Wireless Netw (2008) 14: 29–46
DOI 10.1007/s11276-006-7591-1

A framework for self-healing and optimizing routing techniques for mobile ad hoc networks*

Chao Gui · Prasant Mohapatra

Published online: 9 June 2006

Abstract On demand routing protocols provide scalable and cost-effective solutions for packet routing in mobile wireless ad hoc networks. The paths generated by these protocols may deviate far from the optimal because of the lack of knowledge about the global topology and the mobility of nodes. Routing optimality affects network performance and energy consumption, especially when the load is high. In this paper, we define routing optimality using different metrics such as path length, energy consumption along the path, and energy aware load balancing among the nodes. We then propose a framework of Self-Healing and Optimizing Routing Techniques (SHORT) for mobile ad hoc networks. While using SHORT, all the neighboring nodes monitor the route and try to optimize it if and when a better local subpath is available. Thus SHORT enhances performance in terms of bandwidth and latency without incurring any significant additional cost. In addition, SHORT can be also used to determine paths that result in low energy consumption or to optimize the residual battery power. Thus, we have detailed two broad classes of SHORT algorithms: Path-Aware SHORT and Energy-Aware SHORT. Finally, we evaluate SHORT using the *ns-2* simulator. The results demonstrate that the performance of existing routing schemes can be significantly improved using the proposed SHORT algorithms.

*This research was supported in part by the National Science Foundation under the grants CCR-0296070 and ANI-0296034, and a generous gift from the Hewlett Packard Corporation. A preliminary version of this paper appeared in the proceedings of the ACM MobiHoc 2003.

C. Gui (✉) · P. Mohapatra
Computer Science Department, University of California, Davis, Davis, CA 95616, USA
e-mail:{guic, prasant}@cs.ucdavis.edu

Keywords Ad hoc networks · Routing protocol · Self-healing and optimizing routing techniques · Energy-aware SHORT · Path-aware SHORT · Hop comparison array

1. Introduction

Mobile Ad hoc Networks (MANETs) form a class of dynamic multi-hop network consisting of a set of mobile nodes that intercommunicate on shared wireless channels. Each node in MANETs can operate as a host as well as a router. Since the mobile nodes have limited transmission capacity, distant nodes intercommunicate through multi-hop paths. The ease of deployment and the lack of the need of any infrastructure makes MANETs an attractive choice for a variety of applications. Examples of such applications include communications in battle grounds, in disaster recovery areas, among a group of islands or ships, environment monitoring using sensor nodes, conferencing without the support of any wired infrastructure, and for interactive information sharing in remote locations.

Recent proliferation of mobile wireless devices has considerably increased the need of ad hoc networking environments that connect large number of mobile nodes. Continuously changing network topology, low transmission power, and low available bandwidth are major challenges for routing in MANETs. Routing protocols should adjust to network topology changes, yet should not incur too much overheads in terms of the transmission of control messages. Normal link state routing, established in wired Internet routing, has failed to fulfill the special requirements of ad hoc routing [25]. This challenge has instigated intense research on routing in ad hoc networks in recent years.

With the increase in the size and average route length, scalability becomes an issue for the current ad hoc routing protocols. Table-driven proactive routing protocols [21] that require periodic advertisement and global dissemination of connectivity information are not suitable for large networks [17]. On-demand routing protocols are efficient for routing in large ad hoc networks because they maintain the routes that are currently needed, initiating a path discovery process whenever a route is needed for message transfer. AODV [22] and DSR [16] are two prominent ad hoc routing protocols that have used this approach. In AODV, the routing table at the nodes cache the next hop router information for a destination and use it as long as the next hop router remains active (originates or relays at least one packet for that destination within a specified time-out period). In DSR, which is a source-based routing, identity of all the intermediate nodes are included in the packet header. In large ad hoc networks, the header length could become very long. A survey of several routing protocols and their performance comparisons have been reported in [24] and [3], respectively.

Since there has been considerable effort on the design and evaluation of routing algorithms for ad hoc networks, we refrain from proposing yet another routing algorithm for MANETs. Instead, our approach can be summarized as follows. Given any routing algorithm that provides a path between source and destination, how can we optimize the path dynamically to enhance performance quality, and reduce energy consumption. We have focussed on approaches that attempt to optimize the path without incurring any significant overhead.

In this paper, we have proposed a Self-Healing and Optimizing Routing Technique (SHORT) for ad hoc networks. Using the framework, two broad classes of SHORT are proposed: Path Aware(PA)-SHORT and Energy Aware(EA)-SHORT. PA-SHORT is concerned with optimizing and healing paths to reduce the number of hops. The method is to continuously monitor a routing path, and gradually update parts it, so that it will incrementally converge to the shortest path. On the other hand, the main goal of EA-SHORT is to balance power consumption in MANETs, using the same local monitoring method as the PA-SHORT. The SHORT framework incurs very little overhead as it exploits the broadcast nature of wireless communications. We have implemented both categories of the SHORT approaches and have evaluated the benefits in terms of performance and energy conservation. The results show significant improvement compared to the base routing algorithm.

The rest of the paper is organized as follows. The performance and energy concerns that distinguishes PA-SHORT and EA-SHORT are discussed in Section 2. Detailed descriptions on PA-SHORT and EA-SHORT are reported in Sections 3 and 4, respectively. The evaluation of the SHORT algorithms are detailed in Section 5 followed by the related works in Section 6. The concluding remarks are outlined in Section 7.

2. Performance and energy concerns

In ad hoc network routing, two issues are of prime concern: performance and energy conservation. Our goal is to optimize both of these issues without incurring any significant overheads. In this section, we motivate and describe the role of SHORT in enhancing performance and energy conservation of MANETs.

2.1. Shortest path based routing

Paths generated by the on-demand ad hoc routing protocols can deviate far from the shortest path. Because of the mobility of the nodes in ad hoc networks, the shape of routing paths may change significantly while the connectivity is intact. Most of the previously proposed on-demand routing schemes do not initiate a new path discovery process until there is a link failure (a node fails or moves out of range). The changes in shape can be exploited in deriving better routing paths if we can avoid any significant overheads (at least avoid extra path discovery processes). In Section 3.1, we show some examples of such scenarios and have discussed the potential performance impact of these changes in the shape of the path. Another cause of routing non-optimality is the inclusion of the additional routing optimization techniques. Expanding Ring Search(ER) and local repair [23] are two such techniques, which are effective in reducing routing overhead. Reduction of routing overhead has become a necessity for addressing the routing scalability issues. A chapter in [17] is devoted to routing scalability issues. Results of simulations on large size networks with 10,000 nodes have shown that the average length of paths generated by on-demand routing protocols is much greater when ER and local repair options are turned on—more than double the optimal length.

Ill-formed paths that are much longer than the shortest available paths are not desirable. As data packets pass along these paths, extra bandwidth is consumed. Longer end-to-end delay and shorter route life-time are expected as well. Since the number of hops in a longer path is high, the power consumption due to higher number of broadcasts is also high. In addition, the phenomenon of self-interference by successive nodes in a route is identified in [19], which makes longer routes less attractive. A mechanism that can optimize or shorten the route length will result in significant performance gain over the underlying routing protocols.

2.2. Energy aware routing

The battery of the lightweight mobile devices usually have limited power supply. Wireless network interface has been identified as a major source of power consumption in mobile computers, along with LCD display and hard disk. Thus, when considering extending the lifetime of the mobile devices, as well as the lifetime of the whole network, energy efficiency of routing protocols is a prominent issue. Studies have been carried out aiming at finding minimum-cost multihop paths in terms of energy consumption along the path. Different power-aware metrics for determining routes have been proposed. The metrics are based on battery power level and energy consumption at each node. Our goal is to develop a framework that can enhance these power consumption metrics through route optimization. The dynamics of a network can change the condition of a wireless link rapidly. Current work in this area cannot adjust to this unpredictable changes in link quality, contention rate and so on. We aim at continuously monitoring these conditions, and try to heal and optimize the path by diverting it to a better path, if available.

Energy aware protocols extend node and network lifetime by routing packets through nodes that have sufficient remaining power, and avoiding nodes that are low on battery supply. To achieve routing fairness, we should let routing protocols to self-heal so that it can drag itself out of the situations in which certain nodes are being over-used while other nodes are idle. Energy aware protocols will be able to divert the traffic from the loaded area, balancing the load to all nodes in the network.

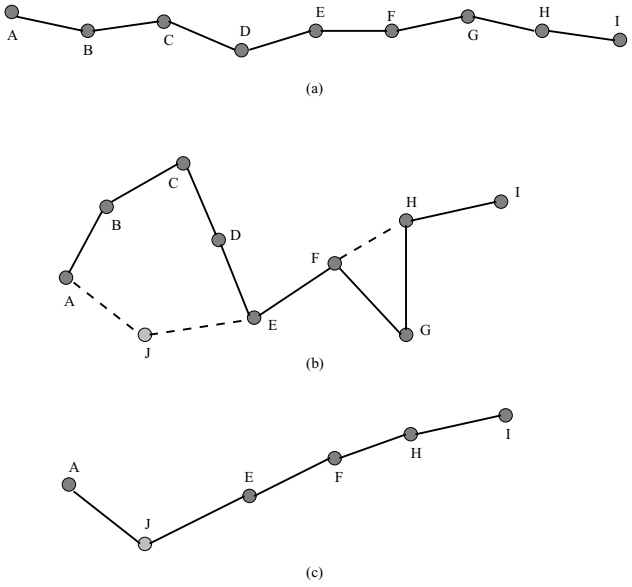


Fig. 1 An example of the changes in routing paths

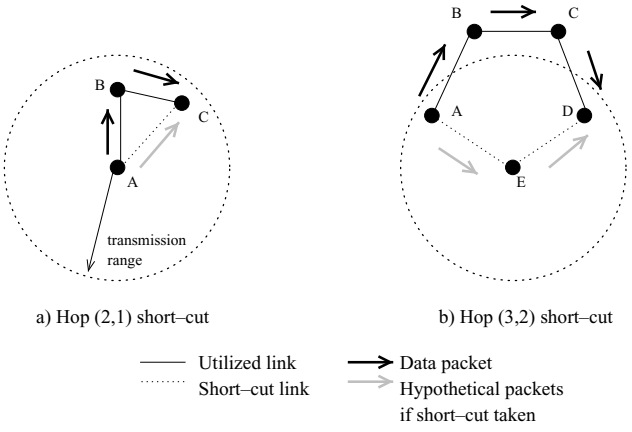


Fig. 2 Basic short-cut path formations

3. Path Aware(PA)-SHORT

3.1. Problem description

Consider a routing path from a source node A to a destination node I as shown in Fig. 1(a). This initial path is determined through the path discovery process, in which the distance between the source and destination is the shortest in terms of the number of hops, or very close to it. A packet takes eight hops while getting routed from A to I. During the course of time, the mobility of the nodes may make the shape of the routing path similar to the one shown in Fig. 1(b) while retaining the connectivity. In this new shape, J is in the transmission range of A, and E is in the transmission range of J. Similarly H is in the transmission range of F. However, because of the usage of route caches and the validity of the existing routing information, the routing table entries are not updated. Although functionally adequate, using the routing paths of Fig. 1(b), a packet still takes eight hops to reach from node A to node I. Ideally, the shortest path from A to H needs only five hops as shown in Fig. 1(c). The goal of this paper is to identify such situations and self-heal and optimize the paths dynamically by modifying the entries of the routing tables.

