Quality of Service Routing in Ad-Hoc Networks Using OLSR

Ying Ge
Communications Research Centre
ying.ge@crc.ca

Thomas Kunz
Carleton University
tkunz@sce.carleton.ca

Louise Lamont
Communications Research Centre
louise.lamont@crc.ca

Abstract

In an ad-hoc network, all communication is done over wireless media, without the help of wired base stations. While many routing protocols have been developed to find and maintain routes based on a best-effort service model, quality-of-service (QoS) routing in an ad-hoc network is difficult because the network topology may change constantly and the available state information for routing is inherently imprecise.

In this paper, we discuss how to support QoS routing in OLSR (Optimized Link State Routing Protocol, one of the routing protocols under study by the IETF MANET Working Group). We develop heuristics that allow OLSR to find the maximum bandwidth path, show through simulation that these heuristics do improve OLSR in the static network case, and finally, we prove that for our ad-hoc network model, two of the heuristics are indeed optimal (i.e., guarantee that the highest-bandwidth path between any two nodes is found).

1. Introduction

An ad-hoc network (MANET) is a dynamic multi-hop wireless network that is established by a group of mobile nodes on a shared wireless channel. Much work has been done on routing in ad-hoc networks, but most of them focus only on best-effort data traffic. However, recently, because of the rising popularity of multimedia applications and potential commercial usage of MANETs, QoS support in ad-hoc networks has become a topic of great interest in the wireless area.

To support QoS, the link state information such as delay, bandwidth, jitter, cost, loss rate and error rate in the network should be available and manageable. However, getting and managing the link state information in a MANET is by all means not trivial because the quality of a wireless link changes with the surrounding circumstance. Furthermore, the resource limitations and the mobility of hosts add to the complexity. In spite of these difficulties, some protocols on QoS routing in MANETs have been proposed, such as CEDAR [1] or ticket-based probing [2]. These protocols provide on-demand routing, where a route is found based on the pre-known QoS requirements. An on-demand routing protocol is easy to understand and design. However, the unpredictable nature of ad-hoc networks and the requirement of quick reaction to QoS routing 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. Based on such consideration, we are studying the approach of proactive QoS routing, which is to pre-compute the best route, based on the QoS constraint among all the possible routes. Our approach is to integrate the QoS feature into OLSR (Optimized Link State Protocol) [3], which is a pro-active routing protocol.

The rest of the report is organized as follows: a brief description of OLSR is given in Section 2, the network model and the consideration of which QoS metrics should be supported in our algorithm is discussed in Section 3. Section 4 discusses three heuristics that we propose to enhance OLSR. Section 5 gives the simulation result based on the static network case. Section 6 contains the proof that two of our heuristics are indeed optimal, i.e., guarantee that the QoS-optimal path is found.

2. Description of OLSR

In [3], the IETF MANET Working Group introduces the Optimized Link State Routing (OLSR) protocol for mobile ad-hoc networks. 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. The node N, which is selected as a multipoint relay by its neighbors, periodically announces the information about who has selected it as an MPR. Such a 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.

Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03)
0-7695-1874-5/03 $17.00 © 2002 IEEE

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 [3] for MPR selection is to iteratively select a 1-hop neighbor which reaches the maximum number of uncovered 2-hop neighbors as MPR. If there is a tie, the one with higher degree (more neighbors) is chosen.

Table 1 shows how node B selects MPR(s), based on the network depicted in Figure 1:

Table 1. MPR selection in OLSR

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).

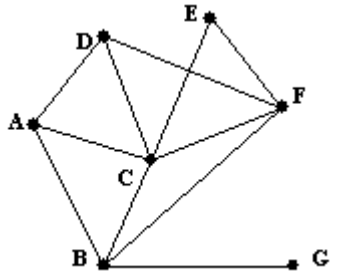

Figure 1. Network example for MPR selection

3. QoS metrics and network model for simulation

QoS routing protocols search for routes with sufficient resources to support various QoS requirements. However, finding a path subject to multiple constraints is inherently hard. Polynomial-time algorithms for the problem may not exist [4].

