

Load-Sensitive Routing for Mobile Ad Hoc Networks*

Kui Wu, Janelle Harms
Computing Science Department
University of Alberta, AB, Canada T6G 2E8

Abstract— A Mobile Ad hoc NETWORK (MANET) is a collection of wireless mobile computers forming a temporary network with no existing wired infrastructure. Due to the dynamic nature of network topology and the resource constraints, routing in MANETs is a challenging task. Distributing the routing tasks fairly has eminent advantages, such as reducing the possibility of power depletion and queuing delay in the hosts with heavy duties. However, most current routing protocols for MANETs do not take load balancing into account. In this paper, we propose a load-sensitive on-demand routing approach, which utilizes the network load information as the main route selection criterion. We perform a simulation study on the proposed routing protocol. Compared with the Dynamic Source Routing (DSR), our protocol shows better performance in terms of packet delivery ratio and average end-to-end delay. Further, with low mobility, these benefits are gained without an increase in the control overhead.

I. INTRODUCTION

A Mobile Ad hoc NETWORK (MANET) is a collection of wireless mobile computers forming a temporary network, which is based on radio to radio multi-hopping and has neither fixed base stations nor a wired backbone infrastructure. Routing in MANETs is a non-trivial task because hosts' movements cause frequent topology changes and require robust and flexible mechanisms to discover and maintain the routes. Due to the power and bandwidth limitations, a routing protocol in MANETs should fairly distribute the routing tasks among the mobile hosts. An unbalanced assignment of data traffic will lead to power depletion on heavily loaded hosts. With more hosts powered down, the connectivity of the network will be reduced, which will lead to call failures due to network partitions. In addition, the nodes with heavy routing duties likely have large queuing delay and high packet loss ratio. Therefore, the end-to-end delay and packet loss ratio are large for the connections using those nodes.

Routing protocols in MANETs could be mainly divided into proactive and reactive methods. Proactive methods maintain routes to all nodes, including nodes to which no packets are sent. They react to topology changes, even if no traffic is affected by the changes. They are also called table-driven methods. On the other hand, reactive methods are based on demand for data transmission. Routes between hosts are determined only when explicitly needed to route packets. It is not necessary to periodically update route information or to find and maintain the routes on which there is no data traffic. Reactive methods are also called on-demand methods.

Although many routing protocols have been proposed for MANETs, most of them have not taken the load information into account. M. R. Pearlman and Z. J. Haas

[2] prove that the multipath routing approach could balance the network load, but they do not use network load information in their method. In [3], S. R. Das et al. conclude that using congestion-related metrics, such as queue length, could improve the routing performance in MANETs. However, they do not point out how to utilize this information in their paper. In [4], H. S. Hassanein and A. Zhou utilize load information as the main path selection metric for routing in MANETs. They define the network load in a node as the total number of routes passing the node and its neighboring nodes. This load metric may not be accurate since the traffic along different paths may not be the same. In their method, each node periodically exchanges load information with its neighbors, and each middle node will broadcast all received route request messages. In [8], S. J. Lee and M. Gerla propose a load-aware routing method. In the network load metric, they define the network load of a node as the number of packets queuing in the interface of the node. They utilize a route adaptation strategy to search for better paths.

In this paper, we propose a Load-Sensitive Routing (LSR) method that utilizes network load information as the main path selection criterion. In LSR, the network load in a node not only includes the local load, but also considers the load in the nodes' neighboring nodes. This metric is more accurate than the one used in [8] due to the broadcast feature of radio channels, which creates contention. The way to obtain network load information in LSR does not require periodic exchanges of load information among neighboring nodes and is suitable for any existing routing protocol. We also provide a unified scheme to utilize the load information and to tune the performance of LSR. Unlike [4] and [8], LSR does not require the destination nodes to wait for all possible routes. Instead, it uses a re-direction method to find better paths effectively. The source node can quickly respond to a call for connection without losing the chance to obtain the best path. Based on the initial status of an active path, LSR can search dynamically for better paths if the active path becomes congested during data transmission. The simulation results show that LSR can improve the performance of DSR, an on-demand protocol that does not utilize network load information in routing.