The primary goal of the solution approach is to discover short-cut routing paths as and when feasible. The basic scenarios of the short-cut discovery process are shown in Fig. 2. In Fig. 2(a), the path A-B-C can be reduced to A-C as C is within the transmission range of A. This short-cut path formation is termed as (2,1) reduction. In general, an $(n,1)$ reduction implies that n routing hops are reduced to only one hop. Figure 2(b) shows that the routing path A-B-C-D can be shortened to A-E-D, since E is in the range of A, and D is in the range of E. This short-cut path formation is termed as (3,2) reduction. Thus, $(n,2)$ reduction implies that n hops along the path can be reduced to only two hops. In general terms, (n, k) reduction implies that n routing hops can be reduced to k hops, where $k < n$. To avoid complexity in terms of overheads, we have considered the values of k

as 1 and 2 for the proposed SHORT algorithms. Intuitively, higher the difference between n and k, the better will be the performance of SHORT.

SHORT is applicable in conjunction with any underlying ad hoc routing protocol that formulates the entries of the routing tables. The underlying routing protocol need not have to be very efficient or optimal. It could be very simple and adequate enough to provide a reachable path from a source node to the destination node. SHORT can optimize the path by self-healing.

3.2. Significance of short-cut occurrence

Before we describe the Path Aware(PA)-SHORT algorithm, it is necessary to show that the scenarios described in the previous section in fact occur with considerable probability. In this section, we derive the probability of the existence of the short-cut paths in mobile ad hoc networks. We model an ad hoc network by a set of random points on a two-dimensional area. N points are plotted on an X-Y Cartesian plane, so that their x-coordinates and y-coordinates are uniformly distributed in the ranges of [0, X] and [0, Y], respectively. If the distance between any two points is less than 1 then they are regarded as adjacent. Among all these points, a sequence of $n+2$ given points are of interest, which are labeled as $v_0, v_1, \ldots, v_{n+1}$. We are given the condition that these nodes form a path, i.e. points v_i and v_{i+1} are adjacent for i from 0 to n. This condition limits the relative positions of v_{i+1} with respect to v_i, since we know v_{i+1} is located within a unit circle centered at v_i. But each point in the circle is equally probable to be the position of v_{i+1}, so there is still a room for freedom for the shape of the path, and for the distance between v_0 and v_{n+1}. For example, if the consecutive points form a straight line, then the distance can approach $n+1$. On the other hand, if they form a cycle, the distance can be as low as zero. The question now is: What is the probability that v_0 and v_{n+1} are adjacent. We denote this probability by $P[n]$, and term it as the probability of adjacency.

Let's consider the two basic short-cut configurations shown in Fig. 2. Using the notations in our geometric model, the probability of hop (2, 1) short-cut is just $P[1]$ in adjacency probability. For a hop (3, 2) short-cut, the adjacency probability is equal to $P[3]$, which is illustrated in Fig. 2(b). Nodes E, A, B, C, and D form a sequence of five points, which can be denoted as v_0, v_1, v_2, v_3, and v_4, respectively.

For deriving $P[1]$, let us refer to Fig. 3. Let P_1^r denote the probability that v_1 is within the shaded cycle band centered at v_0(i.e. $P_1^r = Prob[r \le |v_0v_1| \le r+\Delta r]$.) Let P_2^r denote the probability of v_2 being within both v_0's and v_1's adjacency range. The geometrical derivations of P_1^r and P_2^r are shown in Fig. 3. The value of r is within the range $[0, 1-\Delta r]$. We

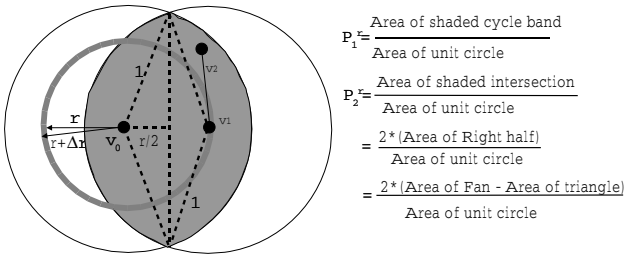


Fig. 3 Adjacency probability of three points

thus have the following equations:

$$P[1] = \lim_{\Delta r \to 0} \sum_{r=0}^{1-\Delta r} P_2^r \cdot P_1^r \tag{1}$$

$$= \int_0^1 \frac{2(\arccos \frac{r}{2} - \frac{r}{2}\sqrt{1-\frac{r^2}{4}})}{\pi} \cdot 2r dr \tag{2}$$

$$= 1 - \frac{3\sqrt{3}}{4\pi} \simeq 58.6\% \tag{3}$$

To derive values of $P[n]$ for larger n, we define the following terms. Let $R^i = |v_0v_{i+1}|$ $(0 \le i \le n)$ be a series of random variables. For each one in the series, we define its probability distribution function $f^i(r)$ as the following.

$$f^i(r) = \lim_{\Delta r \to 0} \frac{Prob[r \le |v_0v_{i+1}| \le r+\Delta r]}{\Delta r}$$

Since any point within the unit circle centered at v_0 is equally probable to be the position of v_1, we can derive the following for $f^0(r)$.

$$f^0(r) = \begin{cases} 2r & : 0 \le r \le 1 \\ 0 & : 1 < r \end{cases}$$

Referring to Fig. 4, we get the recurrence relation of distance distribution function:

$$f^i(r) = \int_0^1 \int_0^{2\pi} f^{i-1}(\sqrt{t^2+r^2-2rt\cos\theta}) f^0(t) d\theta dt$$

Using the distance distribution function, we can obtain the probability of adjacency:

$$P[n] = \int_0^1 f^n(r) dr.$$

For larger values of n, the analytical expressions of $P[n]$ becomes cumbersome. So to obtain values of $P[n]$ for larger n, we conduct random walk experiments simulated by computer. Each experiment starts from the position of v_0 and begin taking steps. For each step, the target point is equally

Table 1 Experimental results for values of P[n]

Attempts	1	2	3	4	5	6
10,000	0.59	0.47	0.38	0.32	0.28	0.24
100,000	0.59	0.46	0.37	0.31	0.27	0.24

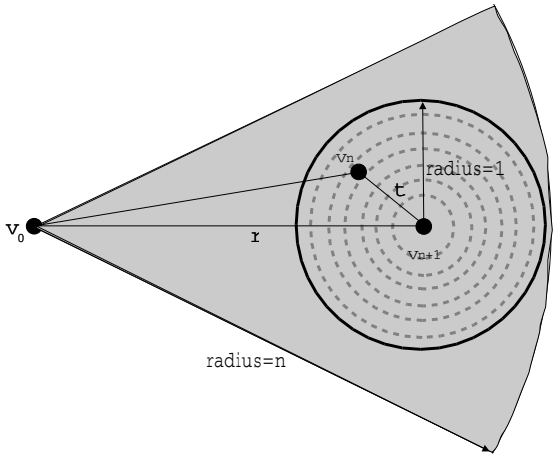


Fig. 4 Recurrence relation of $f^n(r)$

likely to be any position as long as the step length is less than 1. After taking $n+1$ steps, if the final position is still within the reach of one step range, we call the experiment successful. We repeat each experiment 10,000 and 100,000 times. Table 1 gives the frequency of success for different values of n.

This geometric probability model indicates that the probability of (2, 1) and (3, 2) short-cuts are 59% and 37%, respectively. It is based on the following assumption. For the given set of $n+1$ nodes along the n-step path, the position of the next-hop node is uniformly distributed within the unit circle centered at the current node. This is valid only with the condition that the given path has always been connected. Thus the presented values are conditional probabilities. In a mobile ad hoc network, when a routing path is set up by the routing protocol, it is more probable that the next-hop node is toward the direction of the destination node. When it further moves in that direction, the path gets broken and new route is constructed. In a dynamic mobile ad hoc network, a more realistic metric is the percentage of packets taking the short-cut available paths over all the delivered packets.

To further analyze the scenarios, we conduct the following experiments with the ns-2 simulator. (The description of general setup of the simulator is given in Section 5.) A 100-node network is simulated within a field of size 2000 m × 600 m. Mobility of the moving nodes is modeled as the *random waypoint* model [3]. One connection of UDP traffic is setup between a chosen source node S and a destination node D. Since we are only interested in the situation when the hop distance between nodes S and D are significant enough, we restrict the position and the movement of the two nodes to be at the two 500 meter-wide full height strips at each end of the field. We use AODV as the underlying routing protocol and modify it so that each packet records the actual path that it is taking towards the destination node. By consulting the GOD module of the ns-2 simulator, we can study each path and identify the availability of (n,1) and (n,2) short-cuts. With node S generating 4 traffic packets per second at constant rate for 1000 simulation seconds, we let node D count the percentage of the packets that take paths with (n,1) short-cut availability, and/or with (n,2) short-cut availability. The node mobility and node density will affect the short-cut occurrence. We first set the maximum speed to be 1, 5, 10, 15 and 20 m/s, and set the pause time to be 0, 50, 100, 150, 200 and infinite seconds. The higher the pause time, the longer each node remains stationary before the next movement. Thus, the mobility for the whole network is less for higher pause times. We then change the node density by varying the number of nodes in the network as 80, 100, 120 and 140. We run each scenario 10 times with different random seeds.