Considering such difficulties, together with the fact that node movements in ad-hoc networks make the problem even more complex, currently, the following decisions are made for our study and simulation: First, for the time being, we only consider "bandwidth" as the QoS routing constraint. This is because bandwidth guarantee is one of the most critical requirements of real-time applications. Our goal is to find an optimal bandwidth path—the one has the highest bottleneck bandwidth among all the paths from source to destination.

Second, we assume that the ad-hoc network topology is stable at one moment so that we can study the QoS routing problem on that stable graph. Actually, there are various circumstances where ad-hoc networks are rather stable: A wireless network consisting of desktop PCs, laptop computers and printers for home business may keep its original topology for a long time until someone moves one of the laptops to another room, for example. One avenue of future work pointed out below is to explore how fast our routing algorithm tracks changes, both to the underlying topology as well as the available link bandwidths.

Third, 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, bandwidth computation is a complex issue. Many papers such as [5] 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), similar to [6]. MAC protocols such as IEEE 802.11 are based on a carrier-sense capability of each node. We exploit this capability to determine, locally at each node, for what percentage of time the medium has been busy in the recent past. A busy medium may indicate that a neighbor is transmitting data over the shared wireless channel. However, it may also indicate that nodes even further away, but still within interference range, are using the media. A node can only successfully transmit during times when neither its immediate neighbors nor other nodes in its interference range are transmitting. This characterization of the available bandwidth is superior to and with lower overhead than proposals where nodes communicate with their immediate neighbors to exchange information about their committed bandwidth, ignoring nodes further away. The "available bandwidth" over a link connecting nodes A and B is proportional to the minimum of A's idle time and B's idle time, since both nodes have to be available for a successful transmission. Since the number of nodes and the traffic between them in each node's interference range is different, the idle times of two adjacent nodes may well be substantially different. However, due to the shared nature of the wireless medium, it is always the case that the link bandwidth between two adjacent nodes A and B is always equal to or better than the bandwidth over any 2-hop connection between A and B (i.e., via some intermediate node C), as will be explained in more detail in Section 6.. Depending on the underlying MAC protocol, a node may not be able to use

the whole idle time. In IEEE 802.11 networks, for example, a node will wait for a random backoff time after it detects that the link is idle. However, as such backoff times are deliberately kept short, we neglect them in the remainder of this paper. Because of the unstable nature of ad-hoc networks, it is also important to decide how the idle time, which reflects the network traffic condition, should be maintained and updated. However, in this paper, we are dealing with "network snapshots", evaluating the route selection heuristics in OLSR, so this issue is not touched here.

4. Integrating OLSR and QoS routing

4.1. Limitations of OLSR in QoS routing

As mentioned, 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 unreached 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 Section 2 again (see also Figure 2.) The numbers along the lines indicate the available bandwidth over the links. As explained in Section 2, 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, we revise it in two aspects: MPR selection and routing table computation.

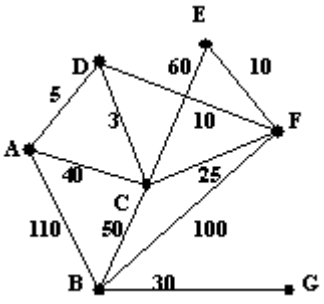

Figure 2. Network example for MPR selection

4.2. Changing the MPR selection criteria

The decision of 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, three revised MPR selection algorithms are presented.

4.2.1. OLSR_R1. In the first algorithm, MPR selection is almost the same as that of OLSR described in Section 2. However, when there are more than one 1-hop neighbors covering the same number of uncovered 2-hop neighbors, the one with the largest bandwidth link to the current node is selected as MPR.

The network in Figure 2 would select MPRs for node B as follows, based on OLSR_R1:

Table 2. MPR selection in OLSR_R1

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

Between C and F, F is selected as B's MPR because it has the larger bandwidth.

4.2.2 OLSR_R2. The idea behind OLSR_R2 is to select the best bandwidth neighbors as MPRs until all the 2-hop neighbors are covered.

For example, using this algorithm, based on Figure 2, node B's MPR(s) would be:

Table 3. MPR selection in 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 largest 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.