II. LOAD-SENSITIVE ROUTING (LSR)

A. Background

A.1 Dynamic Source Routing (DSR)

In the Dynamic Source Routing (DSR) [1], if a source node does not know a route to a destination, it will initiate a route discovery by flooding a Route REquest (RREQ) message. The RREQ message carries the sequence of hops it passed through in the message header. When a node receives a RREQ, if it is the first time for the node to receive this RREQ message, this node will broadcast it again. Otherwise, the node will drop this RREQ packet.

* This research was supported by a grant from the Canadian Institute for Telecommunications Research under the NCE program of the Government of Canada.

Once a RREQ message reaches the destination node, the destination node will reply with a Route REPLY (RREP) packet to the source, using the reverse path contained in the RREQ packet. When the RREP packet traverses backward to the source, the source and all traversed nodes will know a route to the destination. DSR then uses the source routing method, that is, it includes the whole path information in the packet header, to transmit packets.

DSR uses a route cache mechanism to help a node obtain more route information and to reduce control overhead. In DSR, an intermediate node can reply to a RREQ if it has a path to the destination in its own route cache, and can use an alternate route from its own route cache when a data packet can not be transmitted to its next hop according to its source route. When a node overhears a packet not addressed to itself, it can learn and record more route information. Furthermore, it can decide whether this packet can be routed via itself to obtain a shorter path. If this is possible, the node will notify the source node to use the shorter path. When a link breakage is detected, an error message is sent back to the source node. Any node that receives the error message will erase the routes that use the broken link from its route cache.

A.2 Some Definitions

In [6], we demonstrate that traffic along nearby paths could interfere with each other and thus influence the routing performance. The network load in a mobile host depends on the traffic passing through this host as well as the traffic in the neighboring hosts.

Definition 1 Traffic load in a mobile host is defined as the sum of the number of packets queuing in the interface of the mobile host and the number of packets queuing in the interfaces of the host's neighboring hosts.

Definition 2 A path P between node v_1 and node v_n is an ordered sequence of distinct nodes $\langle v_1, \dots, v_n \rangle$ such that (v_{i-1}, v_i) is a link for all $2 \leq i \leq n$. Node v_i is called an *internal node* if $1 < i < n$. There is a link from v_{i-1} to v_i if node v_i is within the transmission range of node v_{i-1} . In this paper, we assume all mobile nodes' transmission ranges are the same.

A host can easily obtain the traffic load information if this host listens promiscuously to neighboring nodes' transmissions. In our Load-Sensitive Routing (LSR) method, every packet includes Local Load (LL) and path load information which includes total Path Load (PL) and the standard deviation of path load (PV) for the path so far. The local load (LL) includes the number of packets that are queued in the current host. Every host maintains a value of Traffic Load (TL). The value of the TL in a node is calculated as the sum of LL of this node and the newest LL s received from neighboring nodes. Obsolete LL values will be removed after a timeout period. The total Path Load (PL) is calculated as the total sum of TL in all hosts along the path. The standard deviation of path load, which is denoted as PV , is defined as the standard deviation of TL in every host along the path. The total Path Load (PL) represents the total traffic along a path and nearby nodes, while the standard deviation of path load (PV) represents whether the traffic is distributed evenly.

Remark 1 For a given path $P = \langle v_1, \dots, v_n \rangle$, if we assume $PL(P)$ denotes the total path load of P and $TL(v_i)$ denotes the traffic load in node v_i ($1 \leq i \leq n$), then

$$PL(P) = \sum_{i=1}^n TL(v_i) \quad (1)$$

Remark 2 For a given path $P = \langle v_1, \dots, v_n \rangle$, if we assume $PV(P)$ denotes the standard deviation of traffic load in the nodes along the path P and $TL(v_i)$ denotes the traffic load in node v_i ($1 \leq i \leq n$), then

$$PV(P) = \sqrt{\frac{\sum_{i=1}^n (TL(v_i))^2 - n * (ML)^2}{n - 1}}$$

$$\text{where } ML = \frac{PL(P)}{n} \quad (2)$$

Remark 3 Using Formula (2), we can calculate $PV(P)$ incrementally from PL and hop count so far, and do not need to keep every value of $TL(v_i)$ in the packet header. For example, when v_i receives a packet from v_{i-1} , if we denote $P_1 = \langle v_1, \dots, v_{i-1} \rangle$, then this packet should carry the values of $PL(P_1)$ and $PV(P_1)$ as described above. Let $P = \langle v_1, \dots, v_i \rangle$, then $PL(P) = PL(P_1) + TL(v_i)$. From the Formula (2), we can solve the equation to obtain $\sum_{i=1}^{n-1} (TL(v_i))^2$ first, then calculate $\sum_{i=1}^n (TL(v_i))^2$, and finally get $PV(P)$.