Figure 5 shows the result of the above simulations, with the bar height representing average value and error bar representing the standard deviation. We observe that about 40% of packets encounter (n,2) short-cuts, while 10% of packets encounter (n,1) short-cuts. Packets that encounter both (n,1) and (n,2) short-cuts are included in both the counters. Note that in this experiment we are measuring the "proportion of packets encountering short-cuts," rather than the "probability of a short-cut" that was evaluated in the previous analysis. A single packet could encounter several (n,1) or (n,2) short-cuts. Furthermore, in this experiment path disconnection is considered, unlike the scenario in the analytical model. It can also be observed from Fig. 5 that higher moving speed, shorter pause time, and higher node density will result in more frequent short-cut availability. In all scenarios, the occurrence of short-cuts is significant. This suggests a potential performance gain with shortest-path based PA-SHORT technique is possible. Finally, we observe that the (n,2) short-cut is much more frequent than (n,1) short-cut. Thus a routing scheme that utilizes both short-cuts will converge to the shortest path much faster than the one that uses only (n,1) short-cuts.

3.3. PA-SHORT: Detailed algorithms

SHORT is a general technique that should work with any underlying routing protocol. Due to the lack of precise global information and distributed operational manner, we should not expect routing protocols to make globally optimal decisions at the route setup time. However, as long as a valid routing path is set up, and packets begin to flow along the path, we will let SHORT monitor the path. It will try to identify

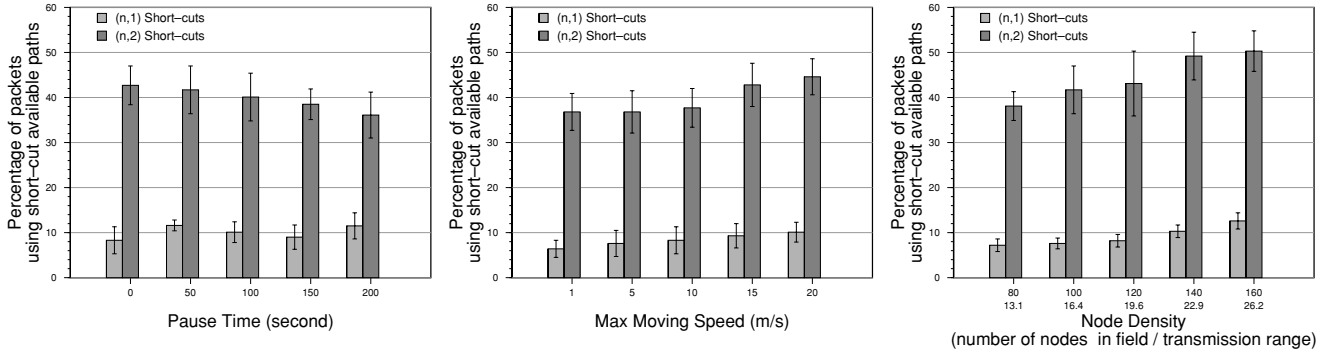


Fig. 5 Occurrence of (n,1) and/or (n,2) short-cuts with regard to node mobility pattern and node density

short-cuts on the route and shorten it according to the up-to-date topology. Different types of routing protocols work in different manner, we need to have different algorithms for short-cut identification and reaction. PA-SHORT-DV algorithm works with distance vector routing protocols such as AODV, while PA-SHORT-SR algorithm works with source routing protocols such as DSR. These two algorithms are described in this section.

3.3.1. PA-SHORT-DV algorithm

When a node wants to forward a packet to another node, it broadcasts the packet and all the nodes within the transmission range can hear the packet. Each of the nodes check the packet header to see if the packet is destined to it (as the next hop) or not. The node that is the next hop destination, captures the packet and processes it for the next hop. Other nodes disregard the packet (unless they are operating in promiscuous mode). So in any case, the neighboring nodes (nodes that are within the transmission range) get to check all the packet headers that they hear.

To support the proposed SHORT algorithm, each packet needs to carry a "hop-count (HC)" field in its header. The HC field is initialized to zero at the source node and gets incremented by one at every hop the packet takes. For every packet, the destination address (DA), source address (SA), and the HC can be obtained from the packet header. This information is maintained as an array termed as the *hop comparison array*. The format of each entry in the array is $\langle SA, DA, HC, NA\rangle$, where NA is the neighbor's address from which the packet was broadcasted (the id of the node that broadcasted the packet during the current hop). Each of the element of the array has an expiration time after which they are invalidated. If the array becomes full with valid entries, the new entries can replace the older entries using the least-recently-used (LRU) policy.

Consider a source node SA_k and a destination node DA_k. Before SA_k sends a packet to the first hop node, an entry with the following information is included in SA_k's hop comparison array: $\langle SA_k, DA_k, 0, SA_k\rangle$.

When node i receives or overhears a packet P, which belongs to the $\langle SA_k, DA_k\rangle$ connection, the steps of the PA-SHORT-DV algorithm can be enumerated as shown in Algorithm 1. If node i is the final destination, the packet is consumed without any events. If node i is the next-hop for a broadcast packet, $\langle SA_k, DA_k, HC_k, NA_k\rangle$ is recorded in the hop comparison array if no entry exist corresponding to the $\langle SA_k, DA_k\rangle$ pair. If an entry for $\langle SA_k, DA_k\rangle$ exist then nothing is updated. The comparison of HCs and the modification of the routing table (when necessary) happens only at the other nodes, i.e., excluding the current node and the next-hop node. In those nodes, if an entry corresponding to $\langle SA_k, DA_k\rangle$ does not exist then it is recorded. Otherwise the current HC is compared with the stored HC. If the difference (current HC—stored HC) is more than 2 then it implies that a short-cut path exists. The value of 2 is used because we can shorten the paths that have a minimum of 2 hops. The node that detects this possibility modifies its routing table entry and that of another appropriate node to enable the short-cut path formation.

Note that the Algorithm 1 is capable of handling both the $(n,1)$ and $(n,2)$ short-cut configuration discussed earlier in Section 3.1.

The proposed SHORT algorithm does not incur any significant overhead. Each of the nodes use a very small amount of memory space to maintain the *hop comparison array*. For every short-cut path formation, only one extra broadcast message (one time only) is necessary to inform one of the neighboring nodes to update its routing table.

Let us analyze an example to show the functionalities and validate the correctness of the algorithm. Consider the routing path shown in Fig. 1(b). We will trace through the algorithm to analyze how the routing path self-optimizes itself without any significant overhead. Assume node A as the source and node I as the destination. The current routing path is shown by the solid lines in the figure. We will track only one entry of the *hop comparison array* maintained at each of the nodes. This

Algorithm 1 PA-SHORT-DV Algorithm

When node i receives or overhears a packet P, **BEGIN**
1.**IF** the node i is the final destination address, consume the packet. **GOTO END**;
2.(Assume P belongs to $<SA_k, DA_k>$ flow.) Compare $<SA_k, DA_k>$ to all the valid entries in the hop comparison array;
3.**IF** there is no match with the entries, store $<SA_k, DA_k, HC_k, NA_k>$ in the hop comparison array;
4.**IF** the packet is destined to i as the next-hop node,process the packet for forwarding further. **GOTO END**;
5.(Assume that it matched with an entry $<SA_k, DA_k, HC_j, NA_j>$) **IF** $(HC_k - HC_j > 2)$, a short-cut is found, node i does the following:
5.1 Send a message to NA_j to update the routing table such that the next hop address for destination node DA_k is modified to the address of node i;
5.2 Modify its routing table by making the next-hop address for destination DA_k as NA_k;
5.3 Modify its hop comparison array, delete the entry corresponding to $<SA_k, DA_k>$;
END

entry corresponds to (A,I) as the source-destination pair. The example is analyzed in steps where a step defines the events during the broadcast of a packet at a node. The entry of the hop comparison array at end of each step at each of the node is shown in the Table 2. Since A and I are fixed, only (HC, NA) is shown for the hop comparison array (A, I, HC, NA).

i A needs to forward a packet to B. An entry is initialized in A's hop comparison array as (A,I,0,A). It broadcasts the packet while marking the HC as 0. B and J are in the transmission range and receive the broadcast. B is the next-hop node and does not have any entries corresponding to (A,I). So it records (A,I,0,A) in its hop comparison array. J also did not have any entry corresponding to (A,I), so it records (A,I,0,A) in its hop comparison array.

ii To forward the packet to C, B broadcasts the packet with HC=1. Only A and C are in the transmission range. C records (A,I,1,B) in its hop comparison array. A finds a match with its entry in the hop comparison array. However, the difference in the HCs is not more than 2, so nothing is updated.

iii C broadcasts the packet to forward it to D with an HC of 2. Only B and D are in range. D records (A,I,2,C), and nothing is updated at B since the difference in HCs is 2 (needs to react only when it is more than 2).

iv Similarly when D forwards the packet with only C and E in the range, the entry of the hop comparison array at E is noted as (A,I,3,D), and everything else remain the same.

v When E broadcasts the packet to send it to F, then D, F, and J are in the transmission range. Nothing is updated at D, and the entry of the hop comparison array at F is recorded as (A,I,4,E). The HC difference at node J is 4. So J sends a message to A to update its routing table so that the next hop of A for destination I is J. J modifies its own routing table to make the next-hop node for destination I as E. Thus the path A-J-E is created as a short-cut for the earlier route A-B-C-D-E. The corresponding entries at nodes A, J, and E are deleted (the reason for the deletion is explained later).

vi F broadcasts the packet with a HC of 5 destined for G. Nodes E and H also receive the packet as they are in the transmission range. E, G, and H record (A,I,5,F) in their hop comparison array.

vii G broadcasts the packet destined to H with a HC of 6. Both the receivers in the range, F and H, have an entry corresponding to (A,I). However, since the difference in HCs is not more than 2, nothing is updated.

viii While forwarding the packet to I, H broadcasts the packet with a HC of 7. I, G, and F are in the transmission range. Node I consumes the packet as it is the destination node.