4.2.3. OLSR_R3. The third algorithm selects the MPRs in a way such that all the 2-hop neighbors have the optimal bandwidth path through the MPRs to the current node. Here, optimal bandwidth path means the bottleneck bandwidth path is the largest among all the possible paths. Looking again at node B in Figure 2 as an example, in order to cover D; A, C, or F need to be chosen as an MPR. Bandwidths available from B to D for three different routes are:

B –110- A –5- D bottleneck bandwidth is 5
B –50- C –3- D bottleneck bandwidth is 3
B –100- F –10- D bottleneck bandwidth is 10

The algorithm chooses the route with the largest bottleneck (in 2 hops). In this case the chosen MPR is F. In

the same way, C is chosen as MPR by B to cover E.

Table 4. MPR Selection in OLSR_R3

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

The revised OLSR MPR selection algorithms may improve the chance that a better bandwidth route is found. However, by using such algorithms, the overhead may also increase compared with the original OLSR algorithm because we may increase the number of MPRs in the network, especially for OLSR_R3, which may select a different MPR for each 2-hop neighbor.

4.3. Routing table calculation

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. Here, the "maximum bandwidth spanning tree" is proposed to compute the optimal bandwidth path.

Similar to the ordinary definition of a "minimum spanning tree", the definition of the "maximum bandwidth spanning tree" is: using the bandwidth over the link between two nodes as weight, a maximum bandwidth spanning tree is a tree connecting all the nodes in the network whose total weight is maximal among all the possible trees.

Theorem 1: **The optimal bandwidth constraint path from source to destination is along the maximum bandwidth spanning tree edge.**

Proof (by contradiction): Suppose there is a maximum bandwidth spanning tree T. Assume that the route from s to d in T is: s->b->g->e->d, and in that route, there is a link l connecting b and g, which is the bottleneck bandwidth link of the route. Assume that there is another route from s to d, whose bottleneck bandwidth is greater than the route in T. Without loss of generality, we assume that there is a better route: s->c->e->d, and the link l' connecting c and e is not in T, see Figure 3a.[1] Furthermore, since l' is a link on a better route, its weight has to exceed the weight of the bottleneck link l: weight(l')>weight(l).

Consider the tree T. When we remove l from T, T is divided into two separate graphs, G' and G", where s is in G'. T is originally a spanning tree, there is only one route from s to d in T through l. So after removing l, s and d are no longer connected, then s and d must be in different parts. As s is in G', d is in G", see Figure 3b. When we add l' to (b), the result is graph G, see Figure 3c.

1. We first show that G is still a spanning tree:

s and d are in two separate graphs, s and d are not connected. As defined, s and d can be connected through l', which means when we add l' to G' and G", the two separate graphs, G' and G", are connected together by l'.

1) From the way we construct G, we can see that all the nodes that are originally connected by T are now connected by G.
2) G' and G" are originally part of spanning tree T, so G' and G" are acyclic. l connects the originally separated G' and G", G'+G"+l is still acyclic. Which means G is acyclic.

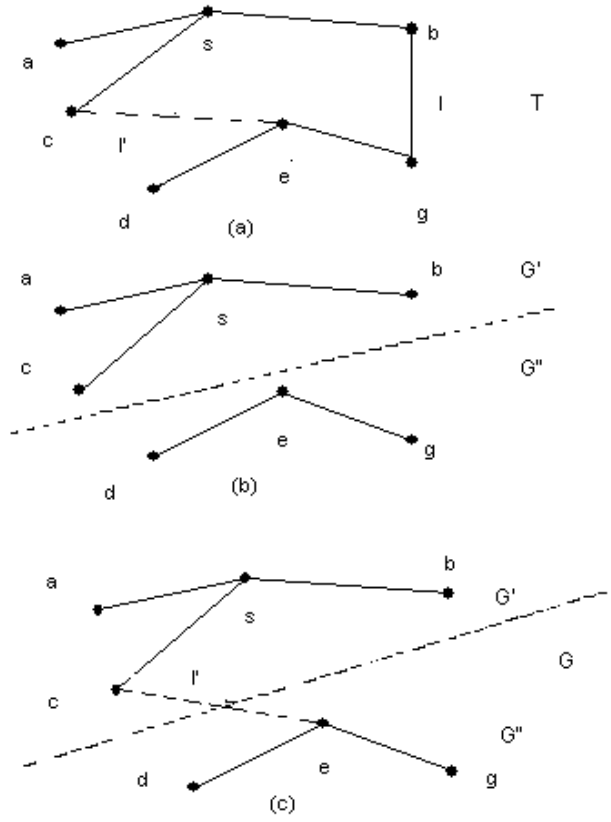

Figure 3. Proof for max bandwidth spanning tree

2. Based on the above, G is a spanning tree, whose weight is: total weight of the original tree – weight(l) + weight (l') > total weight of the original tree.

