Load-Balanced Routing through Virtual Paths: Highly Adaptive and Efficient Routing Scheme for Ad Hoc Wireless Networks

Abdulrahman H. Altalhi **Golden G. Richard, III**

Computer Science Department
University of New Orleans
New Orleans, LA 70148
{aaltalhi, golden} @cs.uno.edu

Abstract

Routing protocols for ad hoc wireless networks consider the path with the minimum number of hops as the optimal path to any given destination. However, this strategy does not balance the traffic load over the network, and may create congested areas. These congested areas greatly degrade the performance of the routing protocols. In this paper, we propose a routing scheme that balances the load over the network by selecting a path based on traffic sizes. We present a simulation study to demonstrate the effectiveness of the proposed scheme.

1. Introduction

Rapid progression in technology for mobile devices, including laptops and handheld computers, and the availability of inexpensive wireless networking hardware has resulted in a large interest in wireless connectivity among mobile users. One approach to providing wireless connectivity is through the formation of Mobile Ad Hoc Networks (MANET) [1]. This approach does not assume the support of any pre-existing infrastructure, but, instead, uses other nodes in the ad hoc network as routers to facilitate message delivery. One of the challenging problems in this type of network is the utilization of an efficient routing process.

Several routing schemes and routing protocols have been proposed for MANETs. These protocols are generally categorized as either table-driven or on-demand protocols. Table-driven protocols maintain routes to all possible destinations by exchanging route information periodically. On the other hand, on-demand routing protocols discover and maintain routes that are needed by the mobile nodes. Simulation studies [2], [3], and [4] showed that on-demand protocols outperform table-driven protocols because they respond more rapidly to topology changes, and they incur a much lower overhead. All these protocols, either table-driven or on-demand, use shortest route as the sole criterion to select a route. Shortest route, in this context, is the route with least number of hops. However, the use of the shortest path as the only criterion does not balance the load among the network nodes, and creates congested areas within the network. These congested areas degrade the protocols' performance by increasing the packet drop rate, packet end-to-end delay, and routing overhead. Moreover, concentrating the traffic in certain nodes within the network, deplete the battery power of these nodes at a faster rate. These problems worsen as the load on the network is increased, as was shown for on-demand protocols in [3] and [5], which are serious performance and scalability problems.

In this paper, we propose a Traffic-Size Aware routing scheme that uses the size of the traffic, through and around the network nodes, as the main route selection criterion. This scheme is meant to balance the load amongst the network nodes, and to avoid creating congested areas. In this scheme, the network nodes keep track of the size of traffic (in bytes) being routed. The nodes are also aware of the

size of the traffic that is routed through their neighbors. For any path that consists of multiple hops, the load metric of the path is the sum of all the traffic that is routed through all the hops that make up that path. Our scheme is an extension to the Virtual Path Routing Protocol (VPR) [6].

The rest of this paper is organized as follows. In Section 2, we describe the related protocols and briefly cover the functionalities of VPR [6]. In Section 3, we detail our Traffic-Size Aware Routing scheme. In Section 4, we integrate our scheme to VPR. In Section 5, we present a simulation study to evaluate the effectiveness of our scheme. We conclude our paper in Section 6.

2. Related Work

In this section, we review the proposed protocols that utilize the traffic load as the principle metric to select a path. As part of the review, we contrast these protocols to our scheme. We also describe the functionalities of VPR, which is the base protocol for our scheme.

2.1. Load Balancing Protocols

The load balancing routing protocols for ad hoc wireless networks can be generally divided into two types based on their basic techniques. The first type is "Traffic-Size" based, in which the load is balanced by attempting to distribute the traffic evenly among the network nodes. The second type is the "Delay" based, in which the load is balanced by attempting to avoid nodes with high delay. Although our scheme belongs to the "Traffic-Size" based type, we also briefly cover protocols that belong to the "Delay" based type for the sake of completeness.

Falling under the "Traffic-Size" type are the following three protocols: Dynamic Load-Aware Routing Protocol (DLAR) [7], Load-Balanced Ad Hoc Routing Protocol (LBAR) [8], and Load-Sensitive Routing Protocol (LSR) [9]. In DLAR, the load metric of a node is defined as the number of packets buffered in the node interface queue, and the load metric of a route is the summation of the load metrics of the nodes on that route. However, this technique does not optimally reflect the actual load since buffered packets may vary in size. Our scheme, on the other hand, measures the traffic through a node in bytes, which reveals the real load on the node. In LBAR, the load metric of a node is the total number of routes flowing through the node and its neighbors. This method is not optimal since it does not account for the various traffic sizes of each route. LSR defines the load metric of a node as the total number of packets buffered in the node interface and its neighbors. This technique is similar to the one used in DLAR, which does not take into account the different sizes of the buffered packets.