Table 2 The entry of the hop comparison array

	A	B	C	D	E	F	G	H	I	J	Short-cut
i	(0,A)	(0,A)								(0,A)	
ii	(0,A)	(0,A)	(1,B)							(0,A)	
iii	(0,A)	(0,A)	(1,B)	(2,C)						(0,A)	
iv	(0,A)	(0,A)	(1,B)	(2,C)	(3,D)					(0,A)	
v	(0,A)	(0,A)	(1,B)	(2,C)	(3,D)	(4,E)				(4,E)	Found at J
vi	(0,A)	(0,A)	(1,B)	(2,C)	(3,D)	(4,E)	(5,F)	(5,F)			
vii	(0,A)	(0,A)	(1,B)	(2,C)	(3,D)	(4,E)	(5,F)	(5,F)			
viii	(0,A)	(0,A)	(1,B)	(2,C)	(3,D)	(7,H)	(5,F)	(5,F)			Found at F

G compares the HC with its stored entry. As the difference in HC is not more than 2, nothing is updated at G. At F the HC difference is 3. So F sends a message to E to update its routing table to point to F for destination I. In this case, the routing table entry already points to F, so no update is necessary. This step is redundant in this case. However, it maintains the correctness of the algorithm. F then updates its own routing table to point to H as the next-hop node for destination I. The corresponding entry of the hop comparison array at E, F, and H are deleted. Thus the path F-G-H is shortened to F-H.

ix The final path that is obtained is as shown in Fig. 1(c).

3.3.2. *Special cases*

There are some special cases in the topologies of ad hoc network which are not exploited by the basic SHORT algorithm. Additional tuning of the algorithm is necessary to capture these occurrences. However, since these occurrences would be isolated, we do not modify the algorithm to make it more complex to handle just a few cases. In any case, the algorithm remains functionally correct in such situations, although not optimized.

Consider the part of an ad hoc network shown in Fig. 6. When the source node broadcasts a message intended to some destination Q, it stores (A,Q,0,A) in its hop comparison array. Both B and C, which are in the transmission range of A, record the same information in their respective hop count arrays. Nothing is updated while B forwards the packet to C. When C forwards the packet to D, the HC is 2. When this HC is compared with the HC stored at A, the difference is not more than 2. So the short-cut route A-C is not detected. Note that this happens only when A is the source node. If A would have been a transit node, then the HC difference should have been more than 2. Thus the short-cut process of (2,1) is not detected by the source node. However, all other short-cuts–(n,1) or (n,2)—are very well detected by the source node when n is more than 2.

Another special case where the SHORT algorithm fails to discover the short-cut path involves the destination node. These isolated configurations are shown in Fig. 7. Since the destination nodes consumes the packet without any further broadcast, the short-cut paths are not detectable by the node B (in Fig. 7(a)) or node G (in Fig. 7(b)). If necessary, these isolated events can be subsumed by the proposed algorithm by enforcing the destination node to do an additional broadcast of the packet-header while consuming the packet. In that case, the node B or G would be able to detect and create the short-cut paths.

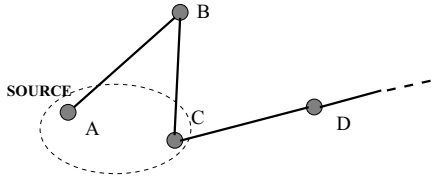


Fig. 6 A (2,1) short-cut involving the source node

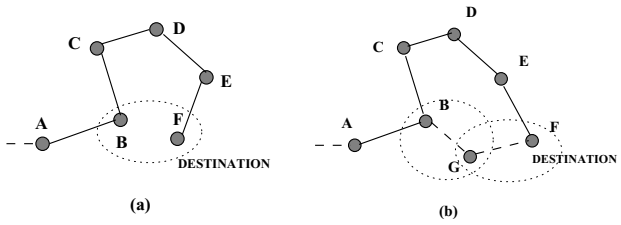


Fig. 7 Short-cut involving the destination node.

3.3.3. *PA-SHORT-SR algorithm*

In source initiated routing techniques, such as DSR [16], the entire path (i.e., the address of all the nodes that the packet will hop through) is encoded in the packet header. In [16], the authors have already shown how the (n,1) short-cut situations are handled. PA-SHORT can be also employed on top of DSR to facilitate path formation with (n,2) and higher short-cuts in addition to the (n,1) short-cuts.

According to the original specification of DSR protocol [4, 16], (n,1) short-cut is discovered as follows. Each node is in promiscuous listening mode. When node A taps a packet sent out by node B, but is not the intended next-hop receiver, node A checks the route given in the packet to see if it is in the up-coming part of the list of intermediate nodes on the route. If A is on the list and it is more than 2 hops away from B on the routing path, A finds a short-cut. A then sends a gratuitous RREP message to the source node of the route informing that A to B part of the route can be reduced to link A-B. The gratuitous RREP message travels back along the shortened path in reverse direction.

The (n,2) short-cuts can be discovered in a similar way. We assume that each node can maintain a list current neighbors from the packets it listens. When node A listens to a packet sent by node B, node A put B into its neighbor list. Then A checks the route given in the packet to see if any of its neighbors, except B, is in the up-coming part of the routing path. If node C, a node in A's neighbor list, is in the path more than 3 hops away from B, node A finds a short-cut. Node A then send out the gratuitous RREP message. The detailed description is given in Algorithm 2. We use $SR_P(S, D)$ to denote the source route path given in the header of packet P. $SR_P(j, D)$ denotes the sub-path of $SR_P(S, D)$ from node j to destination node D, given node j is on the path. Note that (n,2) and higher short-cuts are not incorporated in DSR, but can be healed using SHORT. As shown in Section 3.2, the

Algorithm 2 PA-SHORT-SR Algorithm

When node i receives or overhears a packet P from node j, **BEGIN**
1. **IF** node i is the final destination node, consume the packet. GOTO **END**;
2. Node i updates neighbor list $NB(i)$ for node j;
3. **IF** packet P is destined to i as next-hop node, node i passes P to underlying routing protocol; GOTO **END**;
4. **IF** $(SR_P(j, D) \cap NB(i) = \Phi)$ GOTO **END**;
5. **IF** exist node k in $SR_P(j, D) \cap NB(i)$, and $(HC_k - HC_j > 2)$, node i found a short-cut. Node i send gratuitous RREP message to source node S, telling sub-path $SR_P(j, k)$ can be replaced by short-cut (j-i-k); **END**

proportion of available (n,2) short-cuts is much more than that of (n,1).

3.4. Discussion on implementation

When implementing SHORT-DV algorithm, we find a need to control the aggressiveness of the algorithm. Without this curbing, short-cut messages interfere with the routing tables, causing much more route updates than the original protocols. This problem is caused by ephemeral short-cuts and multiple short-cuts. Ephemeral short-cut happens when a fast-moving node finds itself in a position to form a short-cut and informs the involved on-route nodes, but actually it only can stay in the effective short-cut position range for a short time. Multiple short-cuts means that more than one node stays within the position range of one certain short-cut. So they all report to the same involved on-route nodes, which causes the informed node to change the relevant routing table entry too frequently. However, responding only one short-cut report is enough. For a node to identify ephemeral short-cut, it needs to know its position and moving speed. Such support may not be available for the ad hoc network. Multiple short-cut problem cannot be prevented without introducing extra overhead of message exchanging. In other words, it is not easy to control the credit of short-cut messages. We can resort to the receiver side to solve this problem by making each node conservative in accepting short-cuts. We introduce the concept of a *stable period* for any newly entered or updated entry in routing table, so that each node ignores any update to it within the *stable period*. In this way, we can protect the consistency of routing tables. The fine tuning of this parameter and their consequences are discussed in the following section.

Source routing protocols are more immune to interferences from short-cuts. In SHORT-SR, each short-cut takes the form of a new route. The new route is sent back to the source node, which puts it into the node's route cache. So multiple short-cuts do not impose problems for SHORT-SR, since this only leads to the growth of route cache at the source node. However, ephemeral short-cuts are not tolerable. If a short-cut becomes invalid before the informing message gets to the source node, the route cache at the source node is polluted with dirty information. We address this problem by assuming temporal locality [7], which means the longer a node has stayed within a valid short-cut position range the more is the probability that it can form a stable short-cut path.

3.5. Extending PA-SHORT to EE-SHORT

Often there are more than one shortest path between a node pair. Merely choosing one path at random is not enough. De Couto's experimental testbed [8] shows that the shortest paths between each node pair pose a wide range in packet throughput. We also need to consider other paths as well. For example, there are variable-power scenario where nodes can adjust their transmitting power levels according to physical conditions. Here, energy efficient routing requires choosing the path that minimizes the sum of energy needed for its hops. The shortest path based PA-SHORT merely regard the cost of each link as one unit, optimizing towards minimum hop paths. By carefully choosing a link cost function, we can extend PA-SHORT so that more metrics are considered in achieving routing optimality.

In variable-power scenario, a node chooses transmitting power depending on the distance to the receiving node. In general signal model for wireless links, the energy consumption is as the following.

$$E_{opt}(d) = ad^{\alpha} + c, \tag{4}$$

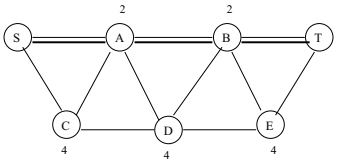
where a and c are technology-specific constant values, and α is a constant typically around 2 for short distances. Because of this non-linear signal attenuation, delivering a packet via more shorter hops is more energy efficient. Power-Aware Routing Optimization(PARO) [12,13] is designed specially for reducing transmission energy. A similar self-optimizing technique is used in PARO. A candidate node monitors the routing path of a flow, and evaluate the potential power saving by inserting itself into the path. In effect, it will replace a direct hop by two smaller hops through itself. In addition to the energy spent in a single transmission, Banerjee et al. [1] proposes an extended metric that includes energy expended

for any necessary retransmission. Thus a link cost function is defined as the following.

