

## Physical Layer Impact on Finding Local Knowledge Information in Ad Hoc and Sensor Networks

Nishith Goel<sup>1</sup>, Kalai Kalaiichelvan<sup>2</sup>, Eric Karmouch<sup>3</sup>, Amiya Nayak<sup>3</sup>,  
Ivan Stojmenovic<sup>4,3\*</sup>, Eduardo Villanueva-Pena<sup>3</sup>

<sup>1</sup> Cistel Technology Inc., Ottawa, ON K2E 7V7  
ngoel@cistel.com

<sup>2</sup> EION Inc., Ottawa, ON K1Y 2X5  
kalai@eion.com

<sup>3</sup> SITE, University of Ottawa, Ottawa, Ontario K1N 6N5, Canada  
{ekarm041, anayak, ivan, pvillanu}@site.uottawa.ca

<sup>4</sup>EECE, University of Birmingham, United Kingdom

**Abstract.** Most existing network layer protocols in ad hoc and sensor networks require the use of topological or geographical knowledge of neighboring nodes in order to make proper decisions. Such protocols obtain this knowledge following the *unit disk graph model*, whereby two nodes are neighbors if and only if the distance between them is at most  $R$ , where  $R$  is the transmission radius of a node ( $R$  is equal for all nodes). Moreover, it is assumed that two nodes are able to successfully receive trivial *hello* messages from each other, used for information exchange, if and only if they are neighbors (i.e. within a distance  $R$  of each other). A problem now arises if a realistic physical layer is applied, such that the previous assumption no longer holds. To the best of our knowledge, the problem of gathering neighbor information using a realistic physical layer has yet to be addressed. In this paper we propose two protocols addressing this problem. Both protocols are described in detail and tested over collision-free static networks. We also incorporate them in localized position based routing protocols.

**Keywords:** ad hoc wireless networks