

# PSR: A Lightweight Proactive Source Routing Protocol For Mobile Ad Hoc Networks

Zehua Wang, *Student Member, IEEE*, Yuanzhu Chen, *Member, IEEE*, and Cheng Li, *Senior Member, IEEE*

**Abstract**—Opportunistic data forwarding has drawn much attention in the research community of multihop wireless networking, with most research conducted for stationary wireless networks. One of the reasons why opportunistic data forwarding has not been widely utilized in mobile ad hoc networks (MANETs) is the lack of an efficient lightweight proactive routing scheme with strong source routing capability. In this paper, we propose a lightweight proactive source routing (PSR) protocol. PSR can maintain more network topology information than distance vector (DV) routing to facilitate source routing, although it has much smaller overhead than traditional DV-based protocols [e.g., destination-sequenced DV (DSDV)], link state (LS)-based routing [e.g., optimized link state routing (OLSR)], and reactive source routing [e.g., dynamic source routing (DSR)]. Our tests using computer simulation in Network Simulator 2 (ns-2) indicate that the overhead in PSR is only a fraction of the overhead of these baseline protocols, and PSR yields similar or better data transportation performance than these baseline protocols.

**Index Terms**—Differential update, mobile ad hoc networks (MANETs), opportunistic data forwarding, proactive routing, routing overhead control, source routing, tree-based routing.

## I. INTRODUCTION

A mobile ad hoc network (MANET) is a wireless communication network, where nodes that are not within the direct transmission range of each other require other nodes to forward data. It can operate without existing infrastructure and support mobile users, and it falls under the general scope of multihop wireless networking. This networking paradigm originated from the needs in battlefield communications, emergency operations, search and rescue, and disaster relief operations. It has more recently been used for civilian applications such as community networks. A great deal of research results have been published since its early days in the 1980s [1]. The most salient research challenges in this area include end-to-end data transfer, link access control, security, and providing support for real-time multimedia streaming [2].

Manuscript received January 11, 2013; revised May 6, 2013; accepted June 22, 2013. Date of publication August 21, 2013; date of current version February 12, 2014. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Discovery Grant 293264-12, Discovery Grant 327667-2010, and Strategic Project Grant STPGP 397491-10. The review of this paper was coordinated by Prof. N. Kato.

Z. Wang is with Electrical and Computer Engineering, University of British Columbia, Vancouver BC V6T 1Z4, Canada (e-mail: zwang@ece.ubc.ca).

Y. Chen is with the Department of Computer Science, Memorial University of Newfoundland, St. John's, NL A1B 3X5, Canada (e-mail: yzchen@mun.ca).

C. Li is with the Faculty of Engineering and Applied Science, Memorial University of Newfoundland, St. John's, NL A1B 3X5, Canada (e-mail: licheng@mun.ca).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TVT.2013.2279111

The network layer has received a great deal of attention in the research on MANETs. As a result, abundant routing protocols in this network with differing objectives and for various specific needs have been proposed [3]. In fact, the two most important operations at the network layer, i.e., data forwarding and routing, are distinct concepts. Data forwarding regulates how packets are taken from one link and put on another. Routing determines what path a data packet should follow from the source node to the destination. The latter essentially provides the former with control input. Despite the amount of effort in routing in ad hoc networks, data forwarding, in contrast, follows the same paradigm as in Internet Protocol (IP) forwarding in the Internet. IP forwarding was originally designed for multihop wired networks, where one packet transmission can be only received by nodes attached to the same cable. However, in wireless networks, when a packet is transmitted over a physical channel, it can be that channel. Traditionally, overhearing a packet not intended for the receiving node had been considered completely negative, i.e., interference. Thus, in a sense, the goal of the research in wireless networking was to make wireless links as good as wired links.

Opportunistic data forwarding represents a promising solution to utilize the broadcast nature of wireless communication links [4]. Opportunistic data forwarding refers to a way in which data packets are handled in a multihop wireless network. Unlike traditional IP forwarding, where an intermediate node looks up a forwarding table for a dedicated next hop, opportunistic data forwarding allows potentially multiple downstream nodes to act on the broadcast data packet. One of the initial works on opportunistic data forwarding is selective diversity forwarding by Larsson [5]. In this paper, a transmitter picks the best forwarder from multiple receivers, which successfully received its data, and explicitly requests the selected node to forward the data. However, its overhead needs to be significantly reduced before it can be implemented in practical networks. This issue was successfully addressed in the seminal work on ExOR [6], outlining a solution at the link and network layers. In ExOR, nodes are enabled to overhear all packets on the air; therefore, a multitude of nodes can potentially forward a packet as long as they are included in the *forwarder list* carried by the packet. By utilizing the contention feature of the medium-access-control (MAC) sublayer, the forwarder *closer* to the destination will access the medium more aggressively. Therefore, the MAC sublayer can determine the actual next-hop forwarder to better utilize the long-haul transmissions.

To support opportunistic data forwarding in a mobile wireless network as in ExOR, an IP packet needs to be enhanced such that it lists the addresses of the nodes that lead to the

packet's destination. This entails a routing protocol where nodes see beyond merely the next hop leading to the destination. Therefore, link state (LS) routing [e.g., optimized LS routing (OLSR)] or source routing [e.g., dynamic source routing (DSR)] would seem to be good candidates. On one hand, LS routing protocols include interconnectivity information between remote nodes, which is hardly useful for a particular source node, but this incurs prohibitively large overhead. This is even true with optimization techniques such as multipoint relaying, as in OLSR [7]. On the other hand, if we wish to support opportunistic data forwarding in a MANET with constantly active data communication between many node pairs, the reactive nature of DSR [8] renders it unsuitable. Meanwhile, source routing is able to tightly control data forwarding paths. Thus, it is not only of interest in opportunistic data forwarding but also in a wider scope such as avoiding congestion, bypassing malicious nodes, and allocating network resources.

In this paper, we propose a lightweight *proactive source routing (PSR) protocol* to facilitate opportunistic data forwarding in MANETs. In PSR, each node maintains a breadth-first search spanning tree of the network rooted at itself. This information is periodically exchanged among neighboring nodes for updated network topology information. Thus, PSR allows a node to have full-path information to all other nodes in the network, although the communication cost is only linear to the number of the nodes. This allows it to support both source routing and conventional IP forwarding. When doing this, we try to reduce the routing overhead of PSR as much as we can. Our simulation results indicate that PSR has only a fraction of overhead of OLSR, DSDV, and DSR but still offers a similar or better data transportation capability compared with these protocols.

The remainder of this paper is organized as follows. Section II reviews related work on routing protocol in MANETs. Section III describes the design and implementation details of our proposed routing scheme. The computer simulation, related experiment settings, and comparisons between PSR and existing algorithms are presented in Section IV. Section V concludes this paper with a discussion of future research.