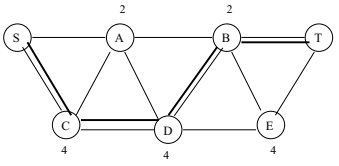
$$C_{i,j}^{approx} = \frac{E_{i,j}}{1 - p_{i,j}}, \tag{5}$$

where $C_{i,j}^{approx}$ is the cost of the link between node i and j, $E_{i,j}$ is energy needed for single packet transmission, and $p_{i,j}$ is link error rate. De Couto et al. [8] gives a similar path metric which is the expected total number of transmissions along the path, including for forwarding and retransmissions.

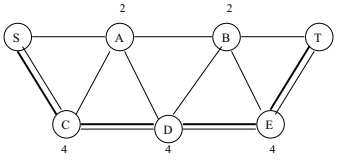
We have studied the details of shortest path based PA-SHORT. PA-SHORT algorithms can be extended to support energy efficient(EE)-SHORT. We need to extend the *hop count* field in the packet header into the field of *link cost accumulator*. Real values, instead of integers, will be filled in the field. Accordingly, the *hop count comparison array* maintained at every node is extended to *link cost comparison array*. We use Fig. 2(b) as an example for explaining the EE-SHORT-DV algorithm. When node E overhears the same packet both from node A and D, it identifies the alternative subpath A-E-D. Node E gets the value of the *link cost accumulator* field of the packets overheard from node A and D. Their difference indicates the cost of subpath A-B-C-D. After calculating link cost values for both links A-E and E-D, node E has the cost of the alternative subpath A-E-D. By comparing costs of both subpaths, node E can decide if the alternative subpath can be used to replace the original one. If the difference is greater than a previously set threshold value, redirection can be carried out by node E, which optimizes the routing path according to the adopted optimality metric.



a) Initial path that drains the residual energy at nodes A and B.

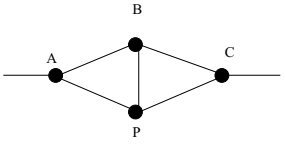


b) After route redirection, node A is circumvented.

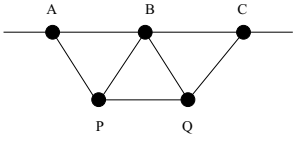


c) After route redirection, node B is circumvented. A node disjoint new path is found.

Fig. 8 Example of successive local route redirection operations. (The links shown in the figure are wireless links)



Case 1: 2–step alternate sub–path, A–P–C



Case 2: 3–step alternate sub–path, A–P–Q–C

Fig. 9 Basic cases of alternative sub-paths. (The links shown in the figure are wireless links)

4. Energy Aware(EA)-SHORT

In this section, we consider energy as the primary resource constraint. The use of SHORT framework for energy conservation is discussed in this section.

4.1. Energy aware load balancing

The goal of energy aware load balancing is to fairly distribute the traffic load among all the participating nodes in the network. For example, consider a part of an MANET shown in Fig. 8(a), the path S-A-B-T is the optimal path for a connection from S to T. The metric for optimality can be hop count for shortest path routing. Thus, nodes A and B will continuously be used in forwarding the traffic, leaving the other nodes free from the traffic load. As a result shown in the figure, the residual energy level of each node which is indicated adjacent to the nodes in Fig. 8(a) become widely varied. If the routing is not energy-aware, it will keep using the path for S-T connection. Nodes A and B will eventually be drained out of battery supply and die early. However, an energy-aware routing scheme will try to divert the traffic to other nodes. Here we propose the EA-SHORT scheme based on route redirection. With successive local redirection operations, the route will gradually converge to an alternative node disjoint path.

Figure 9 shows two basic redirections. In case 1, nodes A, B, and C are three consecutive nodes on the path for S-T connection. Node P is a nearby node which is a common neighbor of all three nodes. In a network that is dense enough to maintain continuous network connectivity, this scenario could occur frequently. As the data packets are successively forwarded by the three nodes (A,B, and C), node P can overhear the same packet three times. With careful bookkeeping, node P can identify this scenario and realize that it can replace node B for the connection, namely, the subpath A-B-C can be redirected to A-P-C. If node P sees the current energy level of node B, and find out that the difference of energy level at node B and itself is significant enough

(power level of P is more than that of B), node P will do the redirection.

In case 2 shown in Fig. 9, we suppose that the energy level difference between B and P is significant enough. The same is true between node Q and B. As a data packet travels along nodes A, B and C, node P will find out that node B needs to be circumvented. Node P also knows that it is a neighbor of both node B and its up-stream on-route neighbor, namely, node A. Similarly, node Q will find out that node B needs to be circumvented and node Q is adjacent with the down-stream on-route neighbor of node B, namely node C. We see that even from the locally overheard information, nodes P and Q can distinguish their different roles. It is up to node Q to find the existence of node P. To facilitate this process, Q broadcasts a message saying that "Node B needs help, and I am its down-stream helper, who is its up-stream helper?" If the up-stream helper, node P, gets this message, it replies with an acknowledgement. The wireless link between P and Q is thus identified. The two helper nodes will do the redirection to replace the A-B-C sub-path by the new A-P-Q-C path.

Consider the example shown in Fig. 8(a), nodes A and B both have relatively low energy level. After one redirection of case 2, node A is circumvented and the S-A-B-T path becomes S-C-D-B-T, shown in Fig. 8(b). Then, after a case 1 redirection takes effect, the new S-C-D-E-T path will replace the original one, as shown in Fig. 8(c).

4.2. EA-SHORT: detailed algorithm

To facilitate the implementation of EA-SHORT, we need to add two more fields to the conventional IP-header of each packet. They are "Hop Counter" and "Residual Energy Level." For a packet P, we use hc(P) and lvl(P) to represent the two additional fields of the packet, respectively. The algorithm needs to access other fields in a packet, such as the source address, destination address, sender and sequence number. Similarly, in the algorithm, they are represented by s(P), d(P), nid(P) and seq(P). We use s-d(P) to represent the source-destination pair of the flow that the packet belongs to.

An "overhear table" is maintained at each node. Each entry in the table contains the following three fields: source-destination pair, sequence number and "overhear list." Each entry represents a traffic flow in the network, identified by the source-destination pair field. Once a node overhears a packet P, it checks its source and destination fields. If the packet belongs to a flow new to this node, a new entry is added in the overhear table. Otherwise, let e be the overhear table entry representing the flow. We use s-d(e), seq(e) and ovlist(e) to represent its three fields. The ovlist(e) only collects packets of the flow carrying the same sequence number identified by seq(e). Thus, if seq(P) is less than seq(e), the packet is ignored. If seq(P) is greater than seq(e), seq(e) is updated to seq(P) and current entries in ovlist(e) are all deleted. Only if seq(P) is equal to seq(e), a new entry representing packet P is added. The new entry has three fields as hop counter, residual energy level, and the sender. The residual energy field records the residual energy level at transmitting node of the packet. The detailed description of EA-SHORT is given in Algorithm 3.

5. Performance evaluation

Our simulation is based on *ns-2* network simulator developed by the VINT [33] project, and further extended by Monarch [34] project to support wireless network. The simulator provides a proper model for signal propagation, and the radio model is based on Lucent Technologies WaveLAN product, with 2Mbps of transmission rate and 250 meter of transmission range. The IEEE 802.11 is simulated at the MAC layer, with implementation of the distributed coordination function(DCF). Next, we show the results obtained for both PA-SHORT and EA-SHORT algorithms.

5.1. PA-SHORT

5.1.1. Simulation setup

In this simulation we let 100 mobile nodes moving within a rectangular field of 2000 m$\times$600 m in size. We choose this rectangular field so that the average number of hops between any two nodes will be larger than that of a square field with same area. The mobility model uses the *random waypoint* model [3]. We change mobility rate by setting different values of pause time to 45, 90 180, 270, 540 and 720 simulation seconds. We also have a scenario in which all nodes are static. The maximum moving speed is 10 m/s. For the traffic model, we use 16 simultaneous flows with source-destination pairs spreading randomly in the whole network. Traffic sources are constant-bit-rate, sending 4 UDP packets per second. Each packet is 256 bytes long, thus resulting in 8 K bps data transfer rate for each flow.

Camp et al. [5] have identified the initial unstable time of random waypoint model, which causes significant room for unpredictable variability in performance results. To avoid this, we have each simulation running for 2010 seconds. The network traffic starts from 1000 second and lasts till 2000 seconds, ignoring the beginning 1000 seconds for passing the unstable state and last 10 seconds for cooling down.

DSR and AODV are two prominent ad hoc routing protocols implemented in the ns-2 simulator. We implemented the proposed algorithms on both AODV and DSR. The enhanced version of the both protocols are referred to as AODV-SHORT and DSR-SHORT, respectively. The goal of this experiment is to see how the SHORT algorithms

Algorithm 3 EA-SHORT Algorithm

```
When node i overhears a packet P,
BEGIN
1.Compare s-d(P) with any valid entry in overhear table;
2.IF there is no match with the entries, add one entry e' as the following:
s-d(e')=s-d(P), seq(e')=seq(P), ov-list(e') initialized with first entry
<hc(P),lvl(P),nid(P)>.  GOTO END;
3.(Assume a match is found at entry e.)  IF seq(P)<seq(e), ignore P. GOTO END;
4.IF seq(P)>seq(e), update e as the following:  seq(e)=seq(P), ovlist(e) reset as
having only one entry <hc(P),lvl(P),nid(P)>.  GOTO END;
5.IF seq(P)==seq(e), do the following:
    5-1.Add entry <hc(P),lvl(P),nid(P)> into ovlist(e);
    5-2.IF ovlist(e) has three entries A, B, C satisfying the following
conditions, a better subpath is found.
1)hc(C)==hc(B)+1==hc(A)+2;
2)lvl(node i)≥MAX(lvl(A),lvl(C));
3)(lvl(node i)-lvl(B))≥2.
Activate this new subpath by updating routing tables.  Delete entry e from
overhear table.  GOTO END;
    5-3.IF ovlist(e) has two entries A and B, such that hc(B)==hc(A)+1 and
lvl(node i) ≥ MAX(lvl(A),lvl(B)+2), add this indicator I in the WaitingIndicator
list:  candidate(I)=B, seq(I)=seq(e), s-d(I)=s-d(e).  GOTO END;
    5-4.IF ovlist(e) has two entries B and C, such that hc(C)==hc(B)+1 and
lvl(node i) ≤ MAX(lvl(B)+2,lvl(C)), node i broadcast one SHORT informing packet
Q as follows:  candidate(Q)=B, seq(Q)=seq(e) s-d(Q)=s-d(e);
END

When node i receives a SHORT informing packet Q, BEGIN
1.  Compare fields of Q with any valid entry in WaitingIndicator list;
2.  IF there is no match with the entries, ignore packet Q;
  ELSE a better subpath is found.  Activate this new subpath by updating routing
tables;
END
```

interact with different types of on-demand routing protocols. For each route discovery, first RREQ packet has a small *time-to-live*(TTL) value given by TTL_START constant. In every subsequent RREQ packet, the TTL field is increased by a value according to TTL_INCREASE. When that value reaches a TTL_THRESHOLD, a network wide flooding is finally used. In the simulation, we have used values of 1, 2, and 7 for TTL_START, TTL_INCREASE, and TTL_THRESHOLD, respectively.