A path comparison function is used to select a better path.

Definition 3 Path comparison function $f(A, B)$, where A, B are two paths, is defined as:

$$f(A, B) = \begin{cases} -1 & \text{if } (PL(A) - PL(B) > \alpha) \vee \\ & ((|PL(A) - PL(B)| \leq \alpha) \wedge \\ & (PV(A) - PV(B) > \beta)), \\ 0 & \text{if } (|PL(A) - PL(B)| \leq \alpha) \wedge \\ & (|PV(A) - PV(B)| \leq \beta), \\ 1 & \text{otherwise.} \end{cases}$$

In the definition of f , we use total path load (PL) as the main comparison criterion and the standard deviation of path load (PV) as an auxiliary criterion. If the total path load of path A is at least α larger than path B , or the difference of total path load of path A and B is not larger than α but the standard deviation of path load of path A is at least β larger than B , then we think path B is better than path A . $f(A, B) = -1$ represents this case. When $f(A, B) = 0$, we consider that path B and path A are almost the same in terms of path load information. $f(A, B) = 1$ represents that path A is better than path B . The two parameters, α and β , are used to make the comparison flexible.

Based on different requirements, we use two sets of path comparison functions: strict and loose path comparison functions. In the strict one, the parameters α and β are set very small, while in the loose one, α and β are relatively larger.

B. Overview of LSR

B.1 Route Discovery

Like DSR [1], LSR is an on-demand routing protocol. Unlike [4] and [8], we use a re-direction method similar to that we developed in [9] to forward Route REPLY (RREP) messages. This method can let the source node obtain better paths without an increase of flooding cost and waiting delay in the destination nodes.

In LSR, if a source node does not know a route to a destination, it will initiate a route discovery by flooding a Route REQuest (RREQ) message. The RREQ message carries the source and destination addresses, a sequence number which is initialized to 0 and increased by 1 for the next route discovery, the sequence of hops which it passed through, and the local load and path load information. The path load information includes two values, PL and PV , and is calculated according to Formula (1) and (2). When a node receives a RREQ, if it is the first time for the node to receive this RREQ message, this node will broadcast it again. Otherwise, the node will cache but not broadcast this RREQ packet. Two RREQ packets are assumed to be the same if they include the same source and destination addresses and same sequence number.

Once a RREQ message reaches the destination node, the destination node will reply with a Route REply (RREP) packet to the source, using the reverse path contained in the RREQ packet. Since the destination does not wait for all possible routes, the source node can quickly obtain the route information and it quickly responds to calls for connections. Similar to DSR [1], LSR lets the destination reply to every received RREQ message in order to provide tolerance for broken links on the return trip and more routing choices for the source node. The RREP packet will re-calculate the path load information when it traverses back to the source node. When a node receives a RREP packet, the remaining path in the RREP to the source node is replaced with a cached RREQ that has the best path load information and does not form a loop. The best path is chosen using a strict path comparison function f . If several best paths exist, choose one randomly.

When the source node first receives a RREP packet, it immediately uses the path included in this RREP packet to forward packets. In the meantime, the source node will record the path load information included in this RREP packet. If a later-received RREP packet includes better path load information, then the source node transfers the traffic to the newly obtained path and updates the path load information.

In the route discovery phase, we use a strict path comparison function to compare the load information of two paths. Since the parameters α and β are set small in a strict path comparison function, we can guarantee that a strictly better path will be chosen.

B.2 Route Adaptation and Maintenance

In [7], we dynamically select a more robust path before the used one breaks. The criterion of route adaptation in [7] is based on link lifetime. In LSR, we adapt the active routes in a different context, by using network load information. When a used path becomes congested, LSR tries to search for a lightweight path. Note that the source node continues to send data traffic along the congested path until a better path is found. In [8], S. J. Lee and M. Gerla also utilize a route adaptation method. The criterion for route adaptation in [8] is based on fixed threshold values. Unlike [8], our route adaptation strategy is based on the initial status and current status of an active path.