However, according to the definition of maximum bandwidth spanning tree, the total weight of such a tree is the largest among all the trees. So our above assumption is contrary to the definition, which means the optimal

[1] We can safely assume that l' is not in T: otherwise, there would be two paths from s to d, which would violate the basic premise that we are dealing with trees. Also, the assumption that only l' is not in T is correct: If the better route contains several links that are not in T, we can easily substitute them with the edges in T because T reaches each node in the graph.

bottleneck bandwidth path is on the maximum bandwidth spanning tree edge. This completes the proof.

Therefore, by building the "maximum bandwidth spanning tree", the node can find the optimal path in its known partial network topology. In Section 6 we will prove that each node indeed has enough partial topology information to correctly construct this graph. In our simulations, we combine the 3 MPR selection heuristics with a distributed "maximum bandwidth spanning tree" algorithm to compare the result with the original OLSR algorithm.

5. Simulation and results

We simulate our MPR selection algorithms and compare the results. Using a simulator written in C++, we randomly generate network topologies, and perform the computations on these fixed graphs, representing snapshots of the ad-hoc network state. As mentioned in Section 3, for the time being, we are currently not investigating how long it would take to propagate and adapt to changes in topology or available bandwidth. We also simplified the experiments by randomly assigning traffic in each network snapshot. The following are the simulation details.

5.1. Network scenario

-- Network area: 1000 M × 1000 M
-- Number of nodes: 100
-- Transmission range: 100 M, 200 M, 300 M
-- Bandwidth: Based on the analysis in Section 3, the available link bandwidth is computed as follows: Each node is randomly assigned an "idle time" ranging from 0 to 1. The available link bandwidth between two nodes is equal to the minimum of their idle time × maximum bandwidth. Here, we consider that in the ad-hoc network, each link has the same maximum bandwidth, 2 Mbps. For example, if node a's idle time is 0.5 and node b's idle time is 0.3, then the available bandwidth over link ab is: 0.3 × 2Mbps = 600 kbps. These randomly generated "idle times" reflect the traffic condition in the network snapshot as the consumed bandwidth over each link reflects the traffic flows over that link.

5.2. Simulation object

We implemented a total of 5 algorithms and applied them to the randomly generated network snapshots:
1) OLSR (Section 2)
2) OLSR_R1 (Section 4.2.1)
3) OLSR_R2 (Section 4.2.2)
4) OLSR_R3 (Section 4.2.3)
5) Pure link state algorithm: each node floods its link state information into the entire network. By doing this, the path with maximum bottleneck bandwidth is guaranteed to be found.

Routes found by algorithms 1) through 4) are compared with the route found by algorithm 5), using the simulation model and metrics discussed below.

5.3. Simulation model

For each transmission range (100m, 200m, 300m), 100 network snapshots are generated. For each connected pair in the network, we run the 5 algorithms mentioned in Section 5.2 to find a route between each pair of nodes in the network. Results obtained show how often the route found by the first 4 algorithms (standard OLSR, OLSR_R1, OLSR_R2, and OLSR_R3) has lower bandwidth than the route found by a pure link state algorithm. If we cannot find the optimal path using the first 4 algorithms, we will present how suboptimal the result is. Also, we characterize and compare the overhead of these 5 algorithms.

5.4. Simulation results

Results are given in two categories: performance and cost. To further analyze the results, we also collect information about specific network characteristics.

5.4.1 Performance. Performance is characterized by "error rate" and "average difference":

"*Error rate*" represents the percentage of times the standard OLSR, OLSR_R1, OLSR_R2, and OLSR_R3 algorithms do not find the optimal bandwidth path. In other words, error rate = total number of bad routes in 100 snapshots computed by OLSR / total number of optimum routes in 100 snapshots.