5.1.2. *Path optimality*

We define *path optimality (or route optimality)* as the ratio of the number of hops a packet took to reach its destination over the shortest number of hops between the source-destination pair at the time the packet is sent. We average this value over all received packets during a simulation run. For an ideal case when every packet takes the shortest path, this metric will be 1. In general, a value closer to 1 indicates better routing optimality for the routing protocol.

Figure 10 shows the average path optimality as a function with respect to pause time. Starting from no mobility

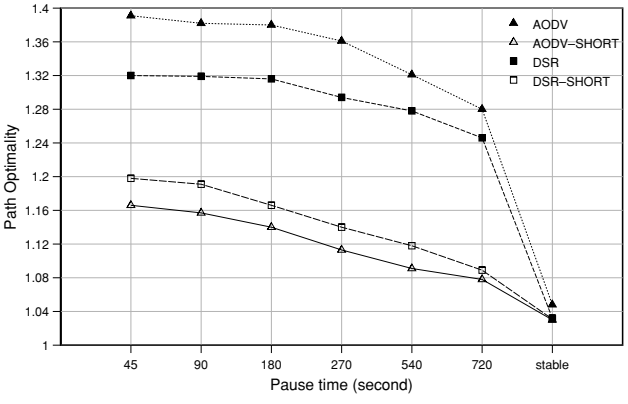


Fig. 10 Path optimality

scenario to higher mobility rate, AODV significantly drops in its routing optimality. This problem is corrected by the SHORT algorithm. For all pause time values, the change in routing optimality is not significant. Note that even at no mobility scenario, AODV-SHORT still achieves better optimality than AODV. This is because AODV does not guarantee shortest routing path even when all nodes are stable, but during the simulation run, SHORT will gradually heal and rectify if there is a short-cut path, thus resulting in better path optimality.

DSR, on the other hand, achieves better performance in terms of path optimality than AODV. This is because it already enjoys (n,1) short-cut support, which also explains that DSR has the same performance as DSR-SHORT for stable (no movement) scenario. We notice that in general DSR-SHORT does not achieve as good a gain in path optimality as AODV-SHORT. This is because in DSR-SHORT, a short-cut will not take into effect until the informing message gets to the source node, which is a relatively long delay that does not exist in AODV-SHORT. However, the noticeable performance gap between DSR and DSR-SHORT indicates a similar fact shown in Fig. 5. The (n,2) short-cuts are much more frequently available than (n,1) short-cuts. The DSR-SHORT, using (n,2) short-cuts as well as as (n,1) short-cuts, converges to the optimal paths much faster than the DSR scheme alone.

5.1.3. Delivery rate

Packet delivery ratio of a flow is the ratio of number of packets that are received by the destination over the number of packets submitted to the network by the CBR source. We look at the overall delivery ratio averaged over all 16 sessions. Figure 11 shows the overall delivery ratio with respect to the pause time. At low mobility rate, there is not much difference when SHORT is added. As the mobility rate increases, AODV's performance drops faster than DSR, which is reflected by the significant drop of path optimality in AODV. In this simulation, the local repair option is turned off in the AODV implementation. It will achieve better delivery rate with the option turned on, but the path optimality will become worse [17]. Both AODV-SHORT and DSR-SHORT achieves a stable performance at all mobility rates. This result conforms with the analysis in previous sections, that the loss rate is proportional to the route length as the loss probability increases with each additional hop.

5.1.4. Routing overhead

We look at routing overheads in two aspects: the total number of routing packets transmitted and the total number of route request packets initiated during the simulation. For each routing packets sent over the multi-hop path, each hop is counted as one transmission. For DSR-SHORT and AODV-SHORT, routing overhead also includes short-cut informing messages as well. For the number of route requests, we only count its initiation at source nodes, and its re-broadcast at other nodes is not counted. Figure 12 compares the normalized routing overhead, which is the the total number of routing packets divided by the total number of delivered packets. Figure 13 compares the number of route request. We see that the AODV-SHORT and DSR-SHORT are very close in performance, while there is a noticeable difference between AODV and DSR. The performance gain of AODV-SHORT is more significant than DSR-SHORT. Another observation is that the shorter the pause time, the greater is the performance gap between normal protocols and SHORT-enhanced protocols. This is mainly because the difference of route life-time with different hop length is much greater in high mobility scenario than in low mobility scenario. To some extent, SHORT refreshes the routing table with more current entries, which will show more effect in high mobility scenario.

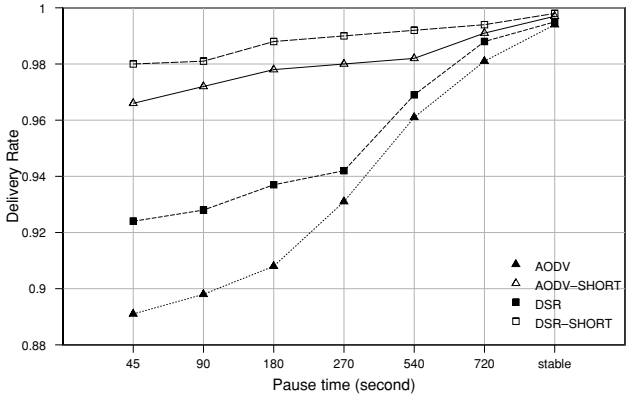


Fig. 11 Delivery rate

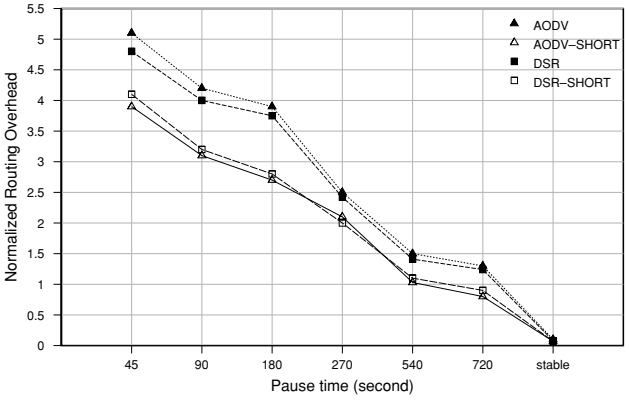


Fig. 12 Normalized routing overhead

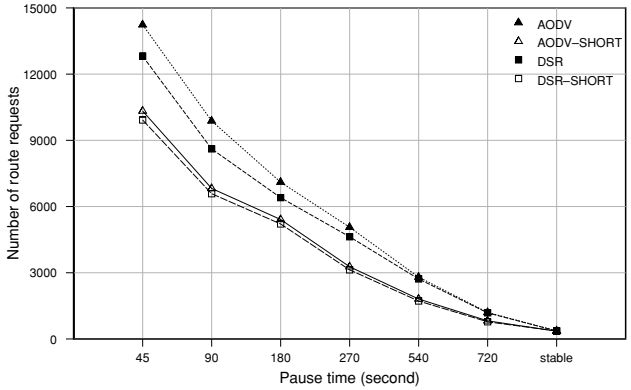


Fig. 13 Number of route requests

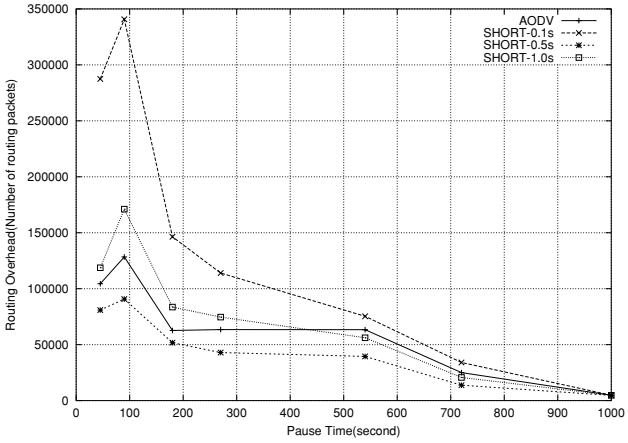


Fig. 14 Number of routing packets

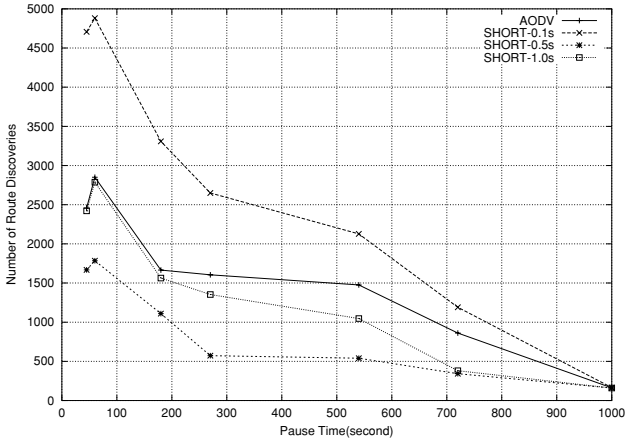


Fig. 15 Number of route discoveries

5.1.5. Stability of SHORT algorithm

In Section 3.4, we discussed the controlling of aggressiveness in implementing SHORT-DV algorithm. We now study the stability of the algorithm. Figure 5.1.5 shows the comparison of routing overhead on AODV and AODV-SHORT. With SHORT algorithm, we set the route stable time as 0.1, 0.5 and 1.0 seconds (and they are refered as SHORT-0.1, SHORT-0.5 and SHORT-1.0 respectively). These figures show that routing overhead changes dramatically with route stable time when mobility ratio is high. SHORT-0.1 performs badly at hight mobility rate. It heavily interferes with the correctness of AODV routing, initiating much more route requests than other protocols. As the the mobility rate becomes lower, the influence of SHORT algorithm decreases, until it becomes nearly zero when mobilty is zero. SHORT-1.0 costs a similar overhead with original AODV. It always requires less number of route discoveries than AODV, but its overhead is a little higher in high mobility. This extra overhead comes from short-cut informing messages. But when mobility becomes lower, SHORT-1.0 out-performs original AODV. SHORT-0.5 has the best performence. Its overhead is stable with respect to the change of mobility rate.