Belonging to the "Delay" based type are the following two protocols. First is the Delay-Oriented Shortest Path Routing Protocol (DOSPR) [10]. The main contribution of DOSPR is the factorization of access contention delay at the MAC layer to the total delay computation. Second is the Load-Aware On-Demand Routing Protocol (LAOR) [11] that computes and uses the total path delay as the load metric to select a route.

2.2. Virtual Paths Routing Protocol (VPR)

The key purpose of VPR [6] is to provide correct, efficient, highly adaptive, and dynamic route creation and maintenance among the nodes. VPR utilizes two well known routing techniques, namely source and table routing. It is a distributed and on-demand protocol that comprises two phases. The first phase is path creation, which is initiated when a source node needs to communicate with a destination node. All of the nodes within the vicinity potentially participate in this phase, which may yield more than one path. The second phase is path maintenance, in which the protocol monitors all the established paths. In this phase, the protocol observes local connectivity, link breakages, node mobility, and traffic size, and uses these observations to adjust its internal parameters accordingly. During this phase, nodes with active paths continually and controllably report their presences by broadcasting periodical HELLO messages.

To deliver a packet between a source and a destination node, a virtual path must be established between the two nodes. A virtual path is simply a route, or sequence of nodes, in which all the nodes are aware of the existence of the path, and they monitor its activities. Each node on any given virtual path knows its predecessor and successor nodes on the path. All the packets to be delivered through a particular virtual path are marked with a key that uniquely identifies the virtual path to be used; and each node passes the packets to the next node on the path until they reach their final destination. Since

every node operates as a router, a node must be able to handle more than one virtual path. To accomplish that, each participating node maintains a virtual paths routing table. This table contains the currently active paths, which are defined as the paths that are in use and fully operational.

The path creation in VPR is a two-phase process. The first phase is the Path Discovery, in which the source node uses a controlled flooding technique to determine the path through the ad hoc network to the destination. At the end of this phase, the source node may have multiple distinctive paths through which it may reach the destination node. Every path consists of nodes that can relay messages from source to destination. The second phase is the Path Set Up on all the nodes on the list that was obtained previously. At the end of this phase, each node on the path has an entry in its virtual paths routing table for the newly created path. At the end of the dialog between a source and destination node or when the virtual path is broken, the protocol deactivates the path by removing it from the virtual paths routing tables of all the involved nodes.

3. Traffic-Size Aware (TSA) Routing

The "Traffic-Size" based load balancing routing protocols we covered in Section 2 measure the traffic size in number of packets. Measuring the load by the number of packets is inaccurate since the size of the packets may differ. A more accurate method is to measure the traffic size in bytes.

When using VPR, every node maintains an entry for every active virtual path it services. The creation time of any entry (i.e., the creation time of a virtual path) is recorded in the entry itself by the node. The node also accumulates the number of packets and the size (in bytes) of every packet that it routes using a particular entry. The accumulated traffic size and number of packets are also recorded in the entry. Thus, any given entry contains the time at which the entry was created, the number of packets, and the size of the traffic that was routed using that entry. Hence, we define the *Entry Load* of entry *j* (EL_j), or the load of the virtual path *j*, at any node as:

$$EL_j = (ETS_j + (PN_j * MHS)) / (Current\ Time - ECT_j) \quad (1)$$

Where

ETS_j is accumulated Traffic Size (bytes) of entry *j*

PN_j is the number of packets routed by entry *j*

MHS is the MAC Header size

ECT_j is the Creation Time (s) of entry *j*

EL_j represents the load of a single entry at any given node, which is the number of bytes per second that were routed by the node using entry *j*. We factorize the MAC layer contention by including the terms (PN_j * *MHS*). PN_j is the number of packets routed using entry *j* and *MHS* is the size of the MAC header (in bytes) as defined by IEEE 802.11 Standard [12]. The use of the terms (PN_j * *MHS*) allows us to roughly treat the MAC contention as a traffic size. The higher the number of the packets, the higher the MAC contention and the load of entry *j* will be. Based on (1), we define the *Local Load* of node *n* (LL_n) as:

$$LL_n = \sum_{j=1}^{k} EL_j \quad (2)$$

Where k is the number of entries at node *n*. LL_n is the summation of the load of all the entries at node *n*, which represents the total traffic load that is routed by node *n*.