"*Average difference*" is the average difference between the optimal bandwidth and current bandwidth found in routing algorithms in percentage: result = average of (bandwidth on optimal path-bandwidth on route computed)/bandwidth on optimal path, when the optimum routes are not found. The larger the figure is, the worse the result.

5.4.2 Cost. The cost of the protocol is measured by "overhead" and "MPR percentage":

"*Overhead*": How many control messages (messages originated by the nodes indicating who select it as MPR) are transmitted/re-transmitted in the network. Overhead = the average number of control messages transmitted per snapshot/100 (the number of nodes in network).

"*MPR number*": Average number of MPRs in the network. The more MPRs in the network, the higher the overhead.

5.4.3 Network characteristics. We collect the average number of 1-hop neighbors and 2-hop neighbors for a node. These values effect the MPR number in the network. On one hand, the more 1-hop neighbors a node has, the less MPRs it may select, because with high probability a small subset of it's 1-hop neighbor can reach a high number of the 2-hop neighbors (assuming high connectivity of the network). On the other hand, the more 2-hop neighbors a node has, the more MPRs may be needed to cover them all.

5.5. Simulation results and analysis

First we consider the results of all 5 algorithms for the same network, using the 300 M transmission range network as example. Considering the performance of the 4 OLSR algorithms, we see that the standard OLSR has the worst performance – it has the highest "Error Rate" and "Average Difference", which means in the 300 M transmission range network, the standard OLSR has the highest probability that it can not find the best bandwidth path. At the same time, the bandwidth difference between the path it finds and that of the optimal path is also large. Although OLSR_R1 uses the same MPR selection algorithm as standard OLSR, it achieves a large improvement in performance, which shows lower "Error Rate" and lower "Average Difference". Such improvement is effected by the "maximum spanning tree" algorithm, which finds the optimal path on the partial network a node learns from the procedure of MPR selector declaration and re-transmission. However, OLSR_R1 does not always find an optimal path, as its MPR selection algorithm may omit the optimal bandwidth link from the partial network topology the node learned (see the example of Section 4.1). However, OLSR_R2 and OLSR_R3 show very good results – each time, these two algorithms find the optimal bandwidth route. The explanation is given in Section 6. As mentioned earlier, costs are directly related to the number of MPRs selected by the algorithms. The higher the number of MPRs in the network, the higher is the overhead. This relationship is clearly shown in the "Cost" category. Of the 5 algorithms, in its MPR section, standard OLSR emphasizes on reducing the number of MPRs in the network so as to lower the overhead. It has the lowest MPR number and overhead compared with OLSR_R2, OLSR_R3 and Pure Link State Algorithm. (OLSR_R1 has almost the same MPR selection mechanism as that of standard OLSR, and these two algorithms therefore have comparable overheads.) Also, as predicted in Section 4, OLSR_R2 and OLSR_R3 select more MPRs, thus produce higher overhead than standard OLSR. Compared with OLSR_R2, OLSR_R3's overhead is even higher, which is also consistent with our prediction. For Pure Link State algorithm, it obviously has the highest overhead, with each node acting as MPR, re-transmitting the messages it receives.

The result of all 5 algorithms in networks with a transmission range of 200 M and 100 M network have similar characteristic as the 300 M transmission range case.

Table 5. Network characteristics

Transmission range	300M	200M	100M
1-hop neighbors	21	10	2
2-hop neighbors	33	15	4

Table 6. Summary of simulation results

Algorithm	Transmission Range	Performance		Cost	
		Error Rate	Average Difference	Overhead	MPR Number
Standard OLSR	300 M	28%	46%	12	65
	200 M	41%	51%	24	68
	100 M	12%	45%	5	42
OLSR_R1	300 M	14%	22%	12	65
	200 M	21%	26%	24	68
	100 M	8%	44%	5	42
OLSR_R2	300 M	0%	0%	18	70
	200 M	0%	0%	33	72
	100 M	0%	0%	5.7	45
OLSR_R3	300 M	0%	0%	26	71
	200 M	0%	0%	38	73
	100 M	0%	0%	5.7	44
Pure Link State Algorithm	300 M	0%	0%	1245	100
	200 M	0%	0%	979	100
	100 M	0%	0%	28	100

We also explored the performance of the individual algorithms.

