A General Framework for Efficient Geographic Routing in Wireless Networks

Seungjoon Lee*, Bobby Bhattacharjee†, Suman Banerjee‡ and Bo Han†

*AT&T Labs - Research, Florham Park, NJ 07932 USA

†Department of Computer Science, University of Maryland, College Park, MD 20742 USA

‡Department of Computer Science, University of Wisconsin-Madison, Madison, WI 53706 USA

Email: slee@research.att.com, {bobby, bohan}@cs.umd.edu, suman@cs.wisc.edu

Abstract

We propose a new link metric called *normalized advance (NADV)* for geographic routing in multihop wireless networks. NADV selects neighbors with the optimal trade-off between proximity and link cost. Coupled with the local next hop decision in geographic routing, NADV enables an adaptive and efficient cost-aware routing strategy. Depending on the objective or message priority, applications can use the NADV framework to minimize various types of link cost.

We present efficient methods for link cost estimation and perform detailed experiments in simulated environments. Our results show that NADV outperforms current schemes in many aspects: for example, in high noise environments with frequent packet losses, the use of NADV leads to 81% higher delivery ratio. When compared to centralized routing under certain settings, geographic routing using NADV finds paths whose cost is close to the optimum. We also conducted experiments in Emulab testbed and the results demonstrate that our proposed approach performs well in practice.

Index Terms

Wireless multihop networks; geographic routing; routing metric; link cost estimation.

I. Introduction

Geographic routing (or position-based routing) uses location information for packet delivery in multihop wireless networks [1], [2], [3], [4], [5]. Neighbors locally exchange location information obtained through GPS (Global Positioning System) or other location determination techniques [6]. Since nodes locally select next hop nodes based on this neighborhood information and the destination location, neither route

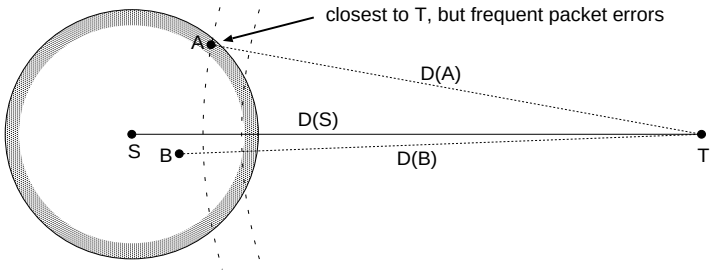


Fig. 1. An example scenario for geographic routing. While among S's neighbors, node A is closest to T, the link between S and A is experiencing a high packet error rate. Consequently, higher performance can be achieved if S forwards packets to B.

establishment nor per-destination state is required in geographic routing. As large-scale sensor networks become more feasible, properties such as stateless nature and low maintenance overhead make geographic routing increasingly more attractive [7]. Also, location-based services such as geocasting [8] can be best realized using geographic routing.

The most popular strategy for geographic routing is simply forwarding data packets to the neighbor geographically closest to the destination [1], [2], [3]. Although this *greedy* method is effective in many cases, packets may get routed to where no neighbor is closer to the destination than the current node. Many *recovery* schemes have been proposed to route around such *voids* for guaranteed packet delivery as long as a path exists [1], [2], [3], [9]. These techniques typically exploit planar subgraphs (i.e., Gabriel graph, Relative Neighborhood graph), and packets traverse faces on such graphs using the well-known *right-hand* rule. Most geographic routing protocols use one-hop information, but generalization to two-hop neighborhood is also possible [10].

In this paper we propose the use of a new link metric called *normalized advance (NADV)* in geographic routing. Instead of the neighbor closest to the destination, NADV lets us select the neighbor with the best trade-off between link cost and proximity. In Figure 1, for example, although A is closest to destination T among S's neighbors, the link between S and A is experiencing high packet error probability. B is slightly farther from T than A, but provides a higher quality link from S. In this scenario, forwarding packets to B is better, and NADV chooses B over A. While this idea has been proposed in other specific contexts [11], [12], we consider a generalized framework and show that a path chosen by NADV approaches the *optimal* minimum cost path for a broad range of costs in networks with sufficiently high node density. Our proposed metric is best understood in the context of greedy mode in geographic routing, but it can also be used

with schemes that route around voids [3], [9] (discussed later in Section VI).

Due to the local rule for next hop decision, the use of NADV in geographic routing provides a unique opportunity for adaptive routing—a feature not offered by most existing on-demand routing protocols. For example, suppose that a source uses an on-demand routing protocol (e.g., AODV [13]) to find a minimum cost path. In dynamic ad hoc networks, it is possible that the link costs change while the path is still in use (e.g., due to mobility or environment changes). If the source detects the change and wants to find a better path, typical on-demand routing protocols require the flooding of a new route discovery message. This solution may incur high control overhead, and it is also difficult to know when the source should initiate the flooding. In contrast, as long as link cost estimation schemes can track link costs change, NADV immediately reflects the change, which in turn would result in the selection of the best next hop in geographic routing.

We present NADV in the context of a general framework for efficient geographic routing. Although a few recent geographic routing schemes consider link costs in the next hop selection [14], [11], [12], they are limited to one specific objective. For example, the *SP-Power* scheme in [14] focuses exclusively on the minimization of transmission power consumption. In contrast, the NADV framework can accommodate a variety of different cost types. Depending on system objectives or message priority, applications can use the NADV framework to take different routing strategies. For example, an urgent message can be routed along the path that minimizes the end-to-end latency, and a low-priority message may take a path that minimizes power consumption to increase the overall network lifetime.

For the effective use of NADV, we present techniques for efficient and adaptive link cost estimation. Some of previous work uses additional probe messages for link cost estimation in the bootstrapping phase [15], [16]. However, such control messages consume already scarce network resources. Also, network environments may change over time (e.g., due to mobility), and old link estimates may become obsolete. We propose to exploit MAC-level information, so that link cost estimation is adaptive to changing environments, yet incurs minimal control overhead. We also provide multiple techniques thus enabling nodes to choose the best scheme for the current network and system setting. In a resource-rich network, for example, nodes can use a method that uses probe messages. In the case of a dense large-scale network with limited resources, such probe messages may prove to be costly, and nodes can use an alternate scheme that uses no extra control messages.

We have performed extensive experiments in simulated environments to evaluate the effectiveness of NADV and link cost estimation techniques. When compared to the current geographic routing scheme in challenging environments with frequent packet losses, NADV leads to 81% higher packet delivery ratio on average (from 16% to 97%). The number of MAC-level data transmissions and end-to-end delay also decrease significantly (by up to 60%). The simulation results also show that when link costs change, the use of NADV in geographic routing enables adaptive path migration, where the quality of found paths is close to the optimum found by the centralized algorithm. We have also conducted experiments in Emulab testbed and the results validate that our proposed strategy performs well in practice.

The rest of this paper is organized as follows. In Section II we define the new link metric. Link cost types and estimation techniques are described in Section III. We present the simulation results in Section IV. Section V summarizes experiment results on real testbeds. Section VI presents related work, and Section VII concludes.

II. New Link Metric for Geographic Routing

In this section, we introduce a new link metric for geographic routing and discuss its optimality in an ideal setting. Here, we assume link cost is positive and known a priori. We discuss link cost estimation in Section III.

A. Background

In this paper we differentiate link *cost* and link *metric*. An example of link *cost* is the power consumption required for a packet transmission over the link. We define link *metric* as "degree of preference" in path selection. For example, even though two neighbors require the same power consumption, in geographic routing we prefer the neighbor closer to the destination. The goal of this section is to propose a new link metric for geographic routing that can be generalized to various cost types (e.g., power consumption, link delay).

In many geographic routing protocols, the current node S greedily selects the neighbor that is closest to destination T whenever possible [1], [2], [3]. The implicit goal of this strategy is to minimize the hop count between source and destination. Let us consider the amount of decrease in distance by a neighbor n, which we call the *advance (ADV)* of n [17]:

$$\mathit{ADV}(n) = D(S) - D(n), \tag{1}$$

where $D(x)$ denotes the distance from node x to T. Then, the above strategy tries to maximize the ADV of next hop, and ADV is the link metric in this case. However, this link metric ADV does not take link cost into account, while different wireless links can have different link costs. For example, Lundgren et al. [18] identify *gray zones*, where due to high error probability, nodes cannot exchange long data packets in most cases. Therefore, the simple policy using ADV may use poor quality links and lead to unnecessarily high communication cost [15].

Clearly, when choosing next hops we want to avoid neighbors with very low quality links. At the same time, we want to gain as large advance as possible for fast and efficient packet delivery. The goal of our work is to balance the trade-off, so that we can select a neighbor with both large advance and good link quality. We can achieve this goal by using the new metric proposed next.

B. Normalized Advance

We now introduce a new metric called *normalized advance (NADV)*. Suppose we can identify the link cost *Cost*(n) of the link to neighbor n. Then the normalized advance of neighbor n is simply:

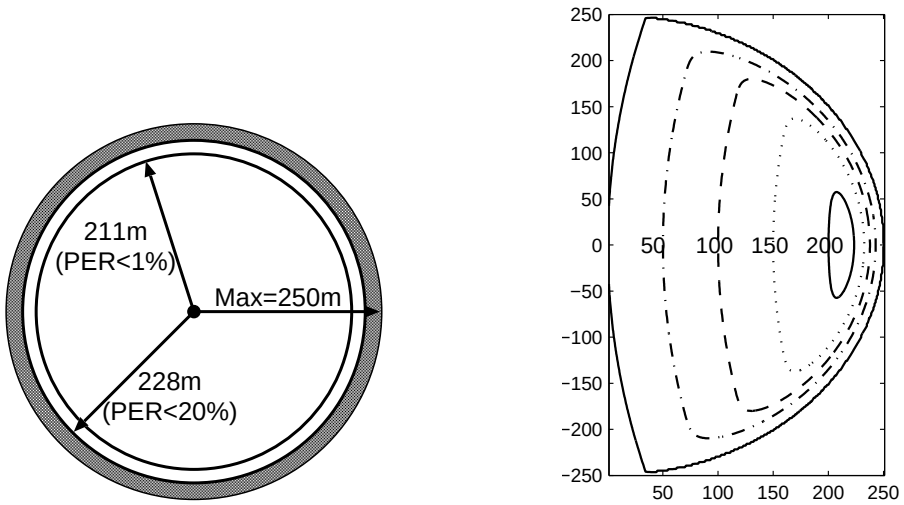
$$\textit{NADV}(n) = \frac{\textit{ADV}(n)}{\textit{Cost}(n)}. \tag{2}$$

Intuitively, NADV denotes the amount of advance achieved per unit cost. For example, suppose we know that only $P^{succ}(n)$ fraction of data transmissions to neighbor n are successful. If we use $1/P^{succ}(n)$ as link cost, $\textit{NADV}(n) = \textit{ADV}(n) \times P^{succ}(n)$, which means the expected advance per transmission.

We propose to use NADV as link metric in geographic routing, such that a node forwards packets to the neighbor with largest NADV. Besides obvious simplicity, NADV has the following desirable properties:

- As shown in Section II-C, the path found by using NADV approaches the optimal path under certain conditions. The experiment results in Section IV show that the use of NADV significantly improves path quality in realistic environments as well.
- It is general and accommodates various types of cost metrics, so that applications can utilize the NADV framework for different objectives. We further describe this feature in Section III.
- Loop freedom is guaranteed as long as we select a node with positive NADV [17].