5.2. EA-SHORT

5.2.1. Simulation setup

The goal of EA-SHORT is to balance the energy consumption at all nodes in the MANET. The impact of EA-SHORT is more prominent during the occurrence of non-uniform traffic patterns and hot spots. Thus to demonstrate the effectiveness and impact of EA-SHORT, we have created a non-uniform traffic scenario. In this simulation, we choose a network with 120 nodes within a square field of size 1200 m $\times$ 1200 m. 100 nodes are silent and moving freely around the whole field, only forwarding the traffic packets. They are called forwarding nodes. The other 20 nodes are traffic nodes, 10 of which are traffic source nodes and 10 are sink nodes. The traffic nodes do not participate in routing. To create a congested traffic scenario, we restrict the position and movement of the traffic nodes as the following. We first evenly divide the square field into 9 (3 by 3) sub-squares, as shown in Figure 16. We let 5 source nodes be restrained within the upper-middle sub-square, and the correspondent 5 destination nodes are all restrained within the lower-middle sub-square. The remaining 5 traffic flows are also restrained to be from the middle-left sub-square to the middle-right sub-square. In this way, the central sub-square becomes a hot area. If a shortest path routing protocol is applied, the nodes staying in this area will be responsible for forwarding much more traffic packets than nodes elsewhere.

We implement EA-SHORT algorithm on AODV, and will call it AODV-EA for comparing with original AODV. The maximum moving speed is chosen to be 1, 10, and 20 m/s. The pause time is chosen to be 0, 200, and infinite seconds. Each source node has a constant bit rate traffic source, generating 4, 6, 8, and 10 packets per second. Each packet is

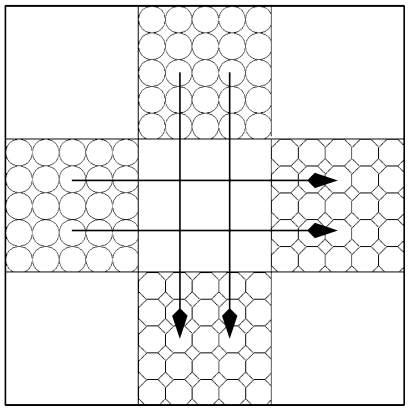


Fig. 16 Network field setup for congested load

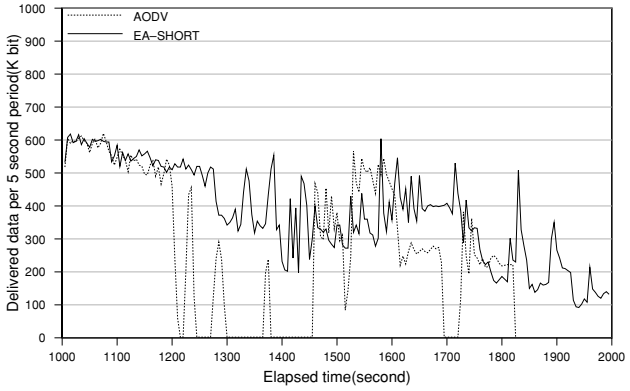


Fig. 17 Time-line of network capacity (Delivered data per 5 second period)

256 byes, resulting in 8k, 12k, 16k, and 20k bps rate at each source.

5.2.2. Network capacity time-line

In this experiment, we assume all forwarding nodes have only limited energy, namely 80 Joules. After expending its energy, the node can no longer forward any packet. At the destination nodes, we count how much data in K-bytes are received during each 5 second period. We define the network capacity as the sum of all received data for all destination nodes. We conduct simulations varying the pause time, moving speed and bit rate, as previous described. Figure 17 shows the result with 200 seconds pause time, 10 m/s moving speed and 6 packets per second per flow.

We compare the time-line of network capacity for both AODV and AODV-EA. At the beginning 200 seconds, both protocols have similar capacity. However, after 1210 seconds time, the data flow routed by AODV begins to suffer interruptions. The delivery rate can rapidly drop to zero, which means all 10 data flows are interrupted. When data delivery resumes, the rate is not sustainable. There are two major up times at the 1450 to 1695 seconds time range and the 1720 to 1825 seconds time range. The interruptions in delivery capacity is due to the network partitions. AODV makes the center sub-square an hot area, the nodes that stayed long in it will be over-used and drain out of energy early. Eventually, there is a much lower distribution of forwarding nodes in the central area.

Our goal with AODV-EA is to slow down the node dying rate by balancing the traffic load. Thus the network lifetime is extended. The delivered data remains above 500 Kbit for the first 250 seconds. We also see that the data flow is still being sustained till the simulation ends. While in AODV, it stops at 1825 seconds.

5.2.3. Nodal distribution of consumed energy

In this experiment, we assume all the nodes have infinite energy to sustain the 1000 seconds simulation time. At the end, we log how much energy has been consumed at each of the forwarding nodes. In Figure 5.2.3, the x-axis represents the consumed energy, in Joules, at the forwarding nodes, and the y-axis represents the nodal distribution of the consumed energy, in cumulative percentages. That is, a point (x, y) on the curve represents that x percent of all forwarding nodes consumed less than or equal to y Joules of energy. We conduct simulations varying the pause time, moving speed and bit rate, as previous described. The scenarios of 1 m/s and 6 packets per second with 0 and 200 seconds pause time are shown in Figure 5.2.3.

The horizontal span of a curve represents how evenly the traffic load is distributed. In case of perfectly even distribution, the curve is vertical, with zero horizontal span. We see that EA-SHORT has much less span, compared with AODV. With 0 pause time, the range of EA-SHORT is [63,132] Joule, with 68% of the nodes fitting in the [80,120] Joule range. The range of AODV is [28,170] Joule, with the bottom 10% of nodes in the [28,69] Joule range and top 10% in the [153,170] Joule.

The area of the region between the curve and the y-axis is proportional to the total energy consumed by all forwarding nodes in the network. We see that EA-SHORT spends much less energy than AODV. EA-SHORT is trying to divert the traffic away from the hot center area using longer paths, which will increase the energy consumption. On the other hand, traffic are more evenly distributed among the forwarding nodes, resulting less collisions in packet transmission. The amount of total consumed energy among all forwarding nodes is proportional to the total number of transmissions of a packet along the path, including the forwarding and retransmission. Taking this into consideration, AODV-EA eventually out-performances AODV.

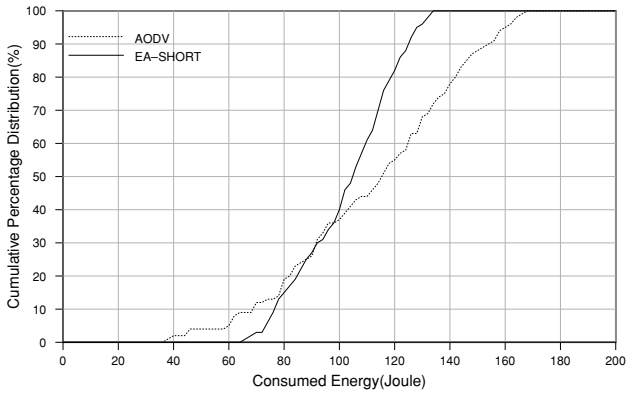


Fig. 18 Pause time is 0 sec

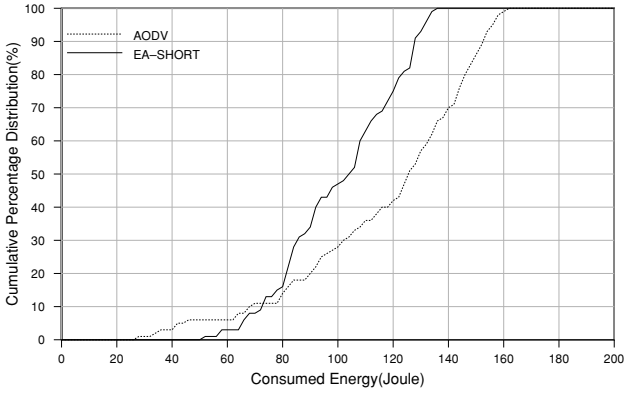


Fig. 19 Pause time is 200 sec

6. Related work

In source routing schemes such as DSR, intermediate on-route nodes have information about the entire route. Caching of these information provides nodes with the knowledge of the topology of nearby nodes [14]. Thus routes can be found within this topology and (n,1) short-cuts can be easily found as well. But (n,2) short-cuts are not supported, and this caching scheme cannot work with other protocols like AODV [22] or TORA [20]. Neighborhood awareness is exploited in neighborhood-aware source routing(NSR) [28]. Topology of nodes within two hops are learned from the source routing protocol. This information is used to repair a broken route by replacing the broken link with an alternative short route. AODV-BR [18] has some similarity to SHORT in the sense that it lets neighboring nodes around a route be aware of the route. Each neighboring node computes an alternative short route for a nearby link on the route, and when the link breaks, relevant neighboring node will salvage the route and help to deliver the affected packets to the destination.