Standard OLSR: At first glance, it may seem strange that a network with a node transmission range of 200 M has the highest overhead. Intuitively, the denser the network is, the higher the overhead: for the same number of nodes and area size, the network contains more edges if the transmission range of a node is higher (see Table 1). However, the result can be explained as follows: in general, the more MPRs are selected, the higher the overhead. In a higher density network (such as for a node transmission range of 300 M), node connectivity is also

high, so a node may need fewer MPRs to cover its 2-hop neighbors. On the contrary, in lower density networks (such as for a node transmission range of 100 M), because of the lower connectivity, a node may have fewer 2-hop neighbors, therefore it also needs less MPRs. However, the transmission range of 200 M falls within these two extremes, so it may well result in the largest number of MPRs to produce the highest overhead. This situation is not found in the Pure Link State Algorithm, where a node's entire neighbor set is its MPR set. So the denser the network is, the more neighbors/MPRs a node has, resulting in a higher overhead.

Also, one may expect that the denser the network is, the worse the performance should be. With higher connectivity, there are more possible routes from a given source to a destination, and the probability that OLSR chooses a non-optimal route is higher. This tendency can be seen when comparing the performance of 300 M and 100 M transmission range networks. But again the 200 M transmission range network is the exception, having the highest "Error Rate". Considering a node in an optimal bandwidth route, its next hop node on the path is its 1-hop neighbor, and the hop after next is its 2-hop neighbor (proof is given in Section 6). In other words, an optimal bandwidth path is composed of segments "node->1-hop neighbor->2-hop neighbor". The route computed by OLSR has that feature as well. For 100 M transmission range, because of it's lower connectivity, the node has less 1-hop neigbhors and 2-hop neighbors. As a result, in this network, there are fewer segments of "node->1-hop neighbor->2-hop neighbor", resulting in a lower propability that OLSR chooses the wrong path. For the dense network (300 M transmission range), a node has many more 1-hop and 2-hop neighbors, resulting in many segments of "node->1-hop neighbor ->2-hop neighbor". The selected MPRs will cover many of the 2-hop neighbours more than once, again resulting in a lower propability for OLSR ignoring the segments belonging to the optimal path. As shown by the difference between OLSR and OLSR_R1, a simple change in how to calculate the paths, based on the same MPR set, can yield significant performance improvements. Again, the 200 M transmission range case falls between these two extremes, resulting in the worst performance.

OLSR_R1: the result shows the same trends as that of standard OLSR. Also, when comparing the performance of standard OLSR and OLSR_R1, it shows that OLSR_R1 achieves larger improvements over standard OLSR in higher density network. That is because for higher density networks, more links are declared to a node. So when computing its routing table, a node has more choice in path selection. Standard OLSR uses the Shortest Path Algorithm for route computation, which is unsuitable for bandwidth QoS routing. So the probability that the standard OLSR picks up a non-optimal path is higher in denser networks.

OLSR_R2 and OLSR_R3: For performance, they both find the optimal path. For the cost, they also exhibit the phenomenon that a 200 M transmission range network has the highest MPR number/overhead. The reason is the same as the one explained above for standard OLSR.

Pure Link State Algorithm: Compared with standard OLSR, we find that the higher the network density, the more obvious the overhead reduction. This is consistent with the declaration in [3] that the denser the network is, the more optimization OLSR will achieve, compared to the Link State Algorithm.

6. Correctness of the revised OLSR algorithms

From the simulation, we find that under the current simulation model, both OLSR_R2 and OLSR_R3 find the optimal path. Actually, these two algorithms do guarantee the optimal result. Following is the proof:

Theorem 2: OLSR_R2 finds the optimal bandwidth path.

(In the proof, 1-h=1-hop-neighbor, 2-h=2-hop-neighbor)

LEMMA 1: The intermediate nodes on the optimal path (the path with the highest bottleneck bandwidth) are all selected as MPRs by the previous nodes on the path.

Proof: A node in the route may not be selected as the MPR by the previous node if: 1) the node does not provide connection to that node's 2-hop neighbors and 2) the node does not meet the MPR selection criteria. In the following proof, we address these two situations separately.

1) If a node has no connection to its previous node's 2-hop neighbors, it will not be included on the optimal path.

Proof: In the following graph, node d only connects to node a's 1-hop neighbor. Considering the two possible paths from a to c: a->b->c and a->d->b->c.