Using NADV, we can select neighbors that balance the advance against the link cost. Depending on the link cost values, NADV can select a neighbor with strictly less advance (e.g., node B over A in Figure 1). We further illustrate this feature in Figure 2. Figure 2-(a) shows the degree of packet errors to simulate



(a) Illustration of gray zone (b) NADV contour map

Fig. 2. Illustration of gray zone and corresponding contour map of NADV. (a) Two inner circles represent the border lines for 1% and 20% packet error rates (PERs) for a 1024-byte frame, respectively. (b) The corresponding contour map of NADV when the packet error probability determines link cost. The current node is at (0,0), and the destination is 900 meters away on the X-axis. Values within the plot denote the NADVs of corresponding lines.

a gray zone.[1] In Figure 2-(b), we present the corresponding contour map of NADV when link cost is a function of packet error probability. We can observe that compared to their ADV values, points within the gray zone have relatively low NADV values. As a result, by using NADV, we can easily avoid neighbors in the gray zone. We next provide the theoretical rationale behind using NADV in geographic routing.

C. Optimality of NADV in an Idealized Environment

We now show that in an idealized environment, paths found by using NADV are optimal. The goal of routing in this discussion is to minimize the sum of link costs along the found path. We make two assumptions: (1) we can find a node at an arbitrary point, and (2) link cost is an *unknown* increasing convex function of distance (e.g., transmission power consumption [14], [19]). Let *DIST* be the distance between the source and the destination, which we assume is relatively large. Since the cost function is increasing, and we can find a node at an arbitrary point, an optimal path will use only nodes along the straight line between the source and the destination. Also, since link cost is a convex function of distance, the sum of link costs is minimized when all links have the same distance. As a result, the optimal policy is to choose nodes on an equidistant basis along the line that connects the source and the destination.

[1]The bit error function used here increases rapidly after a certain distance. A detailed description on the error model is in Section IV-B.

Now, it remains to find the optimal interval. Suppose ADV_X is an interval, and $Cost_X$ is the corresponding link cost. Then we want to minimize:

$$\begin{aligned} \textit{Total Cost} &= \textit{(Link Cost)} \times \textit{(Hop Count)} \\ &= Cost_X \times \left\lceil \frac{DIST}{ADV_X} \right\rceil \\ &\approx DIST \times \frac{Cost_X}{ADV_X}. \end{aligned} \tag{3}$$

The last line comes from the assumption of large *DIST*, which makes the rounding error negligible. From Eq. 3 we can find the minimum cost path by iteratively selecting nodes with minimum $\frac{Cost}{ADV}$, or equivalently *maximum NADV* $= \frac{ADV}{Cost}$.

In practical wireless networks, the above assumptions are unlikely to be true. In low-density networks, nodes may not be able to use the greedy forwarding rule, and the recovery procedure will likely result in performance degradation [3]. Also, although many existing schemes are based on the simplified model, and there usually exists strong correlation [18], [20], the link cost is not a strict function of distance in practice. In Section IV and Section V, we use both simulations and real experiments to show that NADV significantly improves performance in realistic environments as well.

Although the concept of NADV is simple, the implementation for practical use involves a number of challenges. Link cost estimation is one of the most critical elements, and in the next section, we describe a set of methods to infer various types of link costs and show how the NADV framework utilizes them.

III. Link Cost Types and Estimation

For effective link cost inference, we propose a new sublayer named *Wireless Integration Sublayer Extension* (WISE) located on top of the MAC layer (See Figure 3). The WISE closely coordinates with the MAC layer for efficient link cost estimation. It also provides simple primitives for upper layer protocols to retrieve the inferred performance values. When additional control messages are available [15], [16], [21], WISE extracts relevant link cost information from them. Otherwise, WISE exploits MAC-specific information to infer the communication performance. For example, the WISE retrieves the current transmission power from the MAC layer and calculates overall power consumption needed for a packet transmission. We note that this sublayer approach and the estimation schemes below can be used in any cost-aware wireless routing.

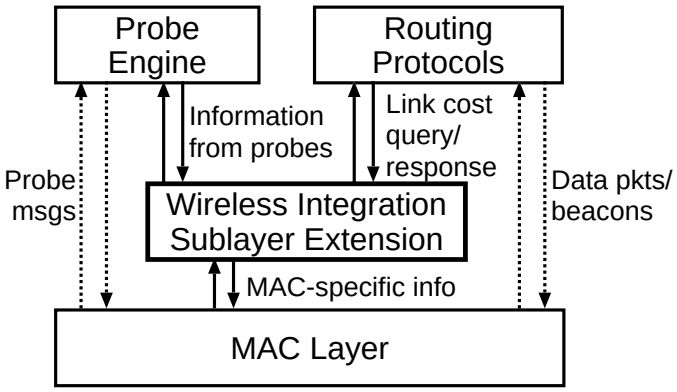


Fig. 3. Wireless Integration Sublayer Extension (WISE) abstracts the performance of wireless link into numeric values and provides simple primitives for upper layer protocols to retrieve specific performance values. For link cost estimation, WISE exploits MAC layer specific information. If available, WISE can leverage probe messages as well.

To illustrate how to use the NADV framework to meet different performance objectives, we discuss three specific types of link cost: packet error rate, link delay, and energy consumption. We also describe how WISE is used to estimate link costs in diverse operating environments. This paper focuses on the independent use of each link cost, and the issue of interdependence among multiple cost criteria is discussed in Section VII.

A. Packet Error Rate (PER) Estimation

Most recent attention has been on how to find a high performance path considering wireless link errors [22], [15]. In this scenario, we use the following as link cost: $C_{error} = 1/(1 - PER)$. It denotes the expected transmission count (ETX) proposed in [15].[2] We use the following link metric, which is the expected advance per transmission:

$$NADV_{error} = \frac{ADV}{C_{error}} = ADV(1 - PER). \tag{4}$$

An equivalent link metric is developed in [11], [12], and we discuss them in Section VI.

We next describe simple wireless bit error models and present four PER estimation methods for $NADV_{error}$, each of which requires a different degree of control overhead and message format modification.

[2]The estimation techniques described here can easily incorporate ACK frame loss probability as in [15], but here we have simplified the description for brevity.

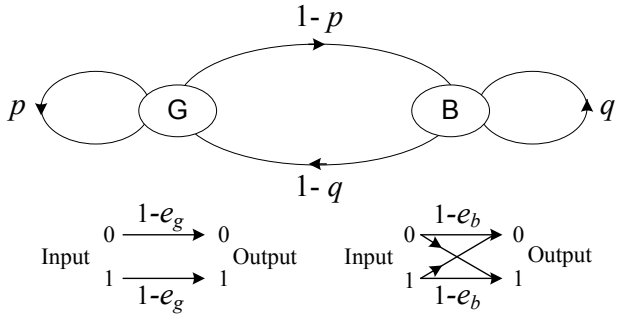


Fig. 4. Gilbert/Elliot model. G denotes *good* state, and B denotes *bad* state.

1) Existing Bit Error Models: Wireless links are typically more prone to packet errors than wired ones, and many theoretical models for wireless packet error have been proposed [23], [15], [24]. In this paper, we describe two simple packet error models that can be easily used in real wireless networks.

The *independent bit error model* is among the simplest for wireless packet errors. In this model, each bit is corrupted independent of other bits in the packet. Specifically, if the bit error rate is p_b, then the error probability for an l-bit packets is:

$$\textit{PER}(l) = 1 - (1 - p_b)^l. \tag{5}$$

A number of previous results show that bit errors are often correlated and occur in a bursty fashion [23], [24], [25], [26]. Some previous works use finite-state Markov models to model such correlated bit errors. Although such a model can have an arbitrary number of states, for simplicity, we use the *two-state Markov model* proposed by Gilbert and Elliot [25], [26] in this paper. In the Gilbert/Elliot (GE) model, a wireless channel is in one of the following two states: *good* and *bad* (Figure 4). If the channel is in good state, then a bit transmission error occurs with the probability of e_g. On the other hand, if the channel is in bad state, the probability of bit transmission error is e_b. Prior to the transmission of each new bit, the channel may change states or remain in the current state. Figure 4 shows the GE model representation with state-transition probabilities. In this paper, we use $e_g = 0$ and $e_b = 1$ for simplicity.

For this model, we can calculate the steady-state probability of being in *good* state (P_G) and *bad* state (P_B) as follows:

$$P_G = \frac{1-q}{2-p-q}, \qquad P_B = \frac{1-p}{2-p-q}.$$

Note that there are two cases where no bit error occurs in a packet. First, the channel is initially in good state and remains there for all bit transmissions, and the probability is $P_G\ p^l$ for l-bit packet. In the

other case, the channel is initially in bad state, but the channel changes into good state for the first bit transmission and remains in good state. This probability is $P_B\ (1-q)p^{l-1}$. A packet error occurs if none of them happens; therefore, the error probability of an l-bit packet is:

$$\textit{PER}(l) = 1 - (P_G\ p^l + P_B\ (1-q)p^{l-1}). \tag{6}$$

2) Using Probe Messages for PER Estimation: If a node is already using probe messages [15], [16], [27], the WISE can extract the link error probability from them. Since most geographic routing schemes use periodic message exchange between neighbors, we can also use the reception ratio to infer the link error probability. However, such control messages are usually shorter than data packets, and as a result, a node may experience higher PERs for actual packets [15]. To obtain more accurate link cost estimation, we need to adjust PER depending on the data packet length. Although more advanced bit error models may be employed for more accurate PER estimation [24], in this paper we focus on the two bit error models described above: the independent and GE models.

Assuming independent bit errors, we can adjust PER as follows: If we use m-bit probe messages, from the observed $\textit{PER}(m)$, we employ Eq. 5 to infer bit error rate as follows: $p_b = 1 - (1 - \textit{PER}(m))^{1/m}$. Then, for an L-bit data frame we can use:

$$\textit{PER}(L) = 1 - (1 - \textit{PER}(m))^{L/m}. \tag{7}$$

In case we want to use the two-state Markov model shown in Figure 4, we need at least two distinct PER values observed for different probe message types. In addition to $\textit{PER}(m)$, consider $\textit{PER}(n)$ for n-bit probe messages. Using Eq. 6, we can get one of the state transition probabilities in Figure 4 as follows:

$$p = \left(\frac{1 - \textit{PER}(n)}{1 - \textit{PER}(m)}\right)^{\frac{1}{n-m}}.$$

Then, we can estimate the PER of L-bit data messages using the following formula:

$$\textit{PER}(L) = 1 - (1 - \textit{PER}(m)) \left(\frac{1 - \textit{PER}(n)}{1 - \textit{PER}(m)}\right)^{\frac{L-m}{n-m}} \tag{8}$$

In Section V, we present some measurement results to demonstrate the effectiveness of these estimation techniques.