## II. RELATED WORK

Routing protocols in MANETs can be categorized using an array of criteria. The most fundamental among these is the timing of routing information exchange. On one hand, a protocol may require that nodes in the network should maintain valid routes to all destinations at all times. In this case, the protocol is considered *proactive*, which is also known as *table driven*. Examples of proactive routing protocols include destination-sequenced distance vector (DSDV) [9] and OLSR [7]. On the other hand, if nodes in the network do not always maintain routing information, when a node receives data from the upper layer for a given destination, it must first find out about how to reach the destination. This approach is called *reactive*, which is also known as *on demand*. DSR [8] and ad hoc on-demand DV (AODV) [10] fall in this category.

These well-known routing schemes can be also categorized by their fundamental algorithms. The most important algorithms in routing protocols are DV and LS algorithms. In an LS,

every node floods the information of the links between itself and its neighbors in the entire network, such that every other node can reconstruct the complete topology of the network locally. In a DV, a node only provides its neighbors with the cost to reach each destination. With the estimates coming from neighbors, each node is able to determine which neighbor offers the best route to a given destination. Both LS and DV support the vanilla IP packets. DSR, however, takes a different approach to on-demand *source* routing. In DSR, a node employs a path search procedure when there is a need to send data to a particular destination. Once a path is identified by the returning search control packets, this entire path is embedded in each data packet to that destination. Thus, intermediate nodes do not even need a forwarding table to transfer these packets. Because of its reactive nature, it is more appropriate for occasional or lightweight data transportation in MANETs.

AODV, DSDV, and other DV-based routing algorithms were not designed for source routing; hence, they are not suitable for opportunistic data forwarding. The reason is that every node in these protocols only knows the next hop to reach a given destination node but not the complete path. OLSR and other LS-based routing protocols could support source routing, but their overhead is still fairly high for the load-sensitive MANETs. DSR and its derivations have a long bootstrap delay and are therefore not efficacious for frequent data exchange, particularly when there are a large number of data sources.

In fact, many lightweight routing protocols had been proposed for the Internet to address its scalability issue, i.e., all naturally "table driven." The path-finding algorithm (PFA) [11] is based on DVs and improves them by incorporating the predecessor of a destination in a routing update. Hence, the entire path to each node can be reconstructed by connecting the predecessors and destinations; therefore, the source node will have a tree topology rooted at itself. In the meantime, the link vector (LV) algorithm [12] reduces the overhead of LS algorithms to a great deal by only including links to be used in data forwarding in routing updates. The extreme case of LV, where only one link is included per destination, coincides with the PFA.

PFA and LV were both originally proposed for the Internet, but their ideas were later used to devise routing protocols in the MANET. The Wireless Routing Protocol (WRP) [13] was an early attempt to port the routing capabilities of LS routing protocols to MANETs. It is built on the same framework of the PFA for each node to use a tree to achieve loop-free routing. Although it is an innovative exploration in the research on MANETs, it has a rather high communication overhead due to the amount of information stored at and exchanged by the nodes. This is exacerbated by the same route update strategy as in the PFA, where routing updates are triggered by topology changes. Although this routing update strategy is reasonable for the PFA in the Internet, where the topology is relatively stable, this turns out to be fairly resource demanding in MANETs. (Our original intention was to include the WRP in the experimental comparison later in this paper, and we have implemented WRP in ns2. Unfortunately, our preliminary tests indicate that the changing topology in the MANET incurs an overwhelming amount of overhead, i.e., at least an order of magnitude higher

than the other mainstream protocols. Thus, we do not include the simulation result of WRP as a baseline in our comparison.)

The PSR protocol proposed in this paper uses tree-based routing as in PFA and WRP. To make our PSR more suitable for the MANETs, we adopt a combined route update strategy that takes advantage of both “event-driven” and “timer-driven” approaches. Specifically, nodes would hold their broadcast after receiving a route update for a period of time. If more updates have been received in this window, all updates are consolidated before triggering one broadcast. The period of the update cycle is an important parameter in PSR. Furthermore, we go an extra mile to reduce its routing overhead. First, we interleave full-dump and differential updates to strike the balance between efficient and robust network operation. Second, we package affected links into forests to avoid duplicating nodes in the data structure. Finally, to further reduce the size of differential update messages, each node tries to minimize the alteration of the routing tree that it maintains as the network changes its structure.

### III. DESIGN OF PROACTIVE SOURCE ROUTING

Essentially, PSR provides every node with a breadth-first spanning tree (BFST) of the entire network rooted at itself. To do that, nodes periodically broadcast the tree structure to their best knowledge in each iteration. Based on the information collected from neighbors during the most recent iteration, a node can expand and refresh its knowledge about the network topology by constructing a deeper and more recent BFST. This knowledge will be distributed to its neighbors in the next round of operation (see Section III-A). On the other hand, when a neighbor is deemed lost, a procedure is triggered to remove its relevant information from the topology repository maintained by the detecting node (see Section III-B). Intuitively, PSR has about the same communication overhead as DV-based protocols. We go an extra mile to reduce the communication overhead incurred by PSR’s routing agents. Details about this overhead reduction will be discussed in Section III-C.

Before describing the details of PSR, we will first review some graph-theoretic terms used here. Let us model the network as undirected graph  $G = (V, E)$ , where  $V$  is the set of nodes (or vertices) in the network, and  $E$  is the set of wireless links (or edges). Two nodes  $u$  and  $v$  are connected by edge  $e = (u, v) \in E$  if they are close to each other and can directly communicate with given reliability. Given node  $v$ , we use  $N(v)$  to denote its open neighborhood, i.e.,  $\{u \in V | (u, v) \in E\}$ . Similarly, we use  $N[v]$  to denote its closed neighborhood, i.e.,  $N(v) \cup \{v\}$  (see [14] for other graph-theoretic notions).

#### A. Route Update

Due to its proactive nature, the update operation of PSR is iterative and distributed among all nodes in the network. At the beginning, node  $v$  is only aware of the existence of itself; therefore, there is only a single node in its BFST, which is root node  $v$ . By exchanging the BFSTs with the neighbors, it is able to construct a BFST within  $N[v]$ , i.e., the star graph centered at  $v$ , which is denoted  $S_v$ .

In each subsequent iteration, nodes exchange their spanning trees with their neighbors. From the perspective of node  $v$ , toward the end of each operation interval, it has received a set of routing messages from its neighbors packaging the BFSTs. Note that, in fact, more nodes may be situated within the transmission range of  $v$ , but their periodic updates were not received by  $v$  due to, for example, bad channel conditions. After all, the definition of a neighbor in MANETs is a fickle one. (We have more details on how we handle lost neighbors subsequently.) Node  $v$  incorporates the most recent information from each neighbor to update its own BFST. It then broadcasts this tree to its neighbors at the end of the period. Formally,  $v$  has received the BFSTs from some of its neighbors. Including those from whom  $v$  has received updates in recent previous iterations, node  $v$  has a BFST, which is denoted  $T_u$ , cached for each neighbor  $u \in N(v)$ . Node  $v$  constructs a union graph, i.e.,

$$G_v = S_v \cup \bigcup_{u \in N(v)} (T_u - v). \quad (1)$$

Here, we use  $T - x$  to denote the operation of removing the subtree of  $T$  rooted at node  $x$ . As special cases,  $T - x = T$  if  $x$  is not in  $T$ , and  $T - x = \emptyset$  if  $x$  is the root of  $T$ . Then, node  $v$  calculates a BFST of  $G_v$ , which is denoted  $T_v$ , and places  $T_v$  in a routing packet to broadcast to its neighbors.