Nodes of an ad hoc network, within certain transmission ranges, communicate by means of a wireless medium. This configuration creates what we call *regions of contention* in which a number of nodes compete to gain access to the wireless medium to route their traffic. A similar observation was made in [8]. Thus, the load at any node not only depends on the traffic that is routed through the node itself, but also on the traffic that is routed through its neighboring nodes. To account for the load of neighboring nodes, we define a *Regional Load* of node *n* (RL_n) as:

$$RL_n = \sum_{\forall g} LL_g \quad (3)$$

Here, *g* is a neighboring node of node *n*. RL_n is the sum of the Local Loads of neighboring nodes of node *n*. From (2) and (3) we define the total load at node *n* (TL_n) as:

$$TL_n = Local\ Load + Regional\ Load$$
$$TL_n = LL_n + RL_n \quad (4)$$

TL_n is the local load plus the regional load of node *n*. It represents the traffic load that is passing through or around node *n*. TL_n is our basic unit to measure the load of the ad hoc network nodes.

Having defined the basic unit to measure the load of a node, we will now define the load metric of a path. A path, in VPR, between a source node *s* and a

destination node *d*, is a set of ordered nodes $[v_1,\ldots,v_n]$. If we assume that *P* denotes the path, then we can define a Path Load function *PL(P)* to calculate the load of that path as:

$$PL(P) = \sum_{i=1}^{n} TL_i \qquad (5)$$

Where *n* is the number of nodes on path *P* exclusive of the source and destination nodes. The function *PL* computes the load of path *P* by summing the total load (*TL*) of all the nodes that belong to the path. The path load that is calculated by the function *PL* is our load metric to compare different paths. The value returned by applying the function on path *P* represents the traffic load that would be experienced by packets to be sent on the path.

Path selection by a node is achieved by applying the function *PL* on all of the candidate paths. The path with the minimum value returned by the function is selected to route the traffic. In case of a tie, the path with the minimum number of hops would be selected.

4. Integrating the Scheme into VPR

The Traffic-Size Aware (TSA) scheme, described in Section 3, was integrated into VPR. The integration was straightforward because VPR was designed to be able to collect data about the network status during the Path Discovery Phase, and because every participating node is aware of the paths that it maintains. The Path Discovery process and the use of HELLO messages were slightly modified to adapt our scheme.

The Path Discovery (as described in [6]) starts when a source node broadcasts a Path Discovery packet to its immediate neighboring nodes. The header of the Path Discovery carries the address of the target node of the search. The header also contains a node list, which is used to record the route between the source and destination as the packet is propagated through the ad hoc network. When an intermediate node that is not the target receives the packet, it adds its address to the node list and then rebroadcasts the packet. Hence, the Path Discovery packet is disseminated through the network until it reaches its target node, which generates a Path Discovery Reply destined to the initiator of the search. The target node copies the Path Discovery packet's node list to the reply packet and appends its own address. The target node then uses a simple source routing technique, which traverses the packet's node list as the path to the source, to deliver the Path Discovery Reply to the initiator node. Upon receiving the Path Discovery Reply, the source node starts the virtual path creation phase.

The Path Discovery process illustrated above was modified as described in this paragraph. We added a new field to the standard VPR header. This field is called the Load Field, and it is initialized to zero by the source node before broadcasting a Path Discovery packet. Every intermediate node that receives the Path Discovery packet calculates its current Total Load (*TL*) and adds it to the value of the Load Field on the incoming packet. The result of the addition is assigned to the Load Field before the node rebroadcasts the packet. When generating the Path Discovery Reply, besides copying the Path Discovery packet's node list to the reply packet, the target node copies the value of the Load Field from the Discovery packet to the Reply packet. When the source node receives the Reply, it is in fact receiving the path to the intended destination and the Load Metric associated with that path.

To allow the source node to obtain more than one path, the destination node must reply to all Path Discovery requests it receives, and the source of the Path Discovery must wait for an interval of time (known as the PATH_RPLY_WAIT) after starting the Path Discovery process. When the node obtains more than one path, it simply selects the path with the minimum load metric associated with it.

The last modification we made to the Path Discovery of VPR is that we forbid intermediate nodes from replying to Path Discoveries using their cached paths. This prohibition guarantees the utilization of nearly current load information.

