Guerin and D. Willimas, "Qos Routing Mechanisms and OSPF Extensions", draft-qos-routing-ospf-00.txt", Internet-Draft, Internet Engineering Task Force, November 1996
7. Iwata, C. C. Chiang, G. Pei, M. Gerla and T. Chen, "Scalable Routing Strategies for Ad-Hoc Wireless Networks", IEEE Journal on Selected Areas in Communications, Vol.17, No.8, pages 1369-1379, August 1999
8. P. Jacquet, P. Muhlethaler, A. Qayyum, A. Laouiti, L. Viennot, T. Clauseen, "Optimized Link State Routing Protocol draft-ietf-manet-olsr-05.txt", INTERNET-DRAFT, IETF MANET Working Group
9. C. R. Lin and J. S. Liu, "QoS Routing in Ad-Hoc Wireless Networks", IEEE Journal On Selected Areas In Communications, Vol.17, No.8, pages 1426-1438, August 1999
10. R. Ramanathan and M. Steenstrup, "Hierarchically-Organized, Multihop Mobile Wireless Networks for Quality-of-Service Support", Mobile Networks and Applications, Vol.3, pages 101-119, 1998
11. P. Sinha, R. Sivakumar and V. Bharghanan, "CEDAR: a Core-Extraction Distributed Ad-Hoc Routing Algorithm", IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, August 1999, pages 1454-1465