When a data packet is sent from the source to the destination, it will calculate the current path load. The initial path load information obtained by the first data packet along a selected path will be recorded in the destination

node. When current path load information is much worse than the initial path load information, network congestion may have occurred and a new better path should be found. In order to check whether the current path has become much worse, a loose path comparison function is used. The parameters α and β , in this case, are threshold values for the needs of route adaptation. As we will show later, if the parameters α and β are too small, too much unnecessary route adaptation will be activated.

If the current path becomes much worse than its initial status, which is calculated based on the first data packet, the destination node will initiate a route adaptation. It broadcasts a Route REQuest message to search for a route to the source node. This RREQ message will calculate the path load, using the same method as in the route discovery stage. It also carries the path load information obtained from the latest data packet received at the destination. Once the source node receives a RREQ, if the path load calculated by this RREQ is much better than the path load of the currently-used path, which is included in the RREQ, the source node transfers its data traffic to the path included in this RREQ packet. Note that the source node does not send back a RREP packet, and the source node uses the same loose path comparison function to check whether the received path in the RREQ packet is much better than the currently-used one.

When an established path breaks due to mobility, an error message is sent back to the source node and the source node activates a route discovery as described above.

Since a strict path comparison function is used in the route discovery phase while a loose one is used in the route adaptation phase, for the convenience of later discussion and without loss of generality, we call the path comparison function used in the route discovery phase a strict path comparison function, and call the one used in the route adaptation phase a loose path comparison function even if we may deliberately set the parameters of it to small values for the purpose of a simulation study.

III. SIMULATION MODEL

We use a simulation model based on GloMoSim [5] to study the performance of LSR. We compare the performance of LSR and the Dynamic Source Routing (DSR) [1], which is an on-demand routing protocol but does not take network load information into account. We also study how the path comparison functions influence the protocol performance. Furthermore, we evaluate the performance of a variant of LSR, which is called LSR-2 and whose promiscuous listening mode is disabled. Like [8], LSR-2 defines the network load in a node as the total number of packets queuing in the node's interface, because a node can not know its neighboring nodes' local load information if the promiscuous listening mode is disabled.

In our simulation, the channel capacity of mobile hosts is set to the same value: 2 Mbps. We assume all nodes have the same transmission range of 250 meters. A free space propagation model with a threshold cutoff is used as the channel model. In the free space model, the power of a signal attenuates as $1/r^2$, where r is the distance between mobile hosts. In the radio model, capture effects are taken into account. We use the Distributed Coordination Function (DCF) of IEEE 802.11 for wireless LANs as the MAC layer protocol. It has the functionality to notify the network layer about link failures.

In our simulation, 50 mobile nodes move in a 1500 meter x 500 meter rectangular region for 900 seconds simulation time. Compared with a square region, the rectangular region can enlarge the average route length so that we can easily observe the performance difference in different scenarios. We assume each node moves independently with the same average speed. The mobility model is the random waypoint model. In this mobility model, a node randomly selects a destination from the physical terrain. It moves in the direction of the destination in a speed uniformly chosen between the minimal speed and maximal speed. After it reaches its destination, the node stays there for a pause time and then moves again. In our simulation, the minimal speed is 1 m/s and the maximal speed is 20 m/s. We change the pause time from 0 seconds to 900 seconds to investigate the performance influence of different mobility. A pause time of 0 seconds presents continuous motion, and a pause time of 900 seconds corresponds to no motion.

The simulated traffic is Constant Bit Rate (CBR). The size of all data packets is 512 bytes. A packet is dropped when no acknowledgement is received after several re-transmissions or when there is no buffer to hold the packet. The buffer size is 128 packets. All traffic is generated and the statistical data are collected after a warm-up time of 30 seconds in order to give the nodes sufficient time to finish the initialization process. For each scenario, eight runs with different random seeds were conducted and the results were averaged. When calculating the confidence intervals, the confidence levels are set to 95%.