Several reports have proposed techniques toward improving the route stability and reliability. Associativity-Based Routing (ABR) [31] protocol defines a routing metric known as degree of association stability. A high value of the metric may indicate a low state of relative mobility between neighboring nodes. In ABR, a route is selected based on the degree of association stability of mobile nodes, so that it can derive longer-lived routes. A similar approach is used in Signal Stability-Based Adaptive Routing(SSR) [9], in which route selection is based on the signal strength between nodes and a node's location stability. Routes that have "stronger" connectivity are preferred, which may also result in more stable route. Mobility prediction [30] is a different technique toward enhancing route stability. Utilizing the position and velocity information provided by GPS support, the expected expiration time of each link can be calculated, and the future state of network topology can be predicted. Based on this prediction, route can be reconstructed before it breaks, so that the disruptions caused by the changing topology are minimized. Nodes that are at the intersection of several routes might fail, which is a great harm to the routing reliability. A reliable routing framework [35] uses a set of reliable nodes deployed in the network to tackle this problem.

Besides [1,12,13], there are many studies on energy-efficient routing and power-aware routing. The difference between energy-efficient and power-aware communications in MANETs is described in [32]. Singh et al. [27] survey different routing metrics for determining routes. These metrics are based on battery power level at each node and necessary transmission energy at each link. Brown et al. [2] studied the fairless of different power aware routing objectives. Network flow-life curve is defined which serves as a new metric for power aware routing objective. With GPS support, a novel link cost function is proposed in [29]. The cost function serves as a localized power aware routing, which includes not only the necessary transmission energy but also the distance between the transmitting node and the final destination node.

7. Conclusion and future work

In this paper, we proposed a framework of self-healing and optimizing routing techniques for mobile ad hoc networks. SHORT improves routing optimality by monitoring routing paths continuously, and gradually redirecting the path towards a currently more optimal one. The basic idea is to let neighboring nodes of a routing path, together with the on-route nodes, monitor the route, so that up-to-date information about relative local topology and link quality is exploited. Furthermore, the same technique is also adapted for energy conservation during the routing in ad hoc networks. As and when better subpath become available, and is estimated to be stable, it will be utilized to redirect the route. We have analyzed and evaluated various SHORT techniques for both AODV and DSR algorithms, using the *ns* simulator. Simulation results show that higher

delivery rate and longer network lifetime are achieved by adopting SHORT. An attractive feature of SHORT is the low overheads it incurs while healing the path for improving the performance and power consumption. We can also use other metrics in self-optimizing operations, such as the QoS requirements. This will be the theme of our future work. In addition, we plan to evaluate the effectiveness of SHORT for optimizing multicast tree/mesh constructions and maintenance.

References

1. S. Banerjee and A. Misra, "Minimum energy paths for reliable communication in multi-hop wireless networks," in: *Proceedings of ACM International Conference on Mobile Computing and Networking (MOBIHOC)* (2002).
2. T.X. Brown, H.N. Gabow and Q. Zhang, "Maximum flow-life curve for a wireless ad hoc network," in: *Proceedings of ACM International Conference on Mobile Computing and Networking (MOBIHOC)* (2001).
3. J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in: *ACM International Conference on Mobile Computing and Networking (MOBICOM)* (1998).
4. J. Broch, D.B. Johnson and D.A. Maltz, "The dynamic source routing protocol for mobile ad hoc networks," IETF Internet draft, draft-ietf-manet-dsr-01.txt (Dec. 1998) (work in progress.)
5. T. Camp, J. Boleng and V. Davies, "A survey of mobility models for ad hoc network research," Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, Vol. 2, No. 5 (2002) pp. 483–502.
6. R. Castaneda and S.R. Das, "Query localization techniques for on-demand routing protocols in ad hoc networks," in: *Proc. ACM International Conference on Mobile Computing and Networking (MOBICOM)* (1999).
7. S. Chen and K. Nahrstedt, "Distributed quality-of-service routing in ad-hoc networks," in: *IEEE Journal of Selected Areas on Communications*, Vol. 17, No. 8 (Aug. 1999).
8. D. De Couto, D. Aguayo, B.A. Chambers and R. Morris, "Performance of multihop wireless networks: shortest path is not enough," in: *Proceedings of HOTNET* (2002).
9. R. Dube, C.D. Rais, K.-Y. Wang and S.K. Tripathi, "Signal stability based adaptive routing (SSA) for ad-hoc mobile networks," in: *IEEE Personal Communications* (Feb. 1997) pp. 36–45.
10. K. Fall and K. Varadhan (Eds.), *ns* notes and documentation (2002) available from http://www-mash.cs.berkeley.edu/ns/.
11. D. Ganesan, R. Govindan, S. Shenker and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," in: *Proc. ACM International Symposium on Mobile Ad Hoc Networking & Computing (MOBIHOC)* (2001).
12. J. Gomez and A. Campbell, "Power-aware routing optimization for wireless ad hoc networks," in: *Proceedings of High Speed Networks Workshop (HSN)* (June, 2001).
13. J. Gomez, A.T. Campbell, M. Naghshineh and C. Bisdikian, "PARO: a power-aware routing optimization scheme for mobile ad hoc networks," draft-gomez-paro-manet-00.txt, work in progress, IETF (March 2001).
14. Y.-C. Hu and D.B. Johnson, "Caching strategies in on-demand routing protocols for wireless ad hoc networks," in: *Proc. ACM International Conference on Mobile Computing and Network (MOBICOM)* (2000).
15. A. Iwata, C.C. Chiang, G. Pei, M. Gerla and T. Chen, "Scalable routing Strategies for ad hoc wireless networks," in: *IEEE Journal of Selected Areas on Communications*, Vol. 17, No. 8 (Aug. 1999).
16. D.B. Johnson and D.A. Maltz, "Dynamic source routing in ad-hoc wireless networks," Mobile Computing, T. Imielinski and H. Korth, Eds., Kluwer (1996) pp. 153–81.
17. S.J. Lee, Routing and Multicasting Strategies in Wireless Mobile Ad Hoc Networks, Ph.D. Thesis, Computer Science Department, UCLA (2000).
18. S.J. Lee and M. Gerla, "AODV-BR: backup routing in ad hoc networks," in: *Proc. IEEE WCNC 2000*, Chicago, IL (Sept. 2000).
19. Jinyang Li, Charles Blake, Douglas S.J. De Couto and Hu Imm Lee and Robert Morris, "Capacity of ad hoc wireless networks," in: *Proceedings of ACM International Conference on Mobile Computing and Networking (MOBIHOC)* (2001).
20. V. Park and S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," *Proc. IEEE INFOCOM '97*, Vol. 3, pp. 1405–13.
21. C.E. Perkins and P. Bhagwat, "'Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in: Comp. Commun. Rev. (Oct. 1994) pp. 234–44.
22. C.E. Perkins and E.M. Royer, "Ad-hoc on-demand distance vector routing," in: *Proc. IEEE Workshop on Mobile Computing Systems and Applications* (Feb. 1999) pp. 90–100.
23. C.E. Perkins, E.M. Royer and S.R. Das, "Ad-hoc on demand distance vector (AODV) routing," Internet Draft, draft-ietf-manet-aodv-05.txt (March 2000) work in progress.
24. E.M. Royer and C.-K. Toh, "'A review of current routing protocols for ad hoc mobile wireless networks," in: *IEEE Personal Communications* (April 1999).
25. C.A. Santivanez, R. Ramanathan and I. Stavrakakis, "Making link-state routing scale for ad hoc networks," in: *Proc. ACM International Symposium on Mobile Ad Hoc Networking & Computing (MOBIHOC)* (2001).
26. C.A. Santivanez, B. McDonald, I. Stavrakakis and R. Ramanathan, "On the scalability of ad hoc routing protocols," in: *Proc. IEEE INFOCOM 2002*, New York, NY (June 2002).
27. S. Singh, M. Woo and C.S. Raghavendra, "Power-aware routing in mobile ad hoc networks," in: *ACM International Conference on Mobile Computing and Networking (MOBICOM)* (1999).
28. M. Spohn and J.J. Garcia-Luna-Aceves, "Neighborhood aware source routing," in: *Proc. ACM International Symposium on Mobile Ad Hoc Networking & Computing (MOBIHOC)* (2001).
29. I. Stojmenovic and X. Lin, "Power-aaware localized routing in wireless networks," in: *IEEE Transactions on Parallel and Distributed Systems*, Vol. 12, No. 10 (Oct. 2001).
30. W. Su, S.-J. Lee and M. Gerla, "Mobility prediction in wireless networks," in: *Proc. IEEE MILCOM 2000* CA (Oct. 2000).
31. C.-K. Toh, "A novel distributed routing protocol to support ad-hoc mobile computing," in: *IEEE 15th Annual International Phoenix Conf. Comp. and Commun.* (March 1996) pp. 480–86.
32. J.E. Wieselthier, G.D. Nguyen and A. Ephremides, "Energy-aware wireless networking with directional antennas: the case of

session-based broadcasting and multicasting," in: *IEEE Trans. on Mobile Computing*, Vol. 1, No. 3 (July–Sept. 2002) pp. 176–191.
33. Virtual InterNetwork Testbed (VINT), http://www.isi.edu/nsnam/vint
34. Mobile Networking Architectures (Monarch), http://www.monarch.cs.rice.edu
35. Z. Ye, S.V. Krishnamurthy and S.K. Tripathi, "A framework for reliable routing in mobile ad hoc networks," IEEE INFOCOM (2003).

Chao Gui is a Technical Research Staff at Kiyon Inc (www.kiyon.com). His research interests include wireless networking and mobile computing. His current efforts are on industrial implementation of wireless mesh networks and embedded systems. Dr. Gui has received Ph.D. in Computer Science from University of California at Davis in 2005.

Dr. Prasant Mohapatra is currently a Professor in the Department of Computer Science at the University of California, Davis. He has also held various positions at Iowa State University, Michigan State University, Intel Corporation, Panasonic Technologies, Institute of Infocomm Research, Singapore, and the National ICT, Australia. Dr. Mohapatra received his Ph.D. in Computer Engineering from the Pennsylvania State University in 1993. He was/is on the editorial boards of the IEEE Transactions on computers, ACM/Springer WINET, and Ad hoc Networks Journal. He has served on numerous technical program committees for international conferences, and served on several panels. He was the Program Vice-Chair of INFOCOM 2004, and the Program Co-Chair of the First IEEE International Conference on Sensor and Ad Hoc Communications and Networks, (SECON-2004). Dr. Mohapatra's research interests are in the areas of wireless networks, sensor networks, Internet protocols and QoS.