The use of HELLO Messages, employed by VPR, is also modified to allow the nodes to exchange their Local Load (*LL*) information. When a node sends a HELLO Message, it includes its current Local Load (*LL*) metric on the message. Every node maintains a list of its neighboring nodes and their local loads. When the node receives a HELLO Message from a neighbor, it checks its list of neighbors. If the neighbor is already on the list, it updates the neighbor's Local Load. Otherwise, it adds the new neighbor to the list. Failing to receive three-consective HELLO Messages from a neighbor already on the list, results in the removal of that neighbor from the list. The Local Loads in the neighbor list are used to calculate the *Regional Load* (*RL)* of the node.

5. Simulation Study

To evaluate the effectiveness of TSA, we decided to simulate the scheme and compare it to one of the routing protocols based on the shortest path routing technique. We chose to simulate and compare TSA to the Implicit Source Routing (ISR)[13] protocol. We chose ISR because it is the closest protocol to VPR in its basic functional mechanism. ISR creates temporary logical flows to route traffic between the nodes. Moreover, ISR is an enhanced version of the Dynamic Source Routing (DSR) [14] protocol, which is a well-studied and competitive routing protocol.

We compare the TSA to ISR using the *ns-2* network simulator [15], which includes a mobility extension that was ported from CMU's Monarch Group's mobility extension to *ns*. CMU's Monarch mobility extension to *ns-2* allows the simulation of multi-hop ad hoc wireless networks. The extension includes functionalities to simulate node movements, and to transmit and receive on wireless channels with a realistic radio propagation model.

We modeled our network interfaces after the Lucent WaveLan DSSS IEEE 802.11 product with a transmission rate of 2 Mbps. The interfaces use the IEEE 802.11 Distributed Coordination Function (DCF) [12] MAC protocol, which utilizes carrier sensing for collision avoidance. For the ISR simulation, we used the latest version available from the VINT project that comes with *ns-2*. That version includes DSR with a full implementation of the Implicit Source Routing (ISR) technique. We added and modified VPR to *ns-2* as described in [6] and in this paper. The parameter values used by both protocols are summarized in Table 1:

Table 1: Values used in the simulation.

Parameter	TSA	ISR
Send Buffer Size	64	64
Routing Table Size	64	30
Reply to Requests from the Cache	Off	On
Interface Queue Size	50	50
HELLO Interval	Dynamic	N/A
Allowed Lost HELLO	3	N/A
PATH_RPLY_WAIT	600 ms	N/A
MAC Header Size	30 Bytes	N/A

5.1. Traffic and Mobility Models

The traffic used in our simulations was Constant Bit Rate (CBR) traffic. The source and destination nodes were randomly selected, and each simulation shows the results of 32 connections. To vary the size of the traffic, we divided the 32 connections into four groups of eight connections each. The size of the CBR packets for the first group was 128 bytes. For the second, third, and fourth group, the sizes of the CBR packets were 256, 512, 1024 bytes, respectively. We modeled a 4 packets/sec send rate for all the groups. The mobility patterns in our simulation followed the *random waypoint* [16] model. In that model, each node starts at a random location, chooses a new location in a rectangular space (1500 m x 300 m) randomly, and starts its trip to the new location at a randomly chosen speed (uniformly distributed between 0–20 m/sec). After reaching its new location, a node pauses for a period of time (called the *pause time),* and then starts a new trip to a new location. We varied the mobility of the nodes by varying the *pause time* values. The results we present in this paper are based on simulation runs of 50 nodes. Each run lasted 500 seconds. Ten runs of different traffic and mobility scenarios are averaged to generate each data point. However, identical traffic and mobility scenarios were used for both protocols.

5.2. Results

We used three performance metrics to compare our schema to ISR. The first metric is the Packet Delivery Ratio, which is defined as the percentage of data packets delivered to their destination nodes of those sent by the source nodes. The second metric is the Routing Overhead of both protocols, which is defined as the number of routing packets "transmitted" per each data packet "delivered." On multi-hop routes, each transmission of the routing packets is counted as one transmission. We chose not to include the forwarding information carried in each data packet in our calculation of the overhead because the size is the same for both protocols. The third metric used is the Average End-to-End Delay of the data packets. We used eleven pause time values (0, 50, 100, 150, 200, 250, 300, 350, 400, 450, and 500 s) to differ the mobility level (with 0 s pause time meaning continually moving nodes and 500 s representing stationary nodes).