Suppose node a, b, c, d's idle time are I_a, I_b, I_c, I_d respectively. Based on the network model defined in Section 5.1, the available link bandwidth of link ab, bc, ad, db are:

Link ab: $\min(I_a, I_b)$, Link bc: $\min(I_b, I_c)$

Link ad: $\min(I_a, I_d)$, Link bd: $\min(I_b, I_d)$

Thus, the bottleneck bandwidth of path adbc is: $\min(ab, bc, ad, bd)=\min(I_a, I_b, I_c, I_d)$; The bottleneck bandwidth of path abc is: $\min(ab, bc)=\min(I_a, I_b, I_c)$

Clearly that path abc has the same or better bandwidth path because: $\min(I_a, I_b, I_c)\geq\min(I_a, I_b, I_c, I_d)$

=>path "node->1-h->2-h" has the same/better bottleneck bandwidth than the path "node->1-h->1-h->2-h".

=>path with the segment "node ->1-h->1-h-> 2-h" is no better than the path with the segment "node ->1-h-> 2-h".

=>If a node has no connection to its neighbors' 2-hop neighbors, it is not on the optimal path.

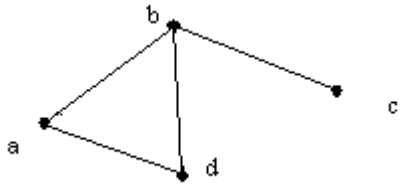

Figure 4: a's 1-hop neighbor and 2-hop neighbor information

2) There is an optimal path from source to destination such that all the intermediate nodes on the path are selected as MPR by their previous nodes on the same path. Without loss of generality, we suppose that in an optimal path, S, M_1, M_2…M_k, M_{k+1},…M_r, D, there are nodes in the route which are not selected as MPRs by their previous nodes. Also, based on the result of 1), we can assume that for each node on the path, its next node on the path is its 1-h, and the node two hops away from it is its 2-h. For example, M_1 is S's 1-h, M_2 is S's 2-h. M_{k+1} is M_k's 1-h, M_{k+2} is M_k's 2-h, etc.

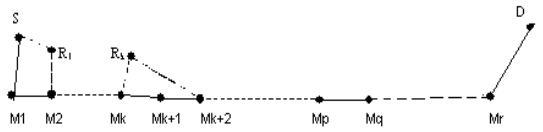

Figure 5: route from source S to destination D

a) Suppose that on the optimal route, the first intermediate node M_1 is not selected as MPR by source S. However, M_2 is the 2-hop neighbor of S. Based on the basic idea of MPR selection that all the 2-hop neighbors of a node should be covered by this node's MPR set, S must have another neighbor R_1, which is selected as its MPR, and is connected to M_2. According to the criteria of MPR selection specified in OLSR_R2, S selects R_1 instead of M_1 as its MPR because the link bandwidth of SR_1 is better than the link bandwidth of SM_1, which means Ir_1 (idle time of node R_1) is larger than or equal to Im_1 (idle time of node M_1).

Define bottleneck bandwidth of route R as B(R).

$B(S->R_1->M_2->\ldots->M_r->D)$
$= \min(B(S->R_1->M_2), B(M_2->\ldots->D))$
$= \min(\min(Is, Ir_1, Im_2), B(M_2->\ldots->D))$
$B(S->M_1->M_2->\ldots->D)$
$= \min(\min(Is, Im_1, Im_2), B(M_2->\ldots->D))$
As $Ir_1 \geq Im_1$, $\min(Is, Ir_1, Im_2) \geq \min(Is, Im_1, Im_2)$
$=>B(S->R_1->M_2->\ldots->M_r->D) \geq B(S->M_1->\ldots->D)$.
As, route $S->M_1->M_2->\ldots->D$ is optimal path
=> $S->R_1->M_2->\ldots->D$ is also an optimal path
=> Source's MPR are on the optimal path.

b) Assume that on the optimal route $S->M_1->\ldots->M_k->\ldots->D$, all the nodes on segment $M_1->M_k$ are selected as MPR by their previous node, we now prove that the next hop node of M_k on the optimal route is M_k's MPR.