In fact, in our implementation, the given update of the BFST happens multiple times within a single update interval so that a node can incorporate new route information to its knowledge base more quickly. To the extreme,  $T_v$  is modified every time a new tree is received from a neighbor. Apparently, there is a tradeoff between the routing agent’s adaptivity to network changes and computational cost. Here, we choose routing adaptivity as a higher priority assuming that the nodes are becoming increasingly powerful in packet processing. Nevertheless, this does not increase the communication overhead at all because one routing message is always sent per update interval.

Assume that the network diameter, i.e., the maximum pairwise distance, is  $D$  hops. After  $D$  iterations of operation, each node in the network has constructed a BFST of the entire network rooted at itself since nodes are timer driven and, thus, synchronized. This information can be used for any source routing protocol. The amount of information that each node broadcasts in an iteration is bounded by  $O(|V|)$ , and the algorithm converges in  $D$  iterations.

#### B. Neighborhood Trimming

The periodically broadcast routing messages in PSR also double as “hello” messages for a node to identify which other nodes are its neighbors. When a neighbor is deemed lost, its contribution to the network connectivity should be removed; this process is called neighbor trimming. Consider node  $v$ . The neighbor trimming procedure is triggered at  $v$  about neighbor  $u$  either by the following cases:

- 1) No routing update or data packet has been received from this neighbor for a given period of time.
- 2) A data transmission to node  $u$  has failed, as reported by the link layer.

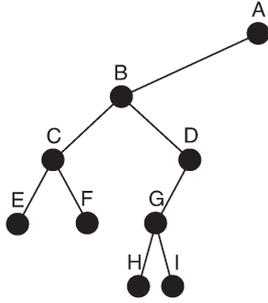


Fig. 1. Binary tree.

Node  $v$  responds by:

- 1) first, updating  $N(v)$  with  $N(v) - \{u\}$ ;
- 2) then, constructing the union graph with the information of  $u$  removed, i.e.,

$$G_v = S_v \cup \bigcup_{w \in N(v)} (T_w - v) \quad (2)$$

- 3) finally, computing BFST  $T_v$ .

Notice that  $T_v$ , which is thus calculated, is not broadcast immediately to avoid excessive messaging. With this updated BFST at  $v$ , it is able to avoid sending data packets via lost neighbors. Thus, multiple neighbor trimming procedures may be triggered within one period.

### C. Streamlined Differential Update

In addition to dubbing route updates as hello messages in PSR, we interleave the “full dump” routing messages, as stated previously, with “differential updates.” The basic idea is to send the full update messages less frequently than shorter messages containing the difference between the current and previous knowledge of a node’s routing module. Both the benefit of this approach and balancing between these two types of messages have been extensively studied in earlier proactive routing protocols. In this paper, we further streamline the routing update in two new avenues. First, we use a compact tree representation in full-dump and differential update messages to halve the size of these messages. Second, every node attempts to maintain an updated BFST as the network changes so that the differential update messages are even shorter.

- 1) Compact tree representation. For the full-dump messages, our goal is to broadcast the BFST information stored at a node to its neighbors in a short packet. To do that, we first convert the general rooted tree into a binary tree of the same size, e.g.,  $s$  nodes, using left-child sibling representation. Then, we serialize the binary tree using a bit sequence of  $34 \times s$  bits, assuming that IPv4 is used. Specifically, we scan the binary tree layer by layer. When processing a node, we first include its IP address in the sequence. In addition, we append two more bits to indicate if it has the left and/or right child. For example, the binary tree in Fig. 1 is represented as A10B11C11D10E00F00G11H00I00. As such, the size of the update message is a bit over half compared with

the traditional approach, where the message contains a discrete set of edges.

The difference between two BFSTs can be represented by the set of nodes that have changed parents, which are essentially a set of edges connecting to the new parents. We observe that these edges are often clustered in groups. That is, many of them form a sizeable tree subgraph of the network. Similar to the case of full dump, rather than using a set of loose edges, we use a tree to package the edges connected to each other. As a result, a differential update message usually contains a few small trees, and its size is noticeably shorter.

- 2) Stable BFST. The size of a differential update is determined by how many edges it includes. Since there can be a large number of BFSTs rooted at a given node of the same graph, we need to alter the BFST maintained by a node as little as possible when changes are detected. To do that, we modify the computation described earlier here, such that a small portion of the tree needs to change either when a neighbor is lost or when it reports a new tree.

Consider node  $v$  and its BFST  $T_v$ . When it receives an update from neighbor  $u$ , which is denoted  $T_u$ , it first removes the subtree of  $T_v$  rooted at  $u$ . Then, it incorporates the edges of  $T_u$  for a new BFST. Note that the BFST of  $(T_v - u) \cup T_u$  may not contain all necessary edges for  $v$  to reach every other node. Therefore, we still need to construct union graph

$$(T_v - u) \cup \bigcup_{w \in N(v)} (T_w - v) \quad (3)$$

before calculating its BFST. To minimize the alteration to the tree, we add one edge of  $T_w - v$  to  $T_v - u$  at a time. When node  $v$  thinks that a neighbor  $u$  is lost, it deletes edge  $(u, v)$  but still utilizes the network structure information contributed by  $u$  earlier. That is, even if it has moved away from  $v$ , node  $u$  may still be within the range of one of  $v$ ’s neighbors. As such,  $T_v$  should be updated to a BFST of

$$(T_v - u) \cup (T_u - v) \cup \bigcup_{w \in N(v)} (T_w - v). \quad (4)$$

Note that, since  $N(v)$  no longer contains  $u$ , we need to explicitly put it back into the equation. Similarly in this case, the edges of  $(T_u - v) \cup \bigcup_{w \in N(v)} (T_w - v)$  are added to  $(T_v - u)$  one at a time, with those just removed because of  $u$  taking priority.

## IV. PERFORMANCE EVALUATION

We study the performance of PSR using computer simulation with Network Simulator 2 version 2.34 (ns-2). We compare PSR against OLSR [7], DSDV [9], and DSR [8], which are three fundamentally different routing protocols in MANETs, with varying network densities and node mobility rates. We measure the data transportation capacity of these protocols supporting the Transmission Control Protocol (TCP) and the

User Datagram Protocol (UDP) with different data flow deployment characteristics. Our tests show that the overhead of PSR is indeed only a fraction of that of the baseline protocols. Nevertheless, as it provides global routing information at such a small cost, PSR offers similar or even better data delivery performance. Here, we first describe how the experiment scenarios are configured and what measurements are collected. Then, we present and interpret the data collected from networks with heavy TCP flows and from those with light UDP streams.