TSA had a better delivery ratio than ISR (see Figure 1). For the data packets sent, TSA delivered an average of 19.65% higher than ISR. The difference in the delivery ratio between both protocols is significant at the high level of mobility where the pause times are 0 s and 50 s. With pause times 0 s and 50 s, TSA

delivered 26.21% and 20.12% higher than ISR, respectively. At its best performance, ISR did not even deliver 50% of data packets sent. Whereas, TSA delivered 70% of the data packets that were sent.

These results are due to fact that TSA distributed the traffic among the network nodes and in a way to avoid the creation of highly congested areas. After further analysis of the simulation trace files, we found ISR to concentrate the traffic through centrally located nodes because it allows the network nodes to reply to path discoveries from their cached routes. The concentration of the traffic in certain nodes caused their interface queues to overflow and suffer from high drop rates. We also found this concentration to cause higher packet collision and, consequentially, the loss of these packets by ISR.

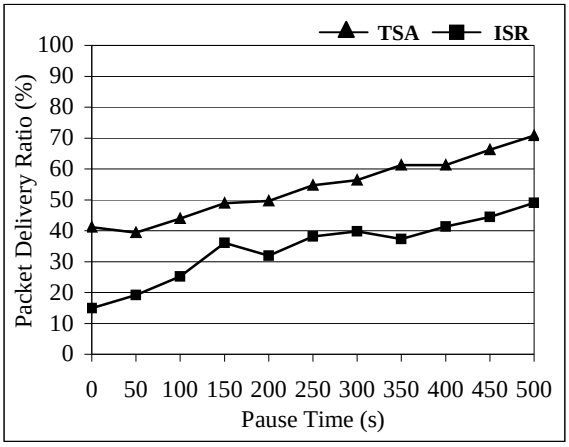

Figure 1: Packet Delivery Ratio.

TSA also outperformed ISR in the routing overhead metric (see Figure 2). ISR incurred an average of 11.42 overhead packets per data packet higher than TSA. We found that the higher the level of mobility, the higher the difference in overhead between TSA and ISR. At the highest level of mobility, ISR incurred about 38.53 of overhead packets per data packet, whereas TSA incurred about 1.04 overhead packets per data packet. At the lowest level of mobility, ISR incurred about 2.29 of overhead packets per data packet, while TSA incurred about 0.42 overhead packets per data packet. The reason for such high overhead is the additional Route Discoveries incurred by ISR through its salvaging process because of the congested network. We found that the data packets experienced delay times long enough to invalidate, due to the mobility, the routes of these packets. For those packets to be salvaged, ISR initiates the Route Discovery process to find alternative routes. Route Discovery, which is a flooding technique, is an expensive process in terms of overhead.

TSA surpassed ISR in the average end-to-end delay metric (see Figure 3). The average end-to-end delay for TSA was 2.59 s while it was 6.11 s for ISR. Generally, the average end-to-end delay of TSA was about three seconds less than that of ISR. The difference is significant at the highest level of mobility, where the average end-to-end delays for TSA and ISR were 3.44 s and 8.70 s, respectively. Because ISR does not balance the traffic load over the network nodes, it created highly congested regions in which the data packets suffered a long buffering time and the network nodes experienced a highly contended access to the medium. On the other hand, TSA avoided the creation of such regions by selecting routes based on load metric not shortest path.

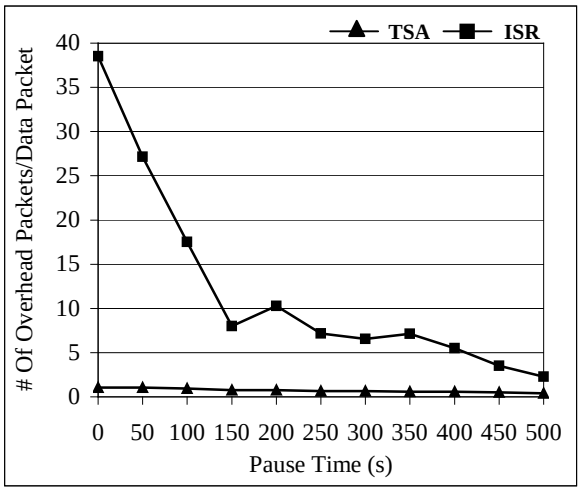

Figure 2: Overhead.

6. Conclusion

In this paper, we presented a novel scheme to route the traffic of ad hoc wireless networks. In contrast to all proposed protocols, our scheme measures the network traffic in bytes, not in number of packets. Measuring the traffic in bytes gives an accurate traffic load metric as opposed to measuring the traffic in number of packets because packet sizes may vary. We integrated our scheme into VPR and compared it to a shortest path based routing protocol,

namely ISR. We presented a simulation study that showed how our scheme outperformed the standard shortest path routing technique.