We evaluate the performance according to the following metrics:

- **Packet delivery ratio:** The packet delivery ratio is defined as the ratio of the total number of data packets received by destinations over the total number of data packets transmitted by sources.
- **Average end-to-end delay:** The end-to-end-delay is averaged over all surviving data packets from the sources to the destinations.
- **Control overhead:** The control overhead is defined as the total number of routing control packets normalized by the total number of received data packets.

IV. SIMULATION RESULTS

As we will show later, the parameters α and β in the strict path comparison function should be set to small values, but they should not be set to small values in the loose path comparison function. According to this principle, in the first experiment, we set the parameter α and β to 5 and 2 respectively in the strict path comparison function and to 200 and 10 respectively in the loose path comparison function. Three types of traffic load are simulated. In the first scenario, 10 source-destination pairs send CBR traffic with an interval time of 200ms and 10 randomly chosen nodes periodically broadcast background traffic to their neighboring nodes with an interval time uniformly distributed from 50ms to 100ms. In this scenario, the connections have medium traffic load, but the randomly chosen nodes have heavy traffic burden and are likely to become “hot spots” in the networks. In the second scenario, 10 source-destination pairs send CBR traffic with an interval time of 250ms. The traffic is lightweight in this case. In the third scenario, 15 source-destination pairs send CBR traffic with an interval time of 100ms. The

traffic is heavy along the established paths.

A. Packet Delivery Ratio

Fig. 1, 2, and 3 show the results of packet delivery ratio of LSR and DSR for the above different traffic loads. In all simulated scenarios, the packet delivery ratio of LSR is better than that of DSR. Two reasons can explain this phenomenon. First, LSR uses the strict path comparison function to choose a path with light traffic. This will reduce packet loss due to buffer overflow in the congested nodes. Second, DSR aggressively uses the route cache mechanism [3], which may use stale and wrong route information and results in packet loss. In the light load scenario, however, the improvement of packet delivery ratio of LSR is relatively small, compared with the results in “hot spots” and heavy load scenarios. This is because the traffic is light and there is little chance to form congested nodes. The advantage of LSR using lightweight routes is not very obvious in this case. In both “hot spots” and heavy load scenarios, the performance of packet delivery ratio in LSR is much better than in DSR. The reason is straightforward since LSR will keep a path away from those heavily loaded nodes whenever a lightweight path exists. In contrast, DSR does not consider the load information in path selection and always chooses a shortest path that may include congested nodes.

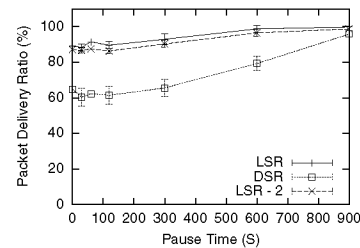


Fig. 1. 10 sources with background traffic

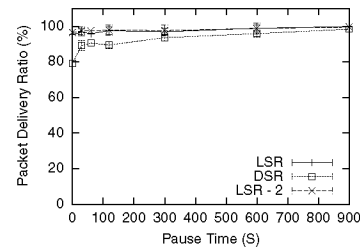


Fig. 2. 10 sources with light traffic

With the increase of mobility, that is, the decrease of pause time, the packet delivery ratio for both LSR and DSR goes down. This is because paths are more likely to break with high mobility. A very interesting phenomenon is that the packet delivery ratio in LSR is more consistent and resilient to mobility. The route adaptation mechanism in LSR may contribute to this feature. With route adaptation, data traffic is likely to be transferred to a new path before the used one fails.

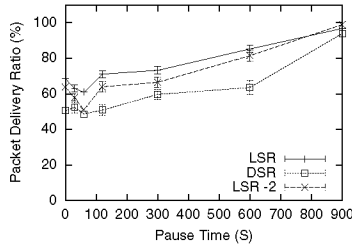


Fig. 3. 15 sources with heavy traffic

Under light traffic, LSR and LSR-2 have no obvious difference. However, when the network includes some “hot spots” or when the end-to-end traffic is heavy, LSR outperforms LSR-2. This is because LSR chooses lightweight paths and the neighboring nodes along the paths are also likely to have light traffic. In contrast, LSR-2 can choose a light loaded path, whose neighboring nodes, however, may have heavy traffic. In this case, the heavily loaded neighboring nodes will contend for the transmission channel and cause network congestion. However, with the help of partial load information, LSR-2 still outperforms DSR.