### A. Experiment Settings

Since many routing protocols' performances are well known in the classic two-ray ground reflection propagation model, we select such a model as well in our simulation to present a consistent and comparable result.<sup>1</sup> Without loss of generality, we select a 1-Mb/s nominal data rate at the IEEE 802.11 links to study the relative performance among the selected protocols. With the default physical-layer parameters of the simulator, the transmission range is approximately 250 m, and the carrier sensing range is about 550 m.

We compare the performance of PSR with that of OLSR, DSDV, and DSR. The reasons that we select these baseline protocols that are different in nature are as follows. On one hand, OLSR and DSDV are both proactive routing protocols, and PSR is also in this category. On the other hand, OLSR makes complete topological structure available at each node, whereas in DSDV, nodes only have distance estimates to other nodes via a neighbor. PSR sits in the middle ground, where each node maintains a spanning tree of the network. Furthermore, DSR is a well-accepted reactive source routing scheme, and as with PSR, it support source routing, which does not require other nodes to maintain forwarding lookup tables. All three baseline protocols are configured and tested out of the box of ns-2.

In modeling node mobility of the simulated MANETs, we use the random waypoint model to generate node trajectories. In this model, each node moves toward a series of target positions. The rate of velocity for each move is uniformly selected from  $[0, v_{\max}]$ . Once it has reached a target position, it may pause for a specific amount of time before moving toward the next position. This mobility model may eventually lead to an uneven node distribution in the network. That is, the nodes' density in the central area of the network may be higher than that at the network boundary. This uneven node distribution coincides with the real case in our daily life. However, at the beginning of simulations, the nodes' positions are evenly assigned; therefore, we discard the simulation data in the first 30 s, and only the data at a steady state is collected. All networks have 50 nodes in our tests. We have two series of scenarios based on the mobility model. The first series of scenarios have a fixed  $v_{\max}$  but different network densities by varying the network dimensions. The second series have the same network density but varying  $v_{\max}$ .

<sup>1</sup>In our previous paper [15], PSR's performance is also tested under a more realistic physical model with opportunistic forwarding techniques.

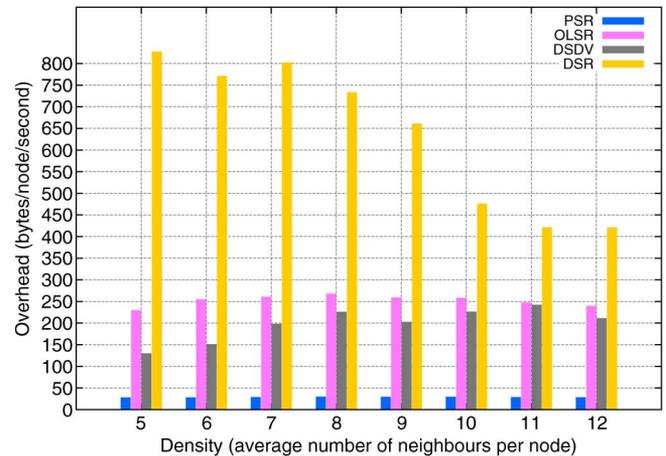


Fig. 2. Routing overhead with density.

We study the data transportation capabilities of these routing schemes and their overhead in doing so by loading the networks with TCP data flows and UDP voice streams.

- 1) To test how TCP is supported, in each scenario, we randomly select 40 nodes out of the 50 and pair them up. For each pair, we set up a permanent one-way File Transfer Protocol (FTP) data transfer. We repeat the selection of the 40 nodes five times and study their collective behavior. Since TCP's congestion avoidance mechanism always tries to inject as much data in the network as possible, this essentially mimics heavily loaded mobile networks. For all four protocols, we measure their TCP throughput, end-to-end delay, and routing overhead in bytes per node per second in each scenario, where each scenario has 20 simulation instances.
- 2) To study their performance in supporting UDP, we use two-way constant-bit-rate (CBR) streams for compressed voice communications. Specifically, we select three pairs of nodes and feed each node with a CBR flow of 160 B/packets and 10 packets/s, which simulates mobile networks with a light voice communication load. We measure the packet delivery ratio (PDR), end-to-end delay, and delay jitter in each scenario.

Results about TCP (see Section IV-B and C) and UDP (see Section IV-D and E) with regard to varying node densities and velocity rates are in the following.

### B. TCP With Node Density

We first study the performance of PSR, OLSR, DSDV, and DSR in supporting 20 TCP flows in networks with different node densities. Specifically, with the default 250-m transmission range in ns-2, we deploy our 50-node network in a square space of varying side lengths that yield node densities of approximately 5, 6, 7, ..., 12 neighbors per node. These nodes move following the random waypoint model with  $v_{\max} = 30$  m/s.

We plot in Fig. 2 the per-node per-second routing overhead, i.e., the amount of routing information transmitted by the routing agents measured in B/node/s, of the four protocols when they transport a large number of TCP flows. This figure shows

that the overhead of PSR (20 to 30) is just a fraction of that of OLSR and DSDV (140 to 260) and more than an order of magnitude smaller than DSR (420 to 830). The routing overhead of PSR, OLSR, and DSDV goes up gradually as the node density increases. This is a typical behavior of proactive routing protocols in MANETs. These protocols usually use a fixed-time interval to schedule route exchanges. While the number of routing messages transmitted in the network is always constant for a given network, the size of such message is determined by the node density. That is, a node periodically transmits a message to summarize changes as nodes have come into or gone out of its range. As a result, when the node density is higher, a longer update message is transmitted even if the rate of node motion velocity is the same. Note that when the node density is really high, e.g., around 10 and 12, the overhead of OLSR flattens out or even slightly decreases. This is a feature of OLSR when its multipoint relaying mechanism becomes more effective in removing duplicate broadcasts, which is the most important improvement of OLSR over conventional LS routing protocols. PSR uses a highly concise design of messaging, allowing it to have much smaller overhead than the baseline protocols. In contrast, DSR, as a reactive routing protocol, incurs significantly higher overhead when transporting a large number of TCP flows because every source node needs to conduct its own route search. This is not surprising as reactive routing protocols were not meant to be used in such scenarios. Later in our experiments (see Section IV-D and E), we test all four protocols supporting only a few UDP streams for a different perspective. Here, the routing overhead of DSR decreases with the node density going up and the network diameter going down. This is because the number of hops to a destination is smaller in a denser network; therefore, the shorter and more robust routes break less frequently and do not need as many route searches. Furthermore, compared with IP forwarding, the fact that DSR is source routing and that intermediate nodes cannot modify the routes embedded in data packets works against its performance in a mobile network, both in terms of the increase in search operations and the loss of data transportation capacity. The reason is that, because a source node can be quite a few hops away from the destination, its knowledge about the path as embedded in the packets can become obsolete quickly in a highly mobile network. As a packet progresses en route, if an intermediate node cannot reach the next hop, as indicated in the embedded path, it will be dropped. This is very different from IP forwarding, where intermediate nodes can have more updated routing information than the source and can utilize that information in forwarding decisions.