3) Using Signal-to-Noise Ratio for PER Estimation: We can also estimate p_b using Signal-to-Noise Ratio (SNR) and theoretical error models for different modulation schemes [23]. Assuming an AWGN (Additive White Gaussian Noise) channel, in the case of BPSK (Binary Phase Shift Keying), the bit error rate is given by:

$$p_b = 0.5 \times \textit{erfc}(\sqrt{\frac{P_r \times W}{N \times f}}), \tag{9}$$

where P_r is the received power, W the channel bandwidth, N the noise power, f the transmission bit rate, and *erfc* the complementary error function. Most wireless cards typically measure SNR = $10 \log \frac{P_r}{N}$ (dB) for each received packet, and using such SNR values and Eq. 9, a node can calculate p_b for its neighbors. Using an appropriate bit error models [24], we can infer the packet error rate from p_b. Then, due to possibly asymmetric link quality, it should inform its neighbors of respective SNR values. This can be done either via additional control messages or by modifying the beacon message format to include the information.

Eq. 9 is useful primarily in free-space environments, but not applicable for indoor environments, where signal path characteristics are more complex. The measurement results using a rooftop mesh network show that it is hard to predict link quality using SNR [28]. However, in different measurement studies using a sensor network and long-distance 802.11b links, Zuniga et al. [20] and Chebrolu et al. [29] report that empirical results closely match their analytical models.

4) Neighborhood Monitoring for PER Estimation: A node can also use passive monitoring to infer link PERs as in [30], [27]. For example, in IEEE 802.11 networks, node A can monitor frames sent by neighbors. In that case, using the MAC sequence number A can count how many frames from neighbor B it has missed, and infer the PER of link from B to A. Again, since the quality of two directional links may differ, A needs to inform B of the PER estimation as in the previous scheme.

5) Self Monitoring for PER Estimation: The previous methods require either additional control messages or the modification of beacon message format. When these are not possible, we suggest the following technique. Whenever a node transmits a data frame to neighbor n, the MAC-layer informs the WISE whether the transmission was successful or not. Let us define an indicator variable F; $F = 1$ when a frame exchange failed, and $F = 0$ otherwise. Then, WISE infers the PER of wireless link to neighbor n as follows:

$$\textit{PER}_n \leftarrow (1 - \alpha)\textit{PER}_n + \alpha F, \tag{10}$$

where α denotes the weight parameter. In the simulation study in Section IV, we use $\alpha = 0.1$, and the default PER value is set to 0. Note that $F = 1$ even when an ACK frame failure occurs in IEEE 802.11 networks [15].

To track the link quality change even when no packets are forwarded to n, we use an *aging* scheme and periodically reduce PERs of unused links. When this reduction makes the estimated PER become lower than the actual one, packets may be forwarded to n, but the estimated PER will increase after transmission failures. The magnitude and frequency of reduction should balance such overhead and prompt adjustment. In the simulation, we multiply PERs of unused links by 0.9 every 30 seconds.

B. Delay

If link delay C_{delay} is used as link cost to reduce the path end-to-end delay, we can use $NADV_{delay} = \frac{ADV}{C_{delay}}$. We can think of two types of link delay. First, due to the broadcast nature of wireless medium, it is desirable to minimize the *medium time*, the time spent in sending a packet over the link [31]. When the underlying physical medium supports multi-rate transmissions (e.g., the IEEE 802.11 standard), it is a function of the current transmission rate. The WISE can easily retrieve the current value of transmission rate from the MAC layer and calculate the necessary medium time to the neighbor.

The other is *total delay*, which denotes the time from the packet insertion into the interface queue until the notification of successful transmission. It includes the queueing delay, backoff timeout, contention period, and retransmissions due to errors or collisions. Using this value as link cost can potentially enable packets to detour congested areas. The design of a routing scheme with such detouring capability is a part of our future work, and we use the medium time as C_{delay} in this work [32].

C. Power Consumption

Many wireless systems have a control mechanism for transmission power adjustment to save battery and reduce interference [21], [23]. We assume that using such a mechanism, nodes know the appropriate transmission power level (p_{tx}) to each neighbor. Then, the WISE can retrieve the p_{tx} value and calculate the actual system power consumption C_{power} considering additional components of power consumption [33]. If C_{power} is used as link cost, a geographic routing protocol can use $NADV_{power} = \frac{ADV}{C_{power}}$ to find a path that minimizes power consumption to deliver packets to a destination.

So far, we have listed interesting cost types and shown how the NADV framework can incorporate them. The NADV framework still can include other types of link cost as well (for example, *reluctance* metric in [34]). However, in this paper we limit our attention to the cost types discussed above and report simulation results in the following sections.

IV. Simulation Results

In this section we first describe the simulation setup and error model. Then we present the results of simulation experiments.

A. Simulation Setup

We use *ns-2* simulations to evaluate the system performance when we employ the proposed NADV metric and link cost estimation schemes. In this section, we describe various aspects of simulation in detail. We present the simulation results in Section IV.

We place nodes uniformly at random on a 1000m by 1000m square. Unless otherwise stated, 100 static nodes are used in the simulation.[3] We usually use only one source-destination pair to capture the individual performance effects accurately. In this scenario, denoting the lower left corner of the square as (0, 0), the static source is located at (50, 500). The destination is placed at (50+D, 500), where D is the distance between the source and the destination. We usually use D=900. The source generates a CBR (Constant Bit Rate) flow, which sends a 1024-byte UDP packet every two seconds from 10 seconds to 1000 seconds of simulation time. The maximum transmission range R is 250 meters.

For geographic routing, we use the simulation code for GPSR.[4] We have slightly modified the next hop selection algorithm to include NADV. The simulation code for GPSR provides an option about whether to exploit transmission failure notification from the MAC layer [2]. If a node exploits the option, then upon receiving a notification, it selects the next best neighbor for retry until the forwarding is successful. This option leads to higher delivery ratio with higher resource consumption. When not using the notification, a node does not attempt to retransmit to other neighbors. We explore both cases in the simulation. The beaconing period in GPSR is set to 1.5 seconds. We use the IEEE 802.11b standard for the underlying

[3]We also experimented using sparser networks with 50 nodes. However, in scenarios with high packet error rates, networks frequently became disconnected (e.g., due to repeated beacon message losses).

[4]Available at http://www-2.cs.cmu.edu/~bkarp/gpsr/gpsr.html

	Noise power ($\times$1.0e-12 W)				
	0.8	1.0	1.2	1.4	1.6
(dBm)	(-91.0)	(-90.0)	(-89.2)	(-88.5)	(-88.0)
BER at 220m	6.0e-8	1.1e-6	7.8e-6	3.2e-5	9.1e-5
BER at 240m	4.4e-6	3.5e-5	1.4e-4	3.9e-4	8.3e-4

TABLE I

BIT ERROR RATE VALUES WITH DIFFERENT LEVELS OF NOISE.

MAC layer protocol [35]. We assume the location of the destination is known to the source. For the simulation results presented in this paper, the transmission rate is fixed to 1Mbps. We also performed experiments using different transmission rates in the MAC layer, and the results are available at [32].

B. Error Model

To simulate a lossy channel, we use two different error models. First, assuming the use of BPSK modulation in the physical layer, we simulate packet errors using Eq. 9 as bit error model. (We assume independent bit errors for simplicity.) In the default *ns-2* propagation model, the signal strength is reduced proportionally to d^2 if the distance d is smaller than a certain threshold. Otherwise, the path loss is proportional to d^4. In this experiment scenario the transmit signal power is fixed at 20 mW (or 13dBm) supported in Cisco Aironet 350 interface cards [36]. Then the received signal strength for a node 250 meters away is -85dBm. The noise channel bandwidth in Eq. 9 is set to 2MHz. In this model, we use ambient noise environments, where the noise value is identical everywhere. Therefore the quality of a link depends only on the distance between two nodes, and C_{error} is a convex function of distance. In Table I we tabulate the used noise values and corresponding bit error rates (BERs).[5]

To examine the performance of NADV in the presence of randomness in packet errors [28], we also perform simulations using a random packet error model. In this model, for each wireless link, we assign a packet error rate, which is distributed uniformly at random between 0 and a maximum value (*max-PER*). We vary the maximum packet error probability for different degrees of packet losses. In practice, shorter packets such as periodic beacons experience lower error probability [15], and we adjust the error probability for these packets according to Eq. 7. Clearly, link cost is not a function of distance in this model.

[5]Noise values from more than 20000 measurements in our building range from -91dBm to -73dBm, with the median at -89dBm. The noise value used in Figure 2 is -89.2dBm.

In some of our simulations, we compare NADV against another scheme called *blacklisting* [37], [11]. This scheme uses a fixed threshold, and when selecting a next hop, a node excludes neighbors with low-quality link based on the threshold. For example, if we use a threshold value of 0.5, then a node excludes neighbors that are closer to the destination and belong to the lower half in the link quality. Among the remaining neighbors, the blacklisting scheme selects the neighbor with largest ADV.

In the rest of this section, we present the results of simulation experiments. We begin with the effect of wireless link errors. We first assume the perfect knowledge of link error rates when we investigate the performance. Then we compare the performance when we combine our proposed estimation techniques with NADV. We then consider the cases when delay and power consumption are used as link costs in turn. Finally we compare geographic routing using NADV against idealized routing.

C. Experiments with Perfect Estimation of Link Errors

We first present the results when nodes experience packet losses due to wireless link errors. In this section, we assume that there exists a perfect estimation scheme that provides accurate link cost values, and compare the performance when NADV and other geographic routing schemes operate based on the knowledge. We later present results when we combine NADV with the proposed PER estimation techniques described in Section III-A.

In the first set of experiments, we use the random packet error model described in Section IV-B, where packet error rates are distributed uniformly at random between 0 and *max-PER*. Although in this model the frequency of links at a given error rate is similar to the previous measurement results in [28], this model does not consider the correlation between the distance and link error. As a result, on average, packets sent to distant neighbors have the same error probability as those sent to close neighbors. In fact, this setting is in favor of ADV, because in practice, transmissions to neighbors with large ADV are likely to suffer from frequent errors [18], [20]. We use an average of ten runs, each with different placement of stationary nodes and fix the data transmission rate at 1 Mbps in this set of experiments.

In Figure 5, we report the number of MAC-level data transmissions (including retransmissions) per delivered packet for each scheme when we vary the value of *max-PER*. In this set of experiments, GPSR employs MAC-level failure notification, and all results are based on 100% packet delivery. We can observe that as the packet error rate increases, the data transmission overhead of ADV increases abruptly (up to 71% higher than that of NADV). This is because ADV often selects neighbors with low-quality link,

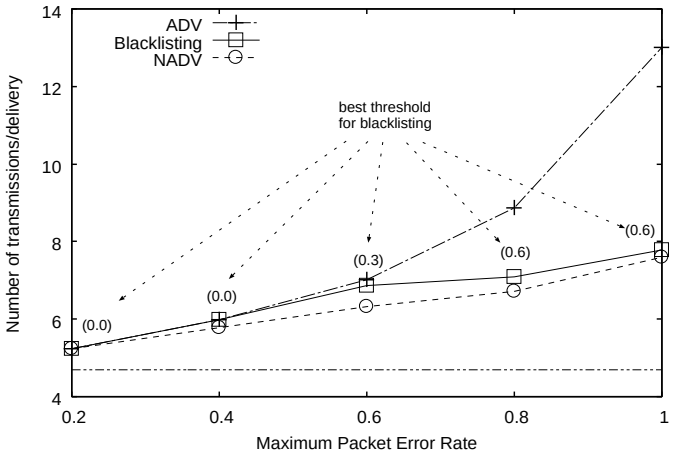


Fig. 5. The number of MAC-level data transmissions per delivered packet with different degrees of packet errors. C_{error} is used as link cost. Unless the error rate is low, next hops chosen by ADV can cause multiple retransmissions, and ADV significantly increases the number of data transmissions. As marked in the bottom, when *max-PER*=0.2, the average path length using ADV is 4.7 hops.

which causes repeated retransmissions. In contrast, NADV intelligently avoids nodes with high PER, and although the data overhead of NADV increases as *max-PER* increases, the number of data transmissions is much smaller than that of ADV. Each transmission requires network bandwidth as well as node resources (e.g. battery power), and NADV uses system resources more efficiently.