Suppose that M_{k+1} is not M_k's MPR. Same as above, M_{k+2} is the 2-hop neighbor of M_k, so M_k must has another neighbor R_k, which is the MPR of M_k and has connection to M_{k+2}.

Again, M_k selects R_k instead of M_{k+1} as its MPR because link bandwidth M_kR_k is better than M_kM_{k+1}, which means Ir_k (idle time of node R_k) is better than Im_{k+1} (idle time of node M_{k+1}).

$B(S->\ldots M_k->M_{k+1}->M_{k+2}->\ldots M_r->D)$
$= \min(B(S->M_k), \min(Im_k, Im_{k+1}, Im_{k+2}), B(M_{k+2}->D))$
$B(S->\ldots M_k->R_k->M_{k+2}\ldots->D)$
$= \min(B(S->M_k), \min(Im_k, Ir_k, Im_{k+2}), B(M_{k+2}->D))$
$\geq B(S->\ldots M_k->M_{k+1}->M_{k+2}->\ldots M_r->D)$
As $S->\ldots->M_k->M_{K+1}->M_{k+2}->\ldots->D$ is optimal route
$=>S->\ldots->M_k->R_k->M_{k+2}\ldots->D$ is also optimal route.
=>In an optimal route, the (k+1)th intermediate node is the MPR of the (k)th intermediate node.

Based on a) and b), all the intermediate nodes of an optimal path are the MPRs of the previous nodes.

LEMMA 2: A node can correctly compute the optimal path for the whole network topology.

Proof: 1) as shown by **Theorem 1**, using a "maximum bandwidth spanning tree" algorithm, a node can compute the optimal path on the known partial network topology
2) In OLSR, each node knows the links between MPRs and their selectors. Based on LEMMA 1, there is an optimal path such that all the intermediate nodes on it are an MPR for of the previous node on the same path. So the optimal path for the whole network topology is included in the partial topology the node knows.
=> The node can correctly compute the optimal path for the whole network topology.
Based on LEMMA 1 and LEMMA 2, OLSR_R2 finds the optimal path.

Theorem 3: OLSR_R3 finds the optimal bandwidth path.

The proof is similar to that of Theorem 2.

7. Conclusion and future work

In this paper, we describe the importance of QoS routing in ad-hoc networks, the challenges we meet, and the approach we take. We discuss in detail our idea of adding support for QoS into OLSR, our three heuristics that allow OLSR to find the maximum bandwidth path, and show initial simulation results of these algorithms under a number of network snapshots. From a

performance perspective, all three heuristic increase the odds of finding a path that is optimal under a bandwidth constraint. Also, we prove that for our ad-hoc model, two of the heuristics (OLSR_R2 and OLSR_R3) are indeed optimal.

Our current simulation model is based on static network snapshots. We are currently adding our heuristics to an OLSR simulation based on OPNET. We will explore the impact of node movement and bandwidth change. The two optimal algorithms will be run under these conditions to evaluate their performance under dynamic topology and link bandwidth changes. Also, in the current simulations, we not only compare the performance of the algorithms, but also show their advantages and limits with respect to costs – the overhead each algorithm generates. From the result, we see that between the two optimal heuristics, OLSR_R2 has fewer overheads than OLSR_R3. Also, again compared to OLSR_R3, OLSR_R2 is simpler and more straightforward; we are very interested in the further comparison of these two algorithms in a mobile and dynamic environment.

Acknowledgements

This work is partly funded by the Defense R&D Canada - Ottawa. The authors thank the CRC RNS mobile networking group for their contributions.

REFERENCES

[1] 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

[2] 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

[3] 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

[4] Z. Wang and J. Crowcroft, "Quality-of-Service Routing for Supporting Multimedia Applications", IEEE Journal On Selected Areas In Communications, Vol. 14. No. 7, September 1996, pages 1228-1234

[5] C.R. Lin and J.S. Lui, "QoS Routing in Ad-Hoc Wireless Networks", IEEE Journal On Selected Areas In Communications, Vol. 17, No. 8, August 1999, pages 1426-1438

[6] G.S. Ahn, A.T. Campbell, A. Veres and L.H. Sun, "SWAN: Service Differentiation in Stateless Wireless Ad Hoc Networks", Proc. IEEE Infocom 2002, New York, New York, 2002 (to appear)