B. Average End-to-End Delay

Fig. 4, 5, and 6 show the results of average end-to-end delay of LSR and DSR under different traffic loads. In the “hot spots” scenario, the average end-to-end delay of LSR is better than that of DSR. This is because LSR chooses a path with light traffic load and thus reduces the queuing delay in the network interfaces. The end-to-end delay will be increased if a path includes nodes with heavy background traffic. This situation may occur in DSR but is less likely to happen in LSR, since LSR has mechanisms to choose lightweight paths and can adjust traffic dynamically from a congested path to a less-loaded path.

However, when traffic load is light, LSR and DSR have little difference. The reason is that light network traffic does not cause network congestion and network load information has little meaning in this context. Another strange phenomenon is that in the heavy load scenario, the end-to-end delay of LSR is slightly larger than that of DSR. Although LSR can dynamically search for a lightweight path, the nodes along the newly established path will be quickly congested since the end-to-end traffic is bursty and heavy. In this extremely bad situation, LSR does not help too much and may make things worse. A newly found path in LSR usually has more hops and the end-to-end delay will be increased. DSR, however, keeps transmitting messages along the shortest path at the cost of more packet losses in the heavy load scenario.

In both “hot spots” and heavy load scenarios, LSR performs better than LSR-2 in terms of average end-to-end delay. The reason is straightforward since LSR-2 may choose a path whose neighboring nodes are heavily loaded. The interference of radio transmissions will increase the end-to-end delay. This phenomenon is also observed in our work in [6].

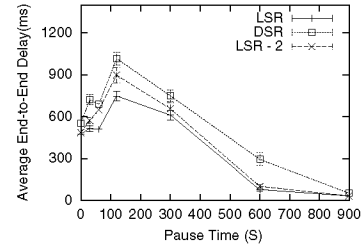


Fig. 4. 10 sources with background traffic

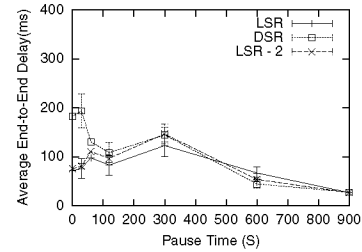


Fig. 5. 10 sources with light traffic

C. Control Overhead

Fig. 7, 8, and 9 show the results of control overhead. With high mobility, DSR has smaller control overhead than LSR. But with low mobility, the control overhead for DSR and LSR exhibits little difference. DSR uses route cache and allows middle nodes to respond to route requests. This will quench the broadcast of route request messages earlier and reduces control overhead. The disadvantage of route caching is that too many route replies will be sent out and obsolete route information may be used. From the simulation results, however, we can see that the route-caching strategy in DSR reduces the control overhead with high mobility because more route requests will be promoted and most route requests can be quenched and replied to the source by middle nodes. On the other hand, LSR does not allow middle nodes to reply to route requests. In addition, in LSR, the route adaptation mechanism to search dynamically for better paths during data transmission will also increase control overhead. Nevertheless, we think the control overhead of LSR is acceptable since LSR can maintain a higher packet delivery ratio than DSR. The higher control overhead in LSR does not influence the performance too much, particularly

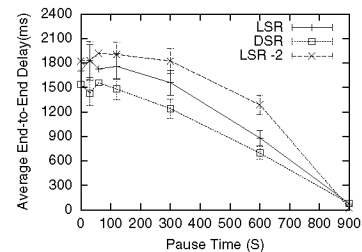


Fig. 6. 15 sources with heavy traffic

for the packet delivery ratio.

LSR and LSR-2 have similar control overhead because enabling the promiscuous listening mode in LSR does not require additional transmissions of control packets.