In Figure 5, we also compare the performance of NADV against the *blacklisting* scheme described in Section IV-B. Blacklisting uses a fixed threshold, and to find the best threshold, we consider nine different blacklisting threshold values between 0.1 and 0.9 with an increment of 0.1. (The use of threshold value 0.0 in blacklisting corresponds to ADV.) In Figure 5, we plot the best result for blacklisting in each setting and mark the corresponding threshold value in parenthesis. We can observe that depending on the network environment, different threshold values lead to best results for blacklisting and that a fixed threshold value in blacklisting does not work well. When packet errors are frequent, it is better to use a high threshold value in blacklisting and exclude many neighbors with low-quality links. However, when there are few low-quality links, the use of a high threshold value may exclude useful neighbors and lead to longer paths. In contrast, NADV adapts to the changing network environment and is able to achieve low data transmission overhead in all cases.

Repeated retransmissions also affect the packet delay. Although not displayed here, the end-to-end latencies of ADV also show an increasing trend similar to Figure 5. Specifically, as *max-PER* changes from 0.2 to 1.0, the average packet latency of ADV increases from 54.9ms to 151.6ms. The performance

Name	Description
NADV-Beacon	Using periodic beacon messages (Eq. 7)
NADV-SNR	Using SNR (Eq. 9)
NADV-Self	Using own data packets (Eq. 10)
NADV-Perfect	Assuming the perfect knowledge of link cost

TABLE II

DIFFERENT SCENARIOS WHEN NADV AND THE PER ESTIMATION TECHNIQUES ARE COMBINED.

	max-PER		
	0.6	0.8	1.0
NADV-Self	7.0 (0.6)	7.8 (0.8)	8.9 (1.1)
NADV-Beacon	6.6 (0.7)	7.2 (0.6)	7.8 (1.2)
NADV-Perfect	6.3 (0.5)	6.7 (0.7)	7.6 (1.2)

TABLE III

THE NUMBER OF DATA TRANSMISSIONS PER DELIVERED PACKET WHEN NADV AND PROPOSED PER ESTIMATIONS ARE USED. VALUES IN PARENTHESES ARE THE STANDARD DEVIATIONS. WHEN THERE ARE NO LINK ERRORS, THE AVERAGE PATH LENGTH IS 4.7 HOPS.

degradation by NADV is less severe (increase from 54.9ms to 81.8ms). Instead of NADV, we also experimented using different combination of ADV and link cost, and NADV outperformed them as well. A more conservative link metric (e.g., $ADV/Cost^2$) results in longer paths, while a different metric such as $ADV/\sqrt{Cost}$ often underestimates high-cost links and causes more retransmissions due to packet errors.

In the previous experiments, we assumed the perfect knowledge of link cost. We next investigate the performance of NADV used with the proposed PER estimation techniques.

D. Experiments using Proposed PER Estimation Techniques

In Table II, we tabulate three schemes when we use NADV and the proposed PER estimation techniques together, in addition to case with the perfect estimation (NADV-Perfect). Note that none of the three estimation schemes use extra control messages. However, in the case of NADV-SNR and NADV-Beacon, we modify the periodic beacon format to include reverse link information, and the message length slightly increases. Storage overhead for the link cost estimation is also negligible since each node in GPSR already maintains neighbor information.

In the first set of experiments, we use the random packet error model used in the previous section and

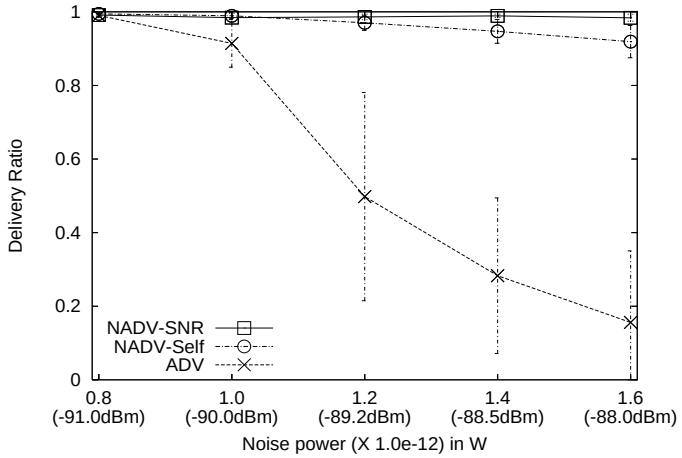


Fig. 6. Packet delivery ratios when noise levels change. Values in parentheses on the X-axis are noise powers in dBm. Each point is an average of ten results, and error bars denote the corresponding standard deviations. The use of ADV may lead to abrupt performance degradation, but with NADV, the delivery ratios are maintained high.

compare the actual performance of NADV against the case with the perfect knowledge of link cost. In this model, there is no correlation between SNR and packet error rate, and we experiment with NADV-Beacon and NADV-Self only. In Table III, we show the data transmission overhead of NADV schemes for high-error scenarios. In this table, the overhead of NADV-Beacon and NADV-Self is reasonably close to that of NADV-Perfect, and we can infer that our proposed schemes provide good link cost estimation. Although NADV-Self has the most flexibility in deployment (e.g., no modification of protocol message format), its performance is slightly worse than NADV-Beacon.

In the previous experiments, we used the random packet error model. In the next experiments, we use the other error model described in Section IV-B, where a bit error occurs according to Eq. 9 (See Table I). To identify the performance of proposed PER estimation schemes, we consider five simulation scenarios. First, we compare packet delivery ratios when we do not employ MAC-level failure notification. Second, we change the ambient noise power over time and observe how each estimation technique adapts to varying environments. Third, we study the number of MAC-level data transmissions (including retransmissions) per delivered packet. Fourth, we increase the number of data flows to vary the network contention level. Fifth, we consider the scenario with node mobility.

Without MAC-level Failure Notification: In these experiments, we compare packet delivery ratios when MAC-level failure notification is not employed. In Figure 6, we plot the delivery ratios achieved by ADV and NADV, respectively, when we vary the noise power values. We use an average of ten runs,

each with different node placement, and the error bars in the figure are the standard deviations. When wireless link errors are rare (noise=-91dBm), the use of ADV in geographic routing leads to relatively good performance. However, as the channel condition degrades, the performance gap between NADV and ADV grows larger. For example, when ADV is used with noise power=-89.2dBm, less than 50% of packets can reach the destination on average. However, when NADV is used, the delivery ratio is maintained high ($>$ 97% on average for both NADV schemes). Since NADV-SNR explicitly utilizes the link characteristic value, it leads to paths of higher-quality links, and the delivery ratios are slightly higher than those of NADV-Self.

The performance gap between NADV and ADV is explained as follows. The neighbor with maximum ADV is relatively far from the current node, and the corresponding link may experience a higher PER. However, when using ADV, the current node blindly uses the neighbor with maximum ADV and suffers from the poor quality link. In contrast, when using NADV, the link estimation schemes find out that the link quality is poor and accordingly reduce the NADV value for the neighbor. Then, the node chooses a different neighbor with a larger NADV value, and the overall network performance improves. We can also observe the high *variance* of delivery ratios when ADV is used. For example, when noise power=-89.2dBm, the delivery ratios of ten runs range between 21.0% and 97.2% for ADV. The reason is that, in some fortunate cases, all forwarding nodes selected by ADV can possibly be outside of gray zones, and the forwarding does not experience high PERs. In contrast, NADV selects the neighbor with the best trade-off and consequently leads to stable performance ($>$ 92% in the worse case).

Changing Noise Power: In these experiments, we investigate the adaptiveness of PER estimation schemes, and we start with a high noise value, change to a low noise value after 300 seconds, and change again to a medium noise value after 700 seconds. In Figure 7 we plot the average path lengths for ADV and NADV when we vary the ambient noise power value. We do not employ MAC-level failure notification in GPSR, and the values in the parentheses show the average delivery ratios for different scenarios. Since the performance of NADV-Beacon is similar to that of NADV-SNR, we do not show the result of NADV-Beacon for clarity.

In Figure 7, the length of the path chosen by ADV is always shortest, but the packet delivery performance is always worse than that of NADV. For example, in the high-error scenarios (noise value=-88dBm), the difference in packet delivery ratio is more than 81% (16.3% vs. 97.7%). In this scenario, although ADV

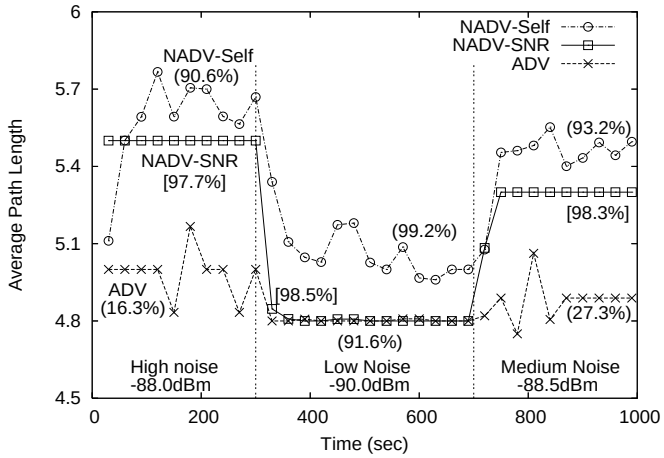


Fig. 7. The average path lengths of NADV and ADV. The noise value changes from high (-88.0dBm) to low (-90.0dBm), and finally to medium (-88.5dBm). Numbers next to the lines are corresponding delivery ratios in each phase. PER estimation schemes enable NADV to choose appropriate neighbors and maintain high delivery ratios.

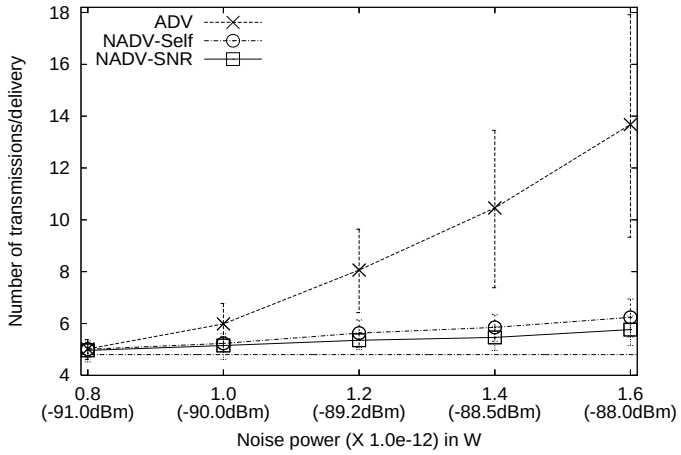


Fig. 8. The number of MAC-level data transmissions per delivered packet with different degrees of packet errors. C_{error} is used as link cost. Due to multiple retransmissions, ADV causes a higher number of total transmissions than NADV. As marked in the bottom, the average path length found by ADV when noise=-91dBm is 4.8 hops.

does not adapt to the environment, beacons from far neighbors are frequently lost, and the path length increases. In the case of NADV, the PER estimation schemes dynamically assign appropriate link costs. As a result, NADV uses different neighbors according to the current environment, and the path length change is more noticeable. NADV-SNR explicitly utilizes the link characteristic value, and in this simulation model, NADV-SNR exhibits more accurate estimation and faster convergence. NADV-Self occasionally employs slightly longer paths than NADV-SNR, but it is also able to adapt to environment changes.