Fig. 3 plots the TCP throughput of the four protocols for the same node density levels as before. The total throughput of the 20 TCP flows of PSR, OLSR, and DSDV is noticeably higher than that of DSR. In addition, while the TCP throughput of DSR decreases with node density, that for the other three are somewhat unaffected, hovering at around 500 kb/s. Apparently, the large routing overhead of DSR, particularly in dense networks, consumes a fair amount of channel bandwidth, leaving less room for data transportation. In most cases, PSR has the highest throughput because it needs to give up the least network resources for routing.

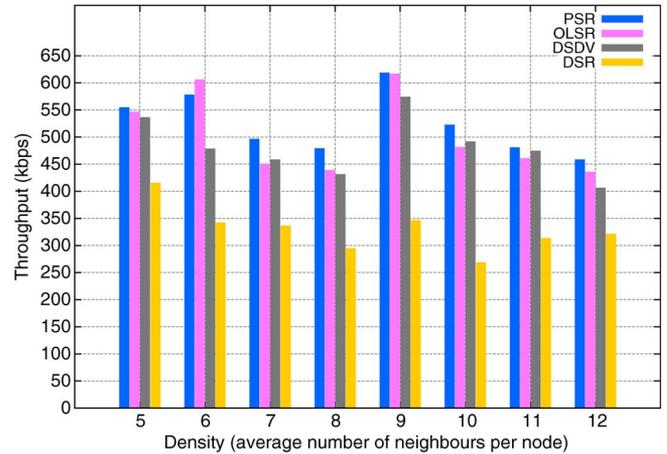


Fig. 3. TCP throughput with density.

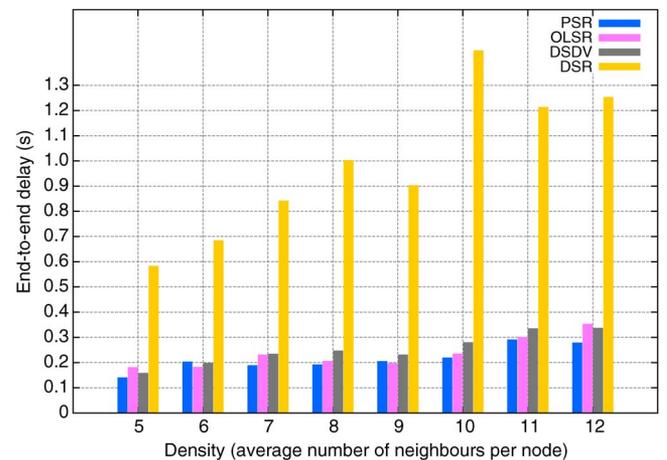


Fig. 4. End-to-end delay in TCP with density.

Next, we focus on the end-to-end delay of TCP flows to investigate how well these protocols support time-sensitive applications. Fig. 4 shows the delay measured for different node densities. As the density increases from 5 to 12 neighbors, the delay of DSR goes up from 0.58 to about 1.5 s, which is significantly higher than the typical value of 0.15 to 0.35 s for the other three protocols. This difference is caused by the initial route search when a TCP flow starts and by the subsequent searches triggered by route errors. As the network becomes denser, all protocols show an increasing trend in end-to-end delay. This may seem counterintuitive as, in denser networks, the average hop distance between source–destination pairs is smaller, which should lead to shorter round-trip time. However, this benefit is completely offset by more intense channel contention. Recall that the node density is inversely proportional to the square of the network diameter. As such, in the interplay between route length and channel contention, the latter dominates the overall effect.

### C. TCP With Velocity

We also study the performance of PSR and compare it to OLSR, DSDV, and DSR with different rates of node velocity. In particular, we conduct another series of tests in networks

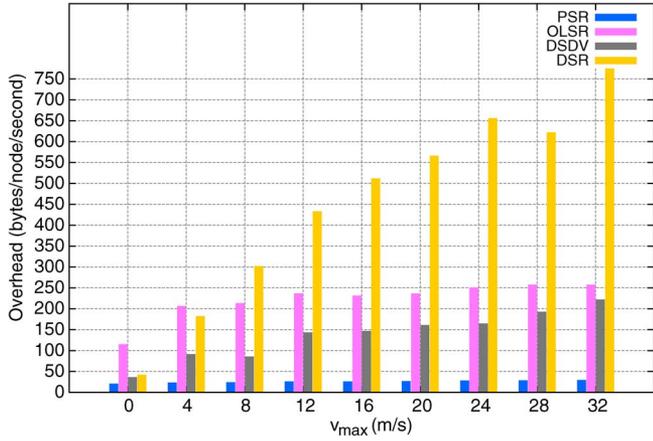


Fig. 5. Routing overhead with velocity.

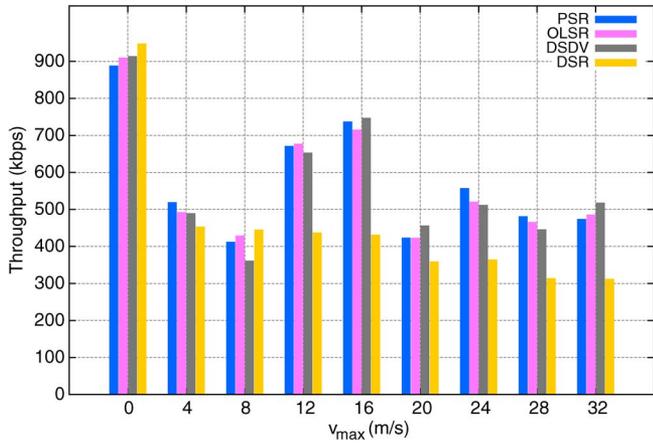


Fig. 6. TCP throughput with velocity.

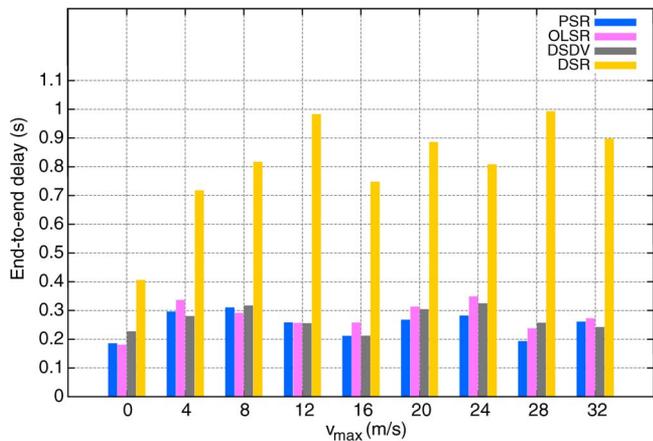


Fig. 7. End-to-end delay in TCP with velocity.

of 50 nodes deployed in a  $1100 \times 1100$  ( $m^2$ ) square area with  $v_{max}$  set to 0, 4, 8, 12, ..., 32 (m/s). The network thus has an effective node density of around seven neighbors per node, i.e., a medium density among those configured earlier. As with before, 20 TCP one-way flows are deployed between 40 nodes, and we measure the routing overhead, TCP throughput, and end-to-end delay (see Figs. 5–7).