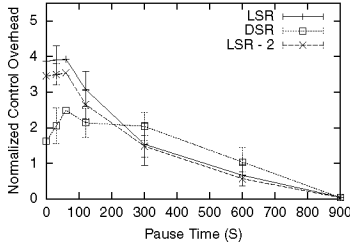


Fig. 7. 10 sources with background traffic

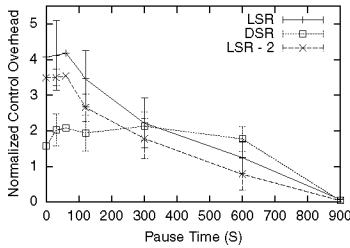


Fig. 8. 10 sources with light traffic

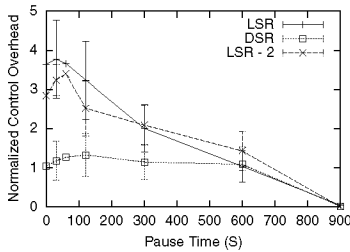


Fig. 9. 15 sources with heavy traffic

D. The Influence of Path Comparison Functions

The path comparison functions in LSR provide us a unified way to select paths using different load information. Two metrics, total path load and standard deviation of path load, are used in the path comparison functions. Since a strict path comparison function is used in the route discovery phase, setting large numbers to the parameters in the strict path comparison function makes no sense (otherwise, the path selection method will be the same as in DSR). Thus we will mainly consider how to set the parameters in the loose path comparison function for route adaptation.

We will study several schemes to use load information. These schemes include: (1) using the total path load information only, (2) using the standard deviation of path load information only, (3) adapting routes frequently, (4) disabling route adaptation, and (5) properly adapting routes

with the help of both total path load and standard deviation of path load information.

As shown in Table I, five groups of values are studied. In Group A, the path comparison functions select a better path based only on the total path load information since β is set to a very large number and plays almost no role in the path comparison. In Group B, the path comparison functions select a better path based only on the standard deviation of path load since α is very large. In Group C, the parameters in the loose path comparison function are set very small. In this case, a small difference in the traffic load information may cause route adaptation. The parameters in the loose path comparison function are set very large in Group D. In this case, LSR may not detect congestion during data transmission. In other words, much fewer route adaptations will be activated. In Group E, the parameters of the path comparison functions are set the same as in the previous experiment. In the simulated traffic model, 15 source-destination pairs send CBR traffic with an interval time of 250ms and 10 randomly chosen nodes periodically broadcast background traffic to their neighboring nodes with an interval time uniformly distributed from 50ms to 100ms.

TABLE I
PATH COMPARISON FUNCTIONS WITH DIFFERENT PARAMETERS

Group	Strict f		loose f	
	α	β	α	β
A	5	3000	200	3000
B	3000	2	3000	10
C	5	2	5	2
D	5	2	3000	3000
E	5	2	200	10

Fig. 10 shows the results of packet delivery ratio. Group C has the lowest packet delivery ratio. In Group C, too much route adaptation will be aroused unnecessarily. A large amount of control traffic will cause network congestion and result in more packet losses. In Fig. 10, although Group E has the highest packet delivery ratio, the performance difference among Groups A, B, D, and E is not very obvious. The packet delivery ratio of Group B is lower than Groups A, D, and E, although the difference may not be significant. In Group B, the role of total path load is almost disabled since a large number is set to the parameter α in both the strict and loose path comparison functions. This means that the total path load information is more important than the standard deviation of path load information. A path with large standard deviation of load may be a good path. For example, there may exist a situation where most nodes along a path are lightly-weighted with only a very small number of the nodes having a heavy load. That is the main reason why we define the total path load information as the main path selection criterion in the path comparison function.

Fig. 11 shows the results of average end-to-end delay. As expected, Group C has the largest average end-to-end delay. In Group C, unnecessary route adaptation may switch the traffic to longer paths and more control packets will contend for the transmission channel. At relatively lower speed (pause time larger than 300S in the simulation), Group A presents similar average end-to-end delay as Group D and E and has better performance than Group

B. This demonstrates that the average end-to-end delay mainly depends on the total path load.

Fig. 12 shows the results of control overhead. Group C has the highest control overhead because it has too much route adaptation. The control overhead of Groups A, B, and E has no large difference, although Group B has the highest control overhead among them. This is because the numbers of route adaptations are not significantly different among these scenarios, although they activate route adaptation according to different criteria. Group D has the smallest control overhead since it has very few route adaptations.