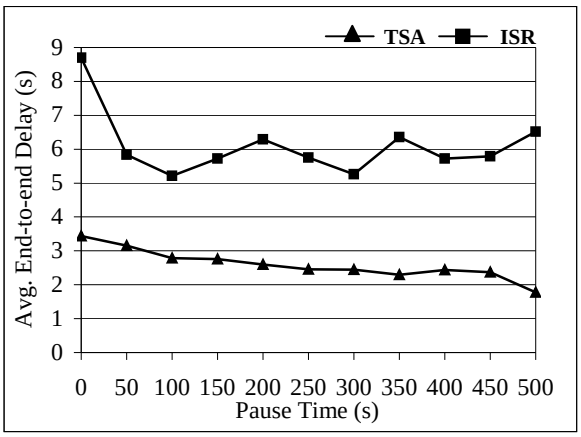

Figure 3: Average end-to-end delay.

ACKNOWLEDGMENT

The authors would like to thank Ann Jones for her assistance in compiling this paper.

REFERENCES

[1] J. Macker and S. Corson, "Mobile Ad Hoc Networks (MANET): Routing Protocol Performance Issues and Evaluation Considerations," *IETF RFC 2501*, January 1999.

[2] J. Broch, D.A. Maltz, D.B. Johnson, Y. Hu and J. Jetcheva, "A Performance Comparison of Multi-hop Wireless Ad Hoc Network Routing Protocols," in *Proceedings of ACM MOBICOM'98*, Dallas, Texas, USA, October 1998.

[3] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, And M. Degermark, "Scenario-based Performance Analysis of Routing Protocols for Mobile Ad-hoc Networks," in *Proceedings of ACM/IEEE MOBICOM'99*, Seattle, Washington, USA, August 1999.

[4] T. Larsson, N. Hedman, "Routing Protocols in Wireless Ad-Hoc Networks – A Simulation Study," Stockholm: Luleå University of Technology. Master Thesis. 1998.

[5] S. Das, C. Perkins, and E. Royer, "Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks," in *Proceedings of INFOCOM2002*, Tel Aviv, Israel, March 2000.

[6] A. Altalhi and G. Richard, III, "Virtual Paths Routing: A Highly Dynamic and Adaptive Routing Protocol for Ad Hoc Wireless Networks", to appear in the proceedings of the 1st International Workshop on Mobile Peer-to-Peer Computing (MP2P'04), Orlando, FL, USA March 14 – 17, 2004.

[7] S. –J. Lee and M. Gerla, "Dynamic Load-Aware Routing in Ad Hoc Networks," in *Proceedings of IEEE ICC'01*, Helsinki, Finland, June 2001.

[8] H. Hassanein and A. Zhou, "Routing with Load Balancing in Wireless Ad Hoc Networks," in *Proceedings of ACM MSWiM*, Rome, Italy, July 2001.

[9] K. Wu and J. Harms, "Load-Sensitive Routing for Mobile Ad Hoc Networks," in *Proceedings of IEEE ICCCN'01*, Scottsdale, AZ, USA, October 2001.

[10] S-T. Sheu and J. Chen, "A Novel Delay-Oriented Shortest Path Routing Protocol for Mobile Ad Hoc Networks," in *Proceedings of IEEE ICC'01*, Helsinki, Finland, June 2001.

[11] J-H. Song, V. Wong, and V. Leung, "Load-Aware On-Demand Routing (LAOR) Protocol for Mobile Ad hoc Networks," in *Proceedings of IEEE Vehicular Technology Conference* (*VTC-Spring*), Jeju, Korea, April 2003.

[12] IEEE Computer Society LAN/MAN Standards Committee, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std. 802.11-1997* (New York: IEEE, 1997).

[13] Yih-Chun Hu and D. B. Johnson, "Implicit Source Routes for On-Demand Ad Hoc Network Routing", in *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2001)*, Long Beach, California, USA, October 2001.

[14] D. B. Johnson, D. A. Maltz, Yih-Chun Hu, and J. G. Jetcheva, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks", Internet Draft, draft-ietf-manet-dsr-04.txt, November 2000.

[15] K. Fall and K. Varadhan, *The ns Manual*, The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, April 14, 2002. Available from http://www.isi.edu/nsnam/ns/.

[16] J. Broch, D. A. Maltz, D. B. Johnson, Yih-Chun Hu, and J. G. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols", in *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98)*, Dallas, Texas, USA, October 1998.