The routing overhead of all four protocols with varying rates of node velocity is plotted as in Fig. 5. Note that the velocity

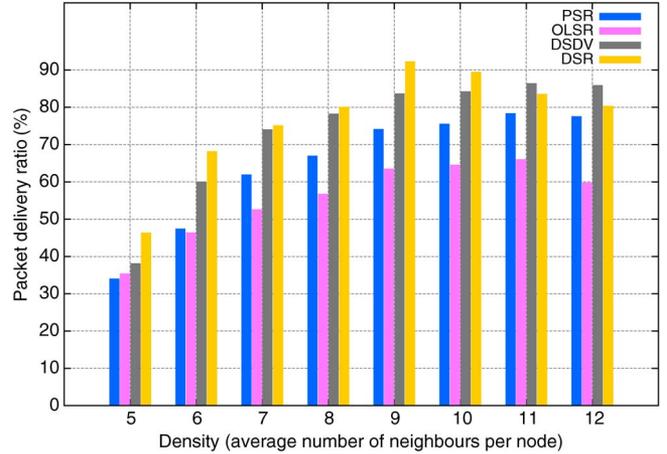


Fig. 8. PDR in UDP with density.

to the right of the  $x$ -axis corresponds to the middle bars in Fig. 2. We observe in the plot here that, as  $v_{max}$  decreases, the overhead of all protocols comes down. The reason for DSR is that, as the network structure becomes more stable, fewer route repair attempts are necessary. For the case of the proactive protocols, it is the reduction in the size of routing messages (i.e., fewer neighbors have changed positions) that cuts down the overhead. Still relative among these four protocols, when the network is not stationary ( $v_{max} \neq 0$ ), the overhead of PSR (20–30 B/node/s) is a fraction of that of OLSR and DSDV (90–300 B/node/s) and more than an order of magnitude lower than DSR (180–770 B/node/s).

The TCP throughput and end-to-end delay are plotted in Figs. 6 and 7, respectively. From these figures, we observe that the performance of PSR, OLSR, and DSDV are similar with PSR leading the pack in most cases. In addition, neither throughput nor delay is affected by the different rates of velocity. The only exception is that, when  $v_{max} = 0$ , all protocols yield a high throughput of 900 kb/s. With a greater portion of the channel bandwidth devoured by routing messages in highly mobile networks, DSR suffers a noticeable performance penalty in TCP throughput and end-to-end delay.

#### D. UDP With Density

We also tested the four protocols for their performance in transporting a small number of UDP streams. This is a typical assumption for ideal scenarios of reactive routing protocols. Here, we deploy three two-way UDP streams to simulate compressed voice communications. To find out about how node density affects these protocols, we use the same network and mobility configurations as in Section IV-B. We measure and plot the PDR (see Fig. 8), delay (see Fig. 9), and delay jitter (see Fig. 10) against varying node densities.

In Fig. 8, the PDRs of all four protocols are in the same ballpark across different node densities, with DSR slightly in the lead and OLSR trailing behind. This verifies that the traffic configuration is favorable for DSR. The relatively high loss rate of OLSR among the proactive routing protocols is caused by the higher routing overhead compared with PSR and DSDV. When

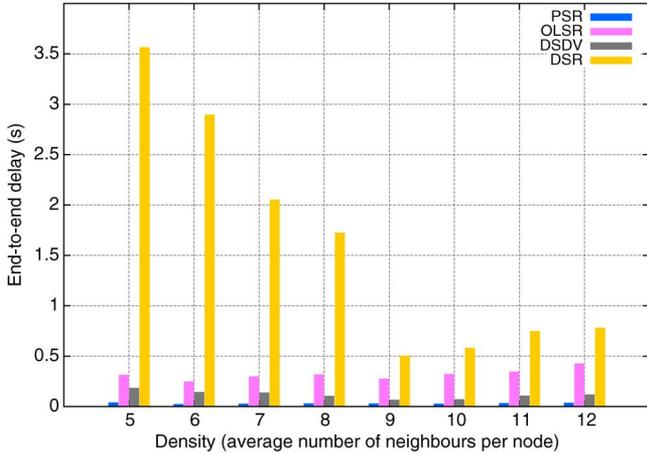


Fig. 9. End-to-end delay in UDP with density.

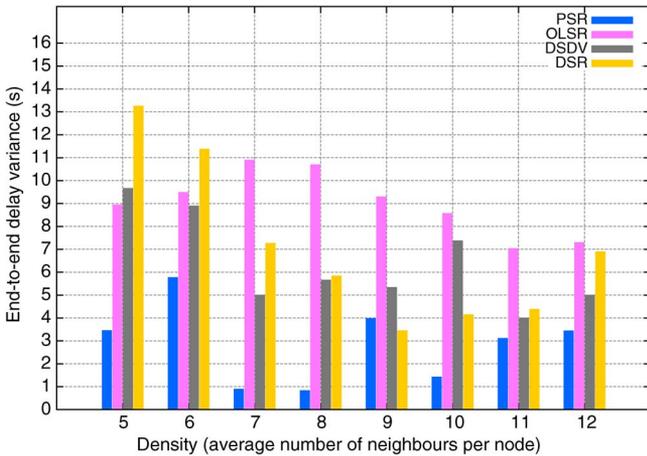


Fig. 10. End-to-end delay jitter in UDP with density.

the nodes are neither too sparse so that the network connectivity is good nor too dense so that the channel can be spatially reused, these protocols have a fairly high PDR of over 70% for PSR, DSDV, and DSR, and of 60%–70% for OLSR.

When we turn to end-to-end delay (see Fig. 9), there is a noticeable difference between DSR and the proactive protocols. In particular, DSR as a reactive protocol has a rather large delay in sparse networks. This is because the long vulnerable routes discovered during the search procedure break frequently, forcing nodes to hold packets back for an extended period before new routes are identified. Conversely, the network sparsity does not affect proactive protocols as much because their periodic routing information exchange makes them more prepared for network structure alteration. While the delay of DSR is off the chart, that of PSR is always less than 0.05 s, which is also much less than that for DSDV and OLSR (0.1–0.43 s). On a related note, the delay jitter (see Fig. 10) of PSR is significantly lower than the other three. Note that voice-over-IP (VoIP) applications usually discard packets that arrive too late. Therefore, the jitter among the packets actually used by the VoIP receiving agent is much smaller. Nevertheless, our metric still reflects how consistent these protocols are in delivering best-effort packets.