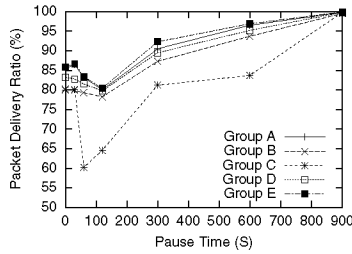


Fig. 10. Packet delivery ratio with different parameters in f

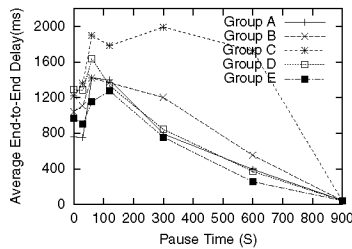


Fig. 11. Average end-to-end delay with different parameters in f

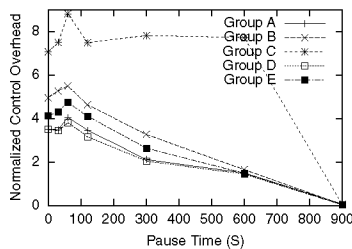


Fig. 12. Control Overhead with different parameters in f

Based on the above observations, two main principles can be applied to setting the parameters in the path comparison functions. First, the parameters α and β in the strict path comparison function should be set always to small values. However, they should not be set to small values in the loose path comparison function. Second, choosing different values can adjust the path selection criterion. For example, setting a very large value to α in the strict and loose path comparison functions will disable the role of total path load information, and setting a

very large value to β in the strict and loose path comparison functions will disable the role of path load deviation information.

Another observation is that the packet delivery ratio and average end-to-end delay of Group E (with route adaptation) is better than Group D (almost without route adaptation). But the control overhead of Group E is worse than group D at high mobility. However, these differences are not large with our simulated traffic.

V. CONCLUSION

In this paper, we proposed a Load-Sensitive Routing (LSR) method for MANETs. Our method utilizes the network load information to assist in routing and adaptively selects lightweight paths during the data transmission. We performed a simulation study on LSR. Compared with DSR, the results show that with the help of load information, LSR can improve packet delivery ratio, and reduce average end-to-end delay when the network includes some congested nodes. Further, the control overhead is acceptable for LSR since it can keep high packet delivery ratio even with high mobility and it has no increase in control overhead with low mobility. We defined the path comparison functions to provide a unified scheme to tune the performance of LSR. In this paper, we also studied the performance influence of the path comparison functions in LSR and provided principles for choosing proper path comparison functions.

ACKNOWLEDGMENTS

Many thanks are due to Dr. Hassanein, who motivated us to investigate this research. We would also like to thank Donnie Chandler for providing useful comments, which greatly improved the quality of the paper.

REFERENCES

- [1] J. Broch, D. Johnson, and D. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad hoc Networks," <http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-04.txt>, Nov. 2000, IETF Internet Draft.
- [2] M.R. Pearlman, Z.J.Hass, P. Sholander, and S.S. Tabrizi, "On the Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Networks," *Proceeding of IEEE/ACM MobiHoc 2000*, Boston, Aug. 2000, pp. 3-10.
- [3] S. R. Das, C. E. Perkins, and E. M. Royer, "Performance Comparison of Two On-Demand Routing Protocols for Ad-Hoc Networks," *Proceedings of IEEE INFOCOM'00*, March 2000, pp. 3-12.
- [4] H. S. Hassanein and A. Zhou, "Routing with Load Balancing in Wireless Ad Hoc Networks," *To appear in the Proceedings of ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Rome, Italy, July 2001.
- [5] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: a Library for Parallel Simulation of Large-Scale Wireless Networks," *Proceedings of PADS'98*, Banff, Canada, May 1998.
- [6] Kui Wu and Janelle Harms, "On-demand Multipath Routing for Mobile Ad Hoc Networks," *Proceedings of 4th European Personal Mobile Communication Conference (EPMCC)*, Vienna, Austria, Feb. 2001.
- [7] Kui Wu and Janelle Harms, "Location Trace Aided Routing in Mobile Ad Hoc Networks," *Proceedings of IEEE ICCCN 2000*, Las Vegas, NV, Oct. 2000.
- [8] S. J. Lee and M. Gerla, "Dynamic Load-Aware Routing in Ad hoc Networks," *To appear in the Proceedings of ICC 2001*, Helsinki, Finland, June 2001.
- [9] Kui Wu and Janelle Harms, "Performance Study of a Multipath Routing Method for Wireless Mobile Ad Hoc Networks," *To appear in the Proceedings of 9th International Symposium on Modeling, Analysis and Simulation (MASCOTS '01)*, Cincinnati, Ohio, Aug. 2001.