MAC-level Data Transmissions: In Figure 8, we report the number of MAC-level data transmissions (including retransmissions) per delivered packet for both ADV and NADV. GPSR employs MAC-level failure notification in this set of experiments, and all results are based on 100% packet delivery. We can see that NADV intelligently avoids nodes with high PER, and the number of data transmissions is accordingly much smaller. Each transmission requires network bandwidth as well as node resources (e.g. battery power), and NADV uses system resources more efficiently. In contrast, the use of ADV leads to the waste of system resources due to repeated retransmissions (up to 130% more transmissions than NADV). Also, the ADV performance degrades rapidly as bit error rates become higher. In contrast, by leveraging PER estimation, NADV enables graceful performance degradation.

Repeated retransmissions also affect the packet delay. Although not displayed here, the end-to-end latencies of ADV also show an increasing trend similar to Figure 8. Specifically, as the noise value changes from -91dBm to -88dBm, the average latency of ADV increases from 52.3ms to 201.6ms. NADV again exhibits graceful performance degradation (increase from 52.0ms to 67.7ms for NADV-Self, and from 51.6ms to 60.2ms for NADV-SNR).

Varying the Number of Data Flows: In the previous experiments, we use only one pair of source and destination. When we have more source-destination pairs, the network contention increases, in which the proposed techniques may estimate PER values incorrectly. For example, although received SNR values may be high, a node can experience high packet error rates due to increased collisions. To identify the performance of estimations schemes in this scenario, we vary the the number of data flows in the next set of experiments. We choose source-destination pairs uniformly at random. As in the previous experiments, we do not use the MAC-level notification of GPSR and report the average packet delivery ratio for each scenario. Among the values in Table I, we fix the noise value at -89.2dBm, which is the closest to the median of noise measurements in our building.

In Table IV, we report the average data delivery ratios in different scenarios. In these experiments, all NADV schemes perform similarly. Although the delivery ratio appears to decrease when we increase the number of flows, the amount of degradation is small. When we experimented using 16 flows without packet errors, we observed low delivery ratios due to the network saturation, and eight data flows in this setting corresponds to relatively high network utilization. These results show that our proposed estimation techniques work well in the presence of high network load. When we use ADV, the average delivery ratio

	Number of Flows			
	1	2	4	8
NADV-Beacon	98.6	98.7	97.3	97.6
NADV-SNR	99.2	99.3	97.9	98.1
NADV-Self	98.8	99.0	97.4	97.8
ADV	71.9	73.8	71.3	77.6

TABLE IV

DATA DELIVERY RATIO (IN %) WHEN THE NUMBER OF DATA FLOWS IS VARIED.

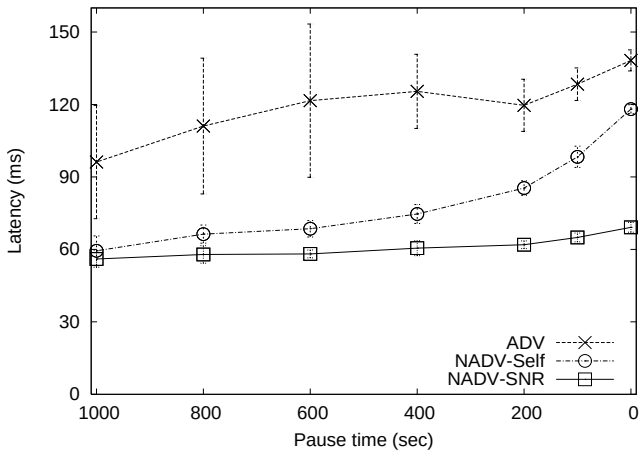


Fig. 9. Average end-to-end latency when nodes are mobile. C_{error} is used as link cost. We changed the pause time for different degrees of mobility. NADV and the proposed cost estimation schemes are effective with node mobility.

lies between 70% and 80%, depending on the random node placement.

Experiments with Mobile Nodes: In the previous results, we have shown that our proposed techniques for PER estimation perform well in static networks. We now investigate how well they adapt to network topology changes. In this scenario, the source and destination pair does not move, but the remaining 98 nodes move according to the random waypoint model. The speed is randomly chosen between 1 and 10 m/s, and we vary the pause time for different degrees of node mobility. We use the MAC-level failure notification and fix the ambient noise power at -89.2dBm.

In Figure 9, we present the end-to-end latency results with varying mobility. As mentioned before, the data transmission overhead shows a similar trend to Figure 9. We observe that average latencies increase as node mobility becomes higher. This is because frequent link failures cause more retransmissions. Compared to ADV, both NADV schemes achieve lower average latency. With NADV-SNR, PER estimation is more

Distance	500m	600m	700m	800m	900m
ADV	22.9	26.5	31.7	36.2	42.9
NADV$_{delay}$	14.5	17.3	20.0	22.7	26.2

TABLE V

AVERAGE END-TO-END LATENCY (IN MS) WITH DIFFERENT SOURCE-DESTINATION DISTANCES.

accurate, and the increase in end-to-end latency is minimal even with the highest mobility (50% lower compared to ADV). When NADV-Self is used in high mobility scenarios, most neighboring nodes move out of range before the estimated values can converge. As a result, the performance gain is smaller than in low mobility cases. Still, its average latency is 15% shorter than that of ADV when nodes move constantly. NADV-Beacon also requires a certain number of beacon messages for good estimation, and the results are similar to those of NADV-Self (within 5% difference in all cases), which we do not show here for clarity.

To summarize, the proposed link estimation schemes are effective even with node mobility, and NADV combined with them provides an efficient and adaptive geographic routing strategy. As the network environment becomes harsher, the performance of NADV degrades gracefully.

E. Using Delay as Link Cost

In this subsection, we use link delay as link cost and assume $NADV \equiv NADV_{delay}$ in this scenario. We use the error model using Eq. 9, and ARF is used for rate adjustment. In this model, due to the interaction with ARF, link cost is not a convex function. In this experiment, we use a low noise value of -91.0dBm in this set of simulations. Note that this scenario is in favor of ADV because with high noise, ADV suffers from increased end-to-end latency as previously discussed. The MAC-level failure notification is used, and the delivery ratios are 100% in all cases. Each value in this experiment is an average of ten runs.

In Table V, we report the average end-to-end latency of each scheme when we vary the distance between the source and the destination. As the distance increases, packets go through more relay nodes, and the latency increases accordingly. Compared to ADV, NADV significantly decreases the end-to-end latency (by up to 35%). It is because when we use ADV, we are likely to choose far neighbors to minimize the distance to the destination. However, the transmission rates to such nodes are usually 1 or 2 Mbps, which causes the transmission to take longer. With the use of NADV, the current node may choose a

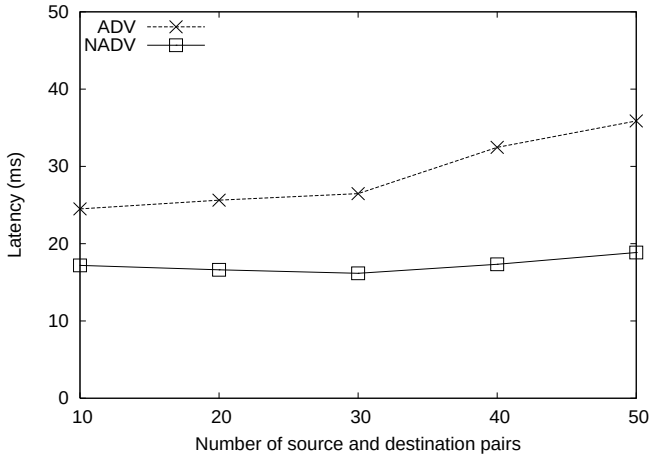


Fig. 10. Average end-to-end delay with multiple flows. ARF and C_{delay} are used. When NADV is used, the network can support more flows without significant increase in the latency.

neighbor that is not the closest to the destination, but the corresponding link is good enough for a higher transmission rate such as 11 Mbps. This strategy eventually leads to shorter transmission time.

When using *NADV*$_{delay}$ in this simulation scenario, the current node usually selects neighbors close to itself, which leads to more relay nodes (e.g., 55% increase when the distance is 900m). Since this increase is based only on the local decision to minimize the medium time, it may degrade the overall performance, especially when multiple traffic flows exist in the network. To investigate this potential problem, we perform experiments using different numbers of source-destination pairs, which we select uniformly at random.

In Figure 10, we plot the average end-to-end latency when we change the number of flows in the network. We can observe that with more flows in the network, ADV increases the average latency noticeably. This is because ADV holds the wireless medium longer than necessary, leading to a higher level of network contention. In contrast, NADV maintains the aggregate medium time low enough, such that the network can support more flows without significant increase in the latency. Consequently, compared to ADV, NADV improves the latency performance even more with higher network traffic load. Specifically, in the case of 10 flows, NADV decreases the average latency by 30%, but with 50 flows the improvement is 48%.[6] In the case of 50 flows, only 2 flows experience slight increase ($<$ 2ms) in the end-to-end delay.

[6]In the experiments for Table IV, we use the fixed data transmission rate of 1 Mbps, and we observe network saturation when we send more than 8 packets per second. In the experiments for Figure 10, the data transmission rate can be up to 11 Mbps, and NADV can support more data flows without network saturation.

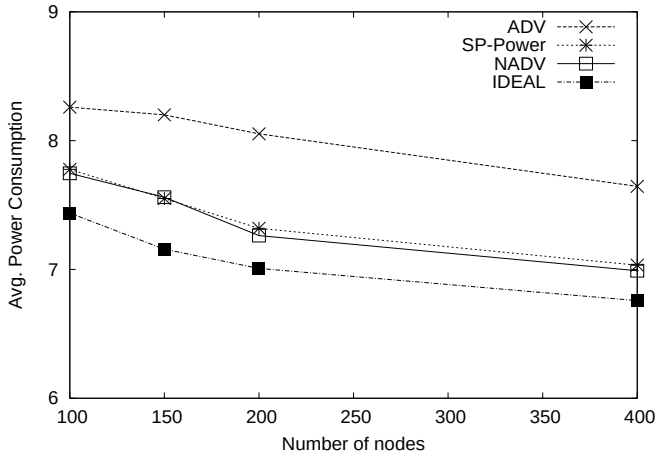


Fig. 11. Average power consumption with different schemes. In dense networks, as more neighbors are available, power consumption decreases. The power consumption values by NADV and SP-Power are similar, which are close to the optimal value.

This experiment result shows that the use of *NADV*$_{delay}$ does not negatively affect the performance of other traffic in the network.