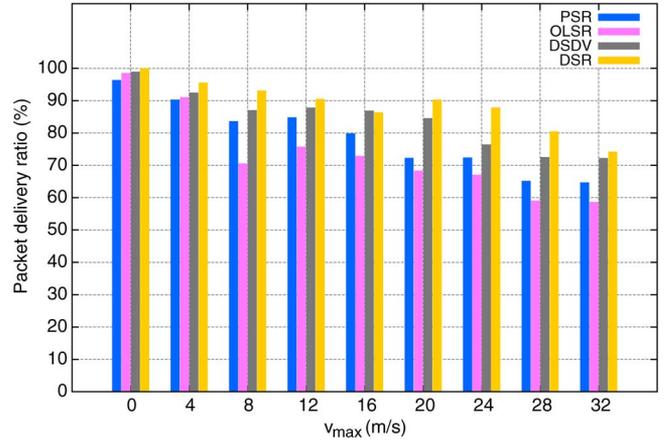


Fig. 11. PDR in UDP with velocity.

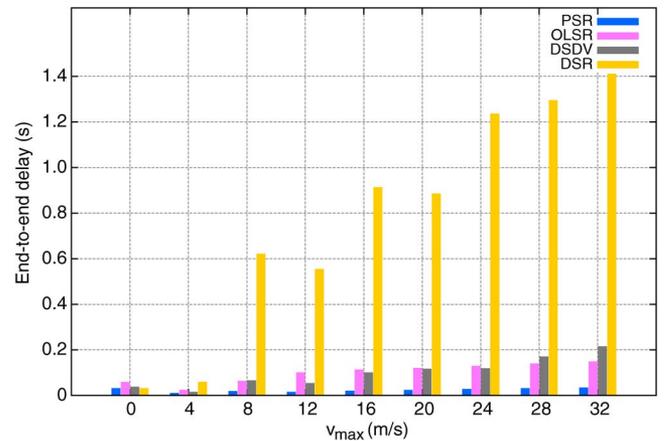


Fig. 12. End-to-end delay in UDP with velocity.

### E. UDP With Velocity

The same measurements are taken to test these protocols in response to different rates of node velocity. As with the case earlier, we pick three node pairs out of the 50 nodes and give them two-way CBR streams. For the entire series of different velocity caps  $v_{max} = 0, 4, 8, 12, \dots, 32$  m/s, the node density is again set to around seven neighbors per node.

From the plot of PDR (see Fig. 11), we observe that DSR is able to support three voice streams with little packet loss. Specifically, the PDR of DSR, PSR, and DSDV is always over 70% even when  $v_{max} = 32$  m/s. The reliability of OLSR is relatively lower, which can go below 60% at high speed ( $v_{max} = 28$  or 32 m/s). Note that all four protocols are very reliable in data delivery when  $v_{max} = 0$  or 4 m/s, where the loss rates are well below 10%. Their performance in terms of PDR degrades gracefully as the rate of node velocity increases.

The end-to-end delay (see Fig. 12) presents a rather distinct landscape. In particular, the number for DSR is significantly higher than the other protocols, except in low-mobility networks with  $v_{max} = 0$  or 4 m/s. In all cases, the delay for PSR is much smaller compared with OLSR and DSDV. On the other hand, the measured delay jitter (see Fig. 13) indicates that all protocols become less consistent when nodes move faster. Relatively speaking, however, the variance of PSR is much smaller than the other three.

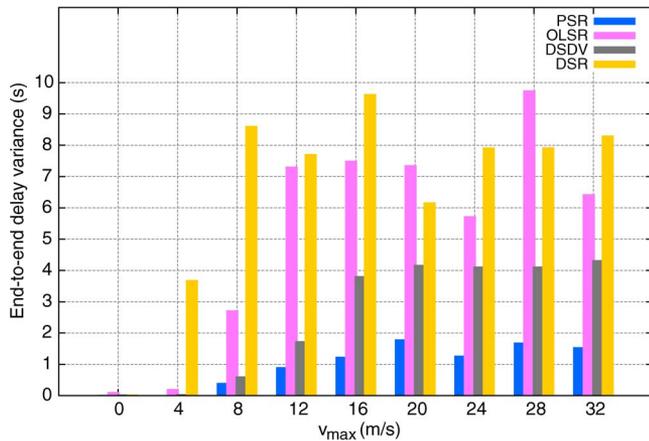


Fig. 13. End-to-end delay jitter in UDP with velocity.

## V. CONCLUSION

This paper has been motivated by the need to support opportunistic data forwarding in *MANETs*. To generalize the milestone work of ExOR for it to function in such networks, we needed a PSR protocol. Such a protocol should provide more topology information than DVs but must have significantly smaller overhead than LS routing protocols; even the MPR technique in OLSR would not suffice. Thus, we put forward a tree-based routing protocol, i.e., PSR, which is inspired by the PFA and the WRP. Its routing overhead per time unit per node is on the order of the number of the nodes in the network as with DSDV, but each node has the full-path information to reach all other nodes. For it to have a very small footprint, PSR's route messaging is designed to be very concise. First, it uses only one type of message, i.e., the periodic route update, both to exchange routing information and as hello beacon messages. Second, rather than packaging a set of discrete tree edges in the routing messages, we package a converted binary tree to reduce the size of the payload by about a half. Third, we interleave full-dump messages with differential updates so that, in relatively stable networks, the differential updates are much shorter than the full-dump messages. To further reduce the size of the differential updates, when a node maintains its routing tree as the network changes, it tries to minimize alteration of the tree. As a result, the routing overhead of PSR is only a fraction or less compared with DSDV, OLSR, and DSR, as evidenced by our experiments. Yet, it still has similar or better performance in transporting TCP and UDP data flows in mobile networks of different velocity rates and densities.

In the simulation in this paper, we used PSR to support traditional IP forwarding for a closer comparison with DSDV and OLSR, whereas DSR still carried source-routed messages. In our simultaneous work, i.e., CORMAN [15], we tested PSR's capability in transporting source-routed packets for opportunistic data forwarding, where we also found that PSR's small overhead met our initial goal. That being said, as indicated earlier in Section IV-B, while alleviating forwarding nodes from table lookup, DSR's source routing is particularly vulnerable in rapidly changing networks. The reason for this is that, as a source-routed packet progresses further from its source, the path carried by the packet can become obsolete, forcing an

intermediate node that cannot find the next hop of the path to drop the packet. This is fundamentally different from traditional IP forwarding in proactive routing with more built-in adaptivity, where the routing information maintained at nodes closer to the destination is often more updated than the source node. Although out of the scope of this paper, it would be an interesting exploration to allow intermediate nodes running DSR to modify the path carried by a source-routed packet for it to use its more updated knowledge to route data to the destination. This is in fact exactly what PSR does when we used it to carry source-routed data in CORMAN. Granted, this opens up an array of security issues, which themselves are part of a vast research area.

As with many protocol designs, in many situations working on PSR, we faced tradeoffs of sorts. Striking such balances not only gave us the opportunity to think about our design twice but also made us understand the problem at hand better. One particular example is related to trading computational power for data transfer performance. During one route exchange interval, a node receives a number of routing messages from its neighbors. It needs to incorporate the updated information to its knowledge base and share it with its neighbors. The question is when should these two events happen. Although incorporating multiple trees at one time is computationally more efficient, we chose to do that immediately after receiving an update from a neighbor. As such, the more accurate information takes effect without any delay. Otherwise, when a data packet is forwarded to a neighbor that no longer exists, it causes link layer retrieval, backlogging of subsequent packets, and TCP congestion avoidance and retransmission. With the broadcast and shared nature of the wireless channel, the effects above are adversary to all other data flows in the area. Therefore, in research on multihop wireless networking, it usually makes sense for us to minimize any impact on the network's communication resources even if there is penalty in other aspects. When it comes to the case when a node should share its updated route information with its neighbors, we chose to delay it until the end of the cycle so that only one update is broadcast in each period. If a node were to transmit it immediately when there is any change to its routing tree, it would trigger an explosive chain reaction and the network would be overwhelmed by the route updates. As we found out in our preliminary tests, this is the primary reason that WRP's overhead was significantly higher than the other protocols under study.

## REFERENCES

- [1] I. Chlamtac, M. Conti, and J.-N. Liu, "Mobile ad hoc networking: Imperatives and challenges," *Ad Hoc Netw.*, vol. 1, no. 1, pp. 13–64, Jul. 2003.
- [2] M. Al-Rabayah and R. Malaney, "A new scalable hybrid routing protocol for VANETs," *IEEE Trans. Veh. Technol.*, vol. 61, no. 6, pp. 2625–2635, Jul. 2012.
- [3] R. Rajaraman, "Topology control and routing in ad hoc networks: A survey," *ACM SIGACT News*, vol. 33, no. 2, pp. 60–73, Jun. 2002.
- [4] Y. P. Chen, J. Zhang, and I. Marsic, "Link-layer-and-above diversity in multi-hop wireless networks," *IEEE Commun. Mag.*, vol. 47, no. 2, pp. 118–124, Feb. 2009.
- [5] P. Larsson, "Selection diversity forwarding in a multihop packet radio network with fading channel and capture," *ACM Mobile Comput. Commun. Rev.*, vol. 5, no. 4, pp. 47–54, Oct. 2001.
- [6] S. Biswas and R. Morris, "ExOR: Opportunistic multi-hop routing for wireless networks," in *Proc. ACM Conf. SIGCOMM*, Philadelphia, PA, USA, Aug. 2005, pp. 133–144.

- [7] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," RFC 3626, Oct. 2003. [Online]. Available: <http://www.ietf.org/rfc/rfc3626.txt>
- [8] D. B. Johnson, Y.-C. Hu, and D. A. Maltz, "On The Dynamic Source Routing Protocol (DSR) for mobile ad hoc networks for IPv4," RFC 4728, Feb. 2007. [Online]. Available: <http://www.ietf.org/rfc/rfc4728.txt>
- [9] C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for mobile computers," *Comput. Commun. Rev.*, vol. 24, pp. 234–244, Oct. 1994.
- [10] C. E. Perkins and E. M. Royer, "Ad hoc On-Demand Distance Vector (AODV) routing," RFC 3561, Jul. 2003. [Online]. Available: <http://www.ietf.org/rfc/rfc3561.txt>
- [11] S. Murthy, "Routing in packet-switched networks using path-finding algorithms," Ph.D. dissertation, Comput. Eng., Univ. California, Santa Cruz, CA, USA, 1996.
- [12] J. Behrens and J. J. Garcia-Luna-Aceves, "Distributed, scalable routing based on link-state vectors," in *Proc. ACM Conf. SIGCOMM*, 1994, pp. 136–147.
- [13] S. Murthy and J. J. Garcia-Luna-Aceves, "An efficient routing protocol for wireless networks," *Mobile Netw. Appl.*, vol. 1, no. 2, pp. 183–197, Oct. 1996.
- [14] D. West, *Introduction to Graph Theory*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, Aug. 2000.
- [15] Z. Wang, Y. Chen, and C. Li, "CORMAN: A novel cooperative opportunistic routing scheme in mobile ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 2, pp. 289–296, Feb. 2012.



**Zehua Wang** (S'13) received the B.Eng. degree from Wuhan University, Wuhan, China, in 2009 and the M.Eng. degree from the Memorial University of Newfoundland, St John's, NL, Canada, in 2011. He is currently working toward the Ph.D. degree with The University of British Columbia, Vancouver, BC, Canada.

His research interests include machine-type communication networking, wireless ad hoc networking, mobile and distributed computing, and generic data networks.

Mr. Wang served as a member of the Technical Program Committees for the IEEE International Conference on Wireless and Mobile Computing, Networking, and Communications in 2011; the IEEE International Conference on Communications (IEEE ICC), the IEEE Global Communications Conference, and the IEEE International Conference on Connected Vehicles and Expo (IEEE ICCVE) in 2012; and the IEEE ICCVE in 2013. He will serve as a member of the Technical Program Committee for the IEEE ICCV in 2014.



computation.

**Yuanzhu Chen** (M'12) received the B.Sc. degree from Peking University, Beijing, China, in 1999 and the Ph.D. degree from Simon Fraser University, Burnaby, BC, Canada, in 2004.

From 2004 and 2005, he was a Postdoctoral Researcher with Simon Fraser University. He is currently an Associate Professor with the Department of Computer Science, Memorial University of Newfoundland, St. John's, NL, Canada. His research interests include computer networking, graph theory, web information retrieval, and evolutionary



**Cheng Li** (SM'07) received the B.Eng. and M.Eng. degrees from Harbin Institute of Technology, Harbin, China, in 1992 and 1995, respectively, and the Ph.D. degree in electrical and computer engineering from the Memorial University of Newfoundland, St. John's, NL, Canada, in 2004.

He is currently a Full Professor with the Faculty of Engineering and Applied Science, Memorial University of Newfoundland. His research interests include mobile ad hoc and wireless sensor networks, wireless communications and mobile computing, switching and routing, and broadband communication networks.

Dr. Li is a registered Professional Engineer in Canada and a member of the IEEE Communication, Computer, Vehicular Technology, and Ocean Engineering Societies. He has served as a Co-Chair of the Technical Program Committee (TPC) of the IEEE Biennial Symposium on Communications in 2010 and the IEEE International Conference on Wireless and Mobile Computing in 2011. He has served as a Co-Chair for various technical symposia of many international conferences, including the IEEE Global Communications Conference (IEEE GLOBECOM) and the IEEE International Conference on Communications (IEEE ICC). He has also served as a member of TPCs for many international conferences, including the IEEE ICC, the IEEE GLOBECOM, and the IEEE Wireless Communications and Networking Conference. He is an editorial board member of *Wiley Wireless Communications and Mobile Computing*, the *Journal of Networks*, the *International Journal of E-Health and Medical Communications*, and the *Korean Society for Internet Information Transactions on Internet and Information Systems*. He is also an Associate Editor for *Wiley Security and Communication Networks*.