F. Using Power Consumption as Link Cost

We compare *NADV* ($\equiv$ *NADV*$_{power}$) against the metric proposed in the *SP-Power* scheme [14]. When the power consumption equation is $C_{power} = 1 + c\,t_{px}$, NADV needs to know the current transmission power p_{tx}, which we assume is available through a control mechanism. SP-Power requires the exact value of path loss exponent, which we also assume is available. In practice, however, the path loss exponent estimation is not trivial, and depending on the measurement parameters, the estimated values can vary significantly [23], [38]. We assume that both schemes know the proportionality value c, which is a hardware-specific constant. In the following set of simulations, the distance between source and destination is 900m, and there are no packet errors. We vary the node density and use average values of 20 runs for each case. We also compare the performance of optimal paths found by the centralized algorithm.

In Figure 11, we plot the average power consumption of each scheme with different node density. The amount of power consumption in each scheme decreases as node density increases. This is because with higher node density, more neighbors become available, and all schemes likely choose better next hops. We also observe that compared to ADV, both NADV and SP-Power find paths that reduce overall

power consumption.[7] The performances of NADV and SP-Power are almost identical; NADV performs slightly better. (NADV and SP-Power find the same path in 15 cases out of 20 in the 400-node scenarios.) Even though we do not report detailed results in this paper, *NADV*$_{power}$ and SP-Power also show very similar performance in other settings (e.g., distance, continuous power adjustment, different path loss exponents, and proportionality constants c). For other aspects of energy consumption (e.g., in idle or receiving mode), we expect that *NADV*$_{power}$ and SP-Power will consume a similar amount of energy and that their performance will be close to each other as well.

When the goal of geographic routing is to minimize the path power consumption, we argue that *NADV*$_{power}$ is the metric of choice. *NADV*$_{power}$ and SP-Power are based on a similar rationale for next hop selection and exhibit almost identical performance. However, as mentioned above, SP-Power needs to estimate the path loss exponent, which can be difficult in practice. In contrast, *NADV*$_{power}$ only requires t_{px}, which nodes can easily determine with the support of existing control mechanisms [21], [23].

G. Experiments with Generic Cost

Recently, new metrics are being proposed for various multihop routing purposes. For example, Draves et al. [39] propose the WCETT (Weighted Cumulative Expected Transmission Time) metric to improve network throughput in multi-radio mesh networks. As multihop wireless networks become more widely used for different objectives, we expect to see other new routing metrics proposed to achieve specific goals. In this section, we apply NADV to a generic cost metric to see whether the use of NADV can be generalized to other types of link cost. We use the following link cost:

$$C_{generic} = 1.0 + r\left(\frac{d}{R}\right)^2, \ 1 \leq r \leq 5, \tag{11}$$

where r is a uniformly distributed random number, d is the distance between two nodes, and R is the maximum transmission range. The above link metric attempts to capture both the correlation with distance and the random property of link quality [28], [20]. In this subsection, we assume the availability of accurate and up-to-date link cost information.

We use the following experiment scenario. The source and the destination are 900 meters apart, and the source starts to send data packets after 10 seconds. At 30 seconds, we assume that the environment

[7]In Figure 11, the performance difference between the optimal case and ADV is not large. It is because the constant term in Eq. **??** constitutes a significant power consumption regardless of the transmission power, as is the case with most existing products [33].

	ADV	Non-adaptive (AODV)	NADV one-hop	NADV two-hop	IDEAL
Initial	14.43	11.14	11.28	10.82	10.32
After change	18.51	14.30	13.50	12.52	11.62

TABLE VI

THE AVERAGE COSTS OF PATHS FOUND BY RESPECTIVE ROUTING SCHEMES WHEN LINK COSTS CHANGE.

of some part of the network changes (e.g., due to new obstacles, increased interference, node mobility), and we randomly select 50% of links and increase their link costs by 50%. For NADV, we additionally consider a geographic routing scheme that uses two-hop neighborhood information [10]. To compare NADV against AODV [13], we modify the AODV simulation code, such that AODV finds paths that minimize the sum of link costs along the paths, not hop count.

In Table VI, we report average path quality of each scheme before and after the link cost change. Each value in the table is an average of ten experiments. In this table, we can see that using NADV, geographic routing (both one-hop and two-hop) can find paths comparable to the optimal paths. Not surprisingly, utilizing two-hop neighborhood information leads to higher-quality paths than the one-hop case. The performance of initial paths by AODV lies between those by one-hop NADV and two-hop NADV. However, even after some link cost values increase after 30 seconds, AODV keeps using the initial path, and the path performance degrades accordingly. In contrast, the use of NADV enables localized geographic routing to detect the change and determine better next hops, which results in better paths.

In summary, geographic routing with NADV can find paths whose costs are comparable to the optimum. It is also able to adapt to network environment changes, due to the localized next hop decision.

V. TESTBED EXPERIMENTS

In this section, we present results from our experiments performed on real testbeds and demonstrate that our estimation strategy performs well in various wireless environments. After describing experiment setup, we first show the PER estimation scheme based on two-state Markov model works well in practice, and then present results when we employ the scheme in the NADV framework in practical scenarios.

Fig. 12. Partial floorplan for the Emulab wireless testbed. Nodes 1 and 9 cannot directly communicate, and all the other node pairs can talk to each other.

A. Experiment Setup

We have performed our experiments in two open access wireless testbeds: Emulab (`http://www.emulab.net`) and ORBIT (`http://www.orbit-lab.org`). Although Emulab is often used to provide emulated network environments for wired networks experiments, the Emulab wireless testbed uses *real air communication* through IEEE 802.11 wireless interfaces between stationary PC nodes scattered around a typical office building. We only use the nodes on the third floor (Figure 12). Each PC has two Netgear WAG311 wireless interface cards based on the Atheros 5212 chipset. It uses Redhat 9.0 with 2.4 kernel and the MadWifi open-source device driver[8]. The ORBIT testbed currently consists of 400 wireless nodes, each equipped with two IEEE 802.11 wireless cards laid out in a 20-by-20 grid with approximately one meter spacing between nearby nodes. Due to the relatively small deployment area, observed packet error rates in ORBIT show less diversity [40]. Thus, we focus on results from Emulab to illustrate that the estimation technique performs well for both low-error and high-error links.

In our experiments, a sender broadcasts 16, 32, 64, 128, 256, 512 and 1024-byte UDP packets every 0.05 seconds in an intermixed fashion to minimize the effect of link condition variation over time on the error rates of different message types. In our experiments, we use only one sender at any instant to minimize the interference and collisions. Each sender broadcasts 10000 packets for each size (70000 packets total). All nodes receiving the packets record the packet size and sequence number to calculate the observed PERs for each message type. In this paper, we use the fixed transmission rate of 1 Mbps for all messages. Investigating the impact of different data rates is an area of our future research. The

[8] `http://www.madwifi.org`

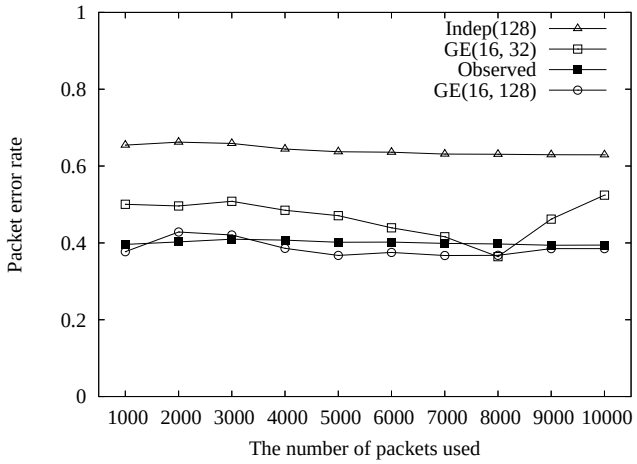


Fig. 13. Estimated and observed PERs for 1024-byte packets over the link from node 1 to node 4 in Emulab (Figure 12).

transmit power is fixed at 31 mW, which is the default value in the device driver.

We compare the estimation performance of the following strategies:

- BASIC(m): This scheme uses the average error rate of m-byte probe messages for data packets of all sizes.
- INDEP(m): This scheme assumes the independent bit error model and extrapolates the expected packet error rate based on Eq. 5 and the statistics of m-byte probe messages.
- GE(m, n): This scheme is based on the GE model, which uses Eq. 6 and the statistics of m-byte and n-byte probe messages.
- OBSERVED: This is the actual observed packet error rate.

Only one measurement value is required for INDEP; GE uses two parameters, and there can be more possible combinations of the two. For both schemes, proper parameter choice can be crucial to correct PER estimation. We consider three different combinations of parameters for GE and two different cases for INDEP and compare the estimation performance.

B. PER Estimation Results

We first consider how well the above estimation strategies perform. In Figure 13, we plot the observed error rate for 1024-byte packets and estimated error rates by different schemes[9]. We use a representative experiment sending 10000 packets for each probe type, and each point in the figure is based on cumulative packet error rates after every 1000 packets. In Figure 13, the estimation by GE(16,128) closely matches the

[9]We include additional 84 bytes of lower layer headers in the calculation.

	Emulab Links						
	8→9	1→13	1→7	1→4	1→8	11→16	16→5
OBSERVED	0.018	0.135	0.145	0.334	0.375	0.548	0.738
GE(16,128)	**0.021**	**0.131**	**0.145**	**0.385**	**0.393**	**0.526**	**0.754**
GE(16,64)	0.025	0.222	0.247	0.465	0.332	0.415	0.791
GE(16,32)	0.046	0.154	0.043	0.524	0.243	0.594	0.677
INDEP(128)	0.052	0.222	0.255	0.629	0.645	0.907	0.996
INDEP(16)	0.092	0.332	0.383	0.816	0.831	0.993	1.000
BASIC(128)	0.010	0.047	0.055	0.173	0.180	0.385	0.646

TABLE VII

COMPARISON OF DIFFERENT ESTIMATION TECHNIQUES AGAINST ACTUAL PACKET ERROR RATES. WE USE 10000 PACKETS FOR EACH OF PROBE AND DATA MESSAGE TYPES. VALUES IN BOLD REPRESENT THE CASES WITH MINIMUM ESTIMATION ERROR.

actual average packet error rate. In general, we observe that the estimation error for GE(16,128) becomes smaller as we use more probe messages; we discuss this issue later in more detail.

In our experiments, GE(16,32) does not perform as well as GE(16,128). In Figure 13 there is considerable difference in the estimated value over time, and the measurement error is often relatively large. One possible explanation is that the estimation by GE(16,32) is less robust because we use extrapolation based on two relatively nearby sample points; a small measurement error can amplify the estimation error. Also, Kopke et al. [24] find that there is difference in bit error probability depending on the bit position, and bit errors occur more frequently at the beginning of a packet. As a result, estimation using short probe messages alone can potentially lead to higher estimation errors. In Figure 13, INDEP does not estimate PER correctly, and although not shown in the figure, the estimation error by INDEP(16) is larger than that of INDEP(128). Although we do not show all the results here, we have experimented with other links and performed multiple experiments for each link, and the results are similar. We later present some of them in Table VII. We have also performed experiments on the ORBIT testbed and gotten similar results which are omitted due to the limited space. Interested readers are referred to [40] for more experiment results. In the rest of this section, we use results from Emulab only.

Experiments with Various Links: In the previous results, we considered results only from a few links. We now present results from various wireless links with diverse link quality. In Table VII, we report

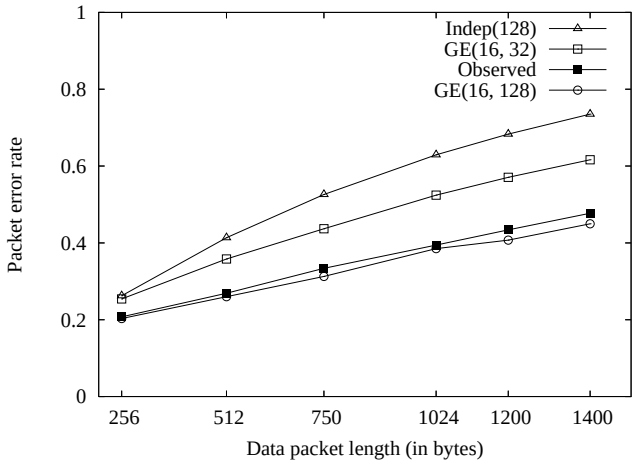


Fig. 14. PER estimation for different packet sizes. We use the link from node 1 to node 4. Again, GE(16,128) performs best for all packet sizes.

estimated PERs by different schemes as well as observed error rates for 1024-byte packets[10]. We observe that GE(16,128) estimation is the most accurate in all cases (highlighted in bold), and the estimation error is small regardless of link quality. GE(16,64) often performs better than GE(16,32), but both of them result in larger estimation errors than GE(16,128). As in Figure 13, INDEP leads to large estimation errors, while INDEP(128) performs better than INDEP(16). Although the independent bit error model has served as a reasonable model in [20], it does not seem to reflect the channel characteristics correctly in our indoor experiments. BASIC(128) uses the error rate of 128-byte probe messages as the estimation for 1024-byte packets, which results in significant underestimation. In Section V-C, we illustrate that this underestimation by BASIC can lead to significant inefficiency when used with existing routing schemes.

Varying Data Packet Sizes: In the previous experiments, we fixed the data packet length to 1024 bytes. In this set of experiments, we vary the data packet size and compare the estimated and observed error rates. In this experiment, we use additional packet sizes (750, 1200, and 1400 bytes). In Figure 14, we plot the estimated and actual packet error rates with varying packet sizes. We use the statistics of 10000 messages for each probe type. Not surprisingly, average packet error rates increase as data packets become larger. We observe that GE(16,128) again performs best in estimating error rates for all packet sizes. Other schemes show similar trends to Figure 13; GE(16,64) performs worse than GE(16,128), while INDEP performs worst. This result illustrates that our proposed technique estimates error rates for various packet sizes.

[10]Nodes 5, 11, and 16 are not shown in Figure 12. The full floorplan is available at `https://www.emulab.net/floormap.php3`.

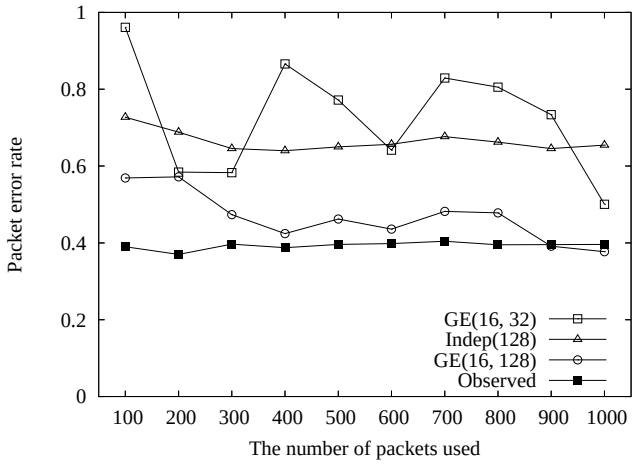


Fig. 15. PER estimation based on 1000 packets. We use the link from node 1 to node 4 again, but the values on the X-axis are smaller than those in Figure 13.

Convergence Time for Accurate Estimation: One of our goals is to estimate error rates quickly with small overhead. In this set of experiments, we look into the number of probe messages needed to achieve reasonable estimation accuracy. As mentioned before, small measurement errors in probe messages can cause large extrapolation errors. However, sending 1000 probe messages (to obtain the first point in Figure 13) takes tens of minutes if we send a probe message every second. In this experiment, we calculate the estimation using the cumulative statistics after every 100 probe messages. One issue with inferring error rates based on a small number of packets is that the observed error rate of 16-byte packets is sometimes higher than that of 128-byte packets, which is contrary to the trend shown in Figure 14. In that case, when we apply Eq. 6, the estimation is often negative for longer data packets. Clearly, it is due to limited number of samples, and we are unlikely to have a good estimation by blindly applying Eq. 6 with only a small number of probe messages. In such a case, we use the maximum of the following three values as the estimated error rate: PER(16), PER(128), and the estimated PER from Eq. 6.

In Figure 15, we consider the estimation performance when we use a smaller number of probe messages. We observe that GE(16,128) converges after 300 probe messages, while GE(16,32) shows a substantial amount of fluctuation. Still, GE(16,128) takes several minutes before achieving reasonable convergence if we send a probe message every second. This amount of time is acceptable for more static wireless mesh networks [28], while more dynamic wireless networks such as ad hoc networks may require faster convergence. We plan to investigate how to reduce the number of required probe messages further in the future.

Next Hop	ADV	NADV		Number of Retransmissions		
		BASIC(128)	GE(16,128)	at source	at relay	total
4	19.16	15.85	11.78	536	188	724
7	25.96	24.54	**22.19**	185	119	**304**
8	**30.00**	**24.61**	18.20	768	4	772
13	20.55	19.59	17.85	250	171	421

TABLE VIII

ROUTING METRICS BASED ON DIFFERENT ESTIMATION SCHEMES AND ACTUAL ROUTING PERFORMANCE. VALUES IN BOLD CORRESPOND TO THE BEST CHOICES UNDER DIFFERENT CRITERIA. THE DELIVERY RATIOS FOR ALL CASES ARE OVER 99.9% DUE TO MAC-LEVEL RETRANSMISSIONS FOR UNICAST MESSAGES.

C. Experiment Results with NADV

We have modified the geographic routing implementation from USC[11] to account for link cost when choosing next hops. We installed the modified code at the nodes shown in Figure 12. In our experiments, node 9 is the destination, and node 1 is the source sending 1000 UDP packets (1024 bytes each) at the rate of 20 packets per second. We use the IEEE 802.11 MAC protocol, and the MAC-level transmit data rate is fixed at 1 Mbps. Depending on the estimation strategies we combined with the routing metric, we can potentially choose different next hops. For each case, we measure the average delivery ratio and number of total retransmissions (overhead). In some of our experiments, we force the routing code to choose a particular next hop to compare the performance. We use an internal variable (*LongRetryCount*) in the MadWifi device driver to retrieve the total number of MAC-level retransmissions. To maintain consistency with the results in Table VII, we use the estimated values in the table as fixed link cost when choosing the next hop. We compare the performance when we use BASIC(128), INDEP(128), and GE(16,128)[12].

In Table VIII, we present (1) routing metrics for each neighbor from the source node 1 when using different estimation schemes and (2) the number of MAC-level data retransmissions when choosing different nodes as the next hop. Node 1 sends 1000 UDP packets total. For the ADV metric, since we do not consider link quality, we choose node 8, which is closest to the destination node 9. When

[11] Available at `http://enl.usc.edu/software.html`

[12] In the implementation we use, periodic messages use 16-byte UDP packets. If periodic messages include neighbor information such as reverse link quality and location information [10], the size of periodic message will be easily over 128 bytes even with a few neighbors. Therefore, our scheme can be implemented without introducing additional overhead.

we use NADV based on BASIC(128), error rates for 1024-byte packets are underestimated, and node 8 is chosen as the best next hop. However, the actual data packet error rate for the link to node 8 is significantly higher (37.5% for 1024-byte packets vs. 18.0% for 128-byte packets), and using node 8 as relay node leads to multiple packet retransmissions due to losses. In contrast, when we use NADV and GE(16,128), we can estimate the actual error rate more accurately. Consequently, we can transfer data messages with minimum data overhead; when using node 7 as relay, we experience 304 retransmissions, which is only around 40% of the case of using node 8 as relay (304 vs. 772). Although not shown here, when using INDEP(128) and NADV, node 7 is still chosen as next hop. However, this selection is based on incorrect PER estimation (25.5% for INDEP(128) estimation vs. 14.5% for observed PER). Such incorrect estimation by INDEP can potentially eliminate the use of links with reasonable quality, which will be often suboptimal. Thus, we expect that INDEP will not work well in other scenarios, and we plan to perform more experiments in various settings.

Although not comprehensive, we believe the results in this section indicate that our proposed scheme can achieve significant performance improvement in practice.

VI. Related Work

Many ideas and techniques have been proposed to find minimum-cost paths in multihop wireless networks, and energy-efficient routing has been an area of intensive research. Rodoplu et al. [19] present a localized algorithm that preserves network connectivity and achieves the globally minimum-energy topology. In PARO [41], a node becomes a relay node if it finds that the relaying leads to lower energy consumption. Given traffic flows and node energy levels, Chang et al. [42] find a set of routes that maximize the system lifetime. More recently, wireless link errors has drawn much attention in multihop wireless networks [18]. Banerjee et al. [22] propose the use of a link metric based on link error probability. De Couto et al. use a similar metric called *ETX (Expected Transmission Count)* in real testbed experiments, and their experiment results show that paths with smaller ETX perform better than shortest paths [15]. These techniques and metrics above typically focus on table-driven or on-demand routing protocols (e.g., AODV [13]); in contrast, our work provides a general framework to incorporate these metrics into geographic routing.

Traditional geographic routing schemes use only geometric information such as the length of projection (called *progress*) and angle value against the straight line between source and the destination (please

see [5] and the references therein). Instead of a straight line, Niculescu et al. [4] propose a forwarding strategy based on a pre-defined curve. More recent schemes consider link costs in the next hop selection. Stojmenovic et al. [14] propose a routing metric for power-efficient routing. Seada et al. [11] focus on the minimum energy consumption in lossy environments and propose threshold-based schemes as well as a link metric in Eq. 4. Zorzi and Armaroli also propose the same link metric [12]. Our work is different from them in that we present a more general framework and provide the rationale behind the use of NADV by proving the optimal tradeoff between hop count and link cost.

Greedy forwarding using NADV still can result in the local minimum problem. To route packets around voids, we can use existing recovery schemes with the NADV metric. For example, *Face Routing* [1] uses the right-hand rule in Gabriel graph, and GPSR employs a similar scheme called *perimeter mode* [2]. Terminode routing uses *Anchored Geodesic Packet Forwarding (AGPF)* similar to loose source routing [9]. Kuhn et al. present GOAFR+, which is efficient on average cases and worst-case optimal [3]. (Although GOAFR+ considers link cost, it still chooses the neighbor closest to the destination in greedy mode.) For example, in recovery mode, Terminode routing and GOAFR+ find an intermediate node, to which packets are forwarded in a greedy manner. Those schemes can use NADV in their recovery phase. In other cases, recovery algorithms are independent of greedy forwarding procedure, so using NADV does not affect the recovery performance. Also, we believe that compared to ADV, greedy forwarding using NADV will have a similar frequency of encountering voids in a given network. The NADV metric can also be used in geocasting, which is similar to multicast, but delivers data packets to nodes located inside a certain region [8]. Geographic routing may exploit location service systems [7] and location computation systems [6]. More information about position-based routing can be found in [5].

VII. Conclusions and Future Work

We have introduced NADV as link metric for geographic routing in multihop wireless networks. Geographic routing with NADV provides an adaptive routing strategy, which is general and can be used for various link cost types. We have presented techniques for link cost estimation. We have performed extensive simulation study and testbed experiments to evaluate the effectiveness of NADV and link cost estimation techniques. In these environments, the combination of NADV and cost estimation techniques outperforms the current geographic routing scheme. NADV also finds paths whose cost is close to the optimum.

Cost-aware routing schemes including NADV benefit greatly from fast and accurate link cost estimation, and we plan to investigate this issue further in the future. In this paper, we have treated each link cost type independently. However, if we consider multiple interdependent costs simultaneously, choosing the next hop based on one cost type may not be always the best choice for other costs. Our future work is to design a link cost model that balances multiple cost criteria, which would allow the NADV framework to leverage the combined link cost to find a low cost path.

REFERENCES

[1] Prosenjit Bose, Pat Morin, Ivan Stojmenovic, and Jorge Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," in *Proceedings of the 3rd International Workshop on Discrete algorithms and methods for mobile computing and communications*. 1999, ACM Press.

[2] Brad Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th ACM/IEEE MobiCom*. 2000, pp. 243–254, ACM Press.

[3] Fabian Kuhn, Roger Wattenhofer, Yan Zhang, and Aaron Zollinger, "Geometric ad-hoc routing: of theory and practice," in *Proceedings of the 22nd annual symposium on Principles of distributed computing*. 2003, pp. 63–72, ACM Press.

[4] Dragos Niculescu and Badri Nath, "Trajectory based forwarding and its applications," in *Proceedings of the 9th ACM/IEEE MobiCom*. 2003, pp. 260–272, ACM Press.

[5] Ivan Stojmenovic, "Position-based routing in ad hoc networks," *IEEE Communications Magazine*, pp. 128–134, July 2002.

[6] Jeffrey Hightower and Gaetano Borriello, "Location systems for ubiquitous computing," *IEEE Computer*, vol. 34, no. 8, pp. 57–66, 2001.

[7] Jinyang Li, John Jannotti, Douglas S. J. De Couto, David R. Karger, and Robert Morris, "A scalable location service for geographic ad hoc routing," in *Proc. of ACM MobiCom*, 2000.

[8] Young-Bae Ko and Nitin H. Vaidya, "Geocasting in mobile ad hoc networks: Location-based multicast algorithms," in *Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications*. 1999, IEEE Computer Society.

[9] L. Blazevic, S. Giordano, and J. Y. Le Boudec, "Self organized terminode routing," *Journal of Cluster Computing*, vol. 5, no. 2, April 2002.

[10] I. Stojmenovic and X. Lin, "Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 10, pp. 1023–1032, Oct. 2001.

[11] Karim Seada, Marco Zuniga, Ahmed Helmy, and Bhaskar Krishnamachari, "Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. 2004, pp. 108–121, ACM Press.

[12] M. Zorzi and A. Armaroli, "Advancement optimization in multihop wireless networks," in *Proceedings of VTC*, Oct. 2003.

[13] C.E. Perkins and E.M. Belding-Royer, "Ad hoc on-demand distance vector (AODV) routing," in *IEEE Workshop on Mobile Computing Systems and Applications*, Feb. 1999.

[14] Ivan Stojmenovic and Xu Lin, "Power-aware localized routing in wireless networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 11, pp. 1122–1133, 2001.

[15] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris, "A high-throughput path metric for multi-hop wireless routing," in *Proceedings of the 9th ACM/IEEE MobiCom*, 2003.

[16] Abtin Keshavarzin, Elif Uysal-Biyikoglu, Falk Herrmann, and Arati Manjeshwar, "Energy-efficient link assessment in wireless sensor networks," in *Proc. of IEEE Infocom*, March 2004.

[17] Tommaso Melodia, Dario Pompili, and Ian F. Akyildiz, "Optimal local topology knowledge for energy efficient geographical routing in sensor networks," in *Proc. of Infocom*, March 2004.

[18] Henrik Lundgren, Erik Nordstr, and Christian Tschudin, "Coping with communication gray zones in IEEE 802.11b based ad hoc networks," in *Proceedings of the 5th ACM international workshop on Wireless mobile multimedia*, 2002, pp. 49–55.

[19] Volkan Rodoplu and Teresa H. Meng, "Minimum energy mobile wireless networks," *IEEE JSAC*, vol. 17, no. 8, pp. 1333–1344, Aug. 1999.

[20] Marco Zuniga and Bhaskar Krishnamachari, "Analyzing the transitional region in low power wireless links," in *Proceedings of IEEE SECON*, Oct. 2004.

[21] IEEE 802.11h Standard, "Part 11. Amendment 5: Spectrum and transmit power management extensions in the 5GHz band in Europe," 2003.

[22] Suman Banerjee and Archan Misra, "Minimum energy paths for reliable communication in multi-hop wireless networks," in *Proceedings of the 3rd ACM MobiHoc*, 2002, pp. 146–156.

[23] Theodore Rappaport, *Wireless Communications: Principles and Practice (2nd Edition)*, Prentice Hall, 2001.

[24] Andreas Kopke, Andreas Willig, and Holger Karl, "Chaotic maps as parsimonious bit error models of wireless channels," in *Proceedings of Infocom*, Apr. 2003.

[25] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell Systems Technical Journal*, vol. 39, pp. 1253–1265, 1960.

[26] E. O. Elliot, "Estimates of error rates for codes on burst-noise channels," *Bell Systems Technical Journal*, vol. 42, pp. 1977–1997, 1963.

[27] Kyu-Han Kim and Kang G. Shin, "On accurate measurement of link quality in multi-hop wireless mesh networks," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*, New York, NY, USA, 2006, pp. 38–49, ACM.

[28] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris, "Link-level measurements from an 802.11b mesh network," in *ACM SIGCOMM*, Sept. 2004.

[29] Kameswari Chebrolu, Bhaskaran Raman, and Sayandeep Sen, "Long-distance 802.11b links: performance measurements and experience," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*, New York, NY, USA, 2006, pp. 74–85, ACM.

[30] Seungjoon Lee, Suman Banerjee, and Bobby Bhattacharjee, "The case for multi-hop wireless local area network," in *Proceedings of Infocom*, Mar. 2004.

[31] Baruch Awerbuch, David Holmer, and Herbert Rubens, "High throughput route selection in multi-rate ad hoc wireless networks," in *First Working Conference on Wireless On-demand Network Systems (WONS)*, 2004.

[32] Seungjoon Lee, Bobby Bhattacharjee, and Suman Banerjee, "Efficient geographic routing in multihop wireless networks," in *Proceedings of MobiHoc*, 2005, pp. 230–241.

[33] Rex Min and Anantha Chandrakasan, "Top five myths about the energy consumption of wireless communication," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, no. 1, pp. 65–67, 2003.

[34] Suresh Singh, Mike Woo, and C. S. Raghavendra, "Power-aware routing in mobile ad hoc networks," in *Proceedings of the 4th ACM/IEEE MobiCom*. 1998, pp. 181–190, ACM Press.

[35] IEEE 802.11 Standard, "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," 1999.

[36] "Cisco aironet 350 series client adapters data sheet," June 2003, Cisco Systems Inc. Available at http://www.cisco.com/.

[37] Omprakash Gnawali, Mark Yarvis, John Heidemann, and Ramesh Govindan, "Interaction of retransmission, blacklisting, and routing metrics for reliability in sensor network routing," in *Proceedings of the First IEEE Conference on Sensor and Ad hoc Communication and Networks*, Santa Clara, California, USA, October 2004.

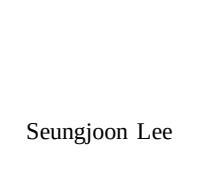
[38] Scott Y. Seidel, Theodore S. Rappaport, Sanjiv Jain, Micheal L. Lord, and Rajendra Singh, "Path loss, scattering, and multipath delay statistics in four European cities for digital cellular and microcellular raiodtelephone," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 4, pp. 721–730, November 1991.

[39] Richard Draves, Jitendra Padhye, and Brian Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*. 2004, pp. 114–128, ACM Press.

[40] Bo Han and Seungjoon Lee, "Efficient packet error rate estimation in wireless networks," in *Proceedings of the 3rd IEEE/CREATE-NET International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom 2007)*, May 2007.

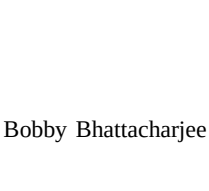
[41] J. Gomez, A. Campbell, M. Naghshineh, and C. Bisdikian, "PARO: Supporting transmission power control for routing in wireless ad hoc networks," *ACM/Baltzer Journal on Mobile Networks*, 2002.

[42] Jae-Hwan Chang and L. Tassiulas, "Energy conserving routing in wireless ad-hoc networks," in *Proc. of IEEE Infocom*, 2000.



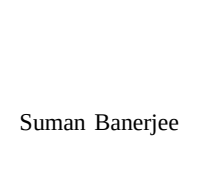
Seungjoon Lee

Seungjoon Lee received his Bachelor's and Master's degrees in Computer Science from Seoul National University, Seoul, Korea, in 1996 and 2000 and his Ph. D. in Computer Science from the University of Maryland in 2006. Currently, he is a senior member of technical staff in AT&T Labs, Research. His research interests include wireless networks, mobile computing, peer-to-peer systems, and multicasting.



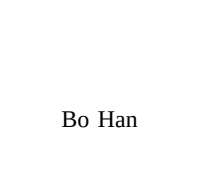
Bobby Bhattacharjee

Bobby Bhattacharjee received Bachelor's degrees in Computer Science and Mathematics from Georgia College in 1994 and his Ph. D. in Computer Science from the College of Computing at Georgia Tech. in 1999. He is currently an Associate Professor in the Department of Computer Science at the University of Maryland. Dr. Bhattacharjee's research interests are in the design and implementation of wide-area networking, distributed systems, and security protocols. His current focus is on the design of decentralized secure systems for multi-party applications especially in the context of peer-to-peer and overlay systems.



Suman Banerjee

Suman Banerjee received the B.Tech. degree in computer science and engineering from the Indian Institute of Technology, Kanpur, India, in 1996, and the M.S. and Ph.D. degrees in computer science from the University of Maryland, College Park, in 1999 and 2003, respectively. He has been an Assistant Professor in the Department of Computer Sciences, University of Wisconsin-Madison since January 2004. He received an NSF Career Award in 2008. He currently leads the WiNGS laboratory at UW-Madison that conducts research in wireless and mobile networking systems. His other areas of interest include overlay architectures, peer-to-peer systems, network measurements, and security.



Bo Han

Bo Han received the Bachelor's degree in Computer Science and Technology from Tsinghua University in 2000 and the M.Phil. in Computer Science from City University of Hong Kong in 2006. He is currently a Ph.D. student in the Department of Computer Science at the University of Maryland, College Park. He worked as research intern at AT&T Labs, Research for summer 2007 and 2008. His research interests include wireless communication, distributed algorithms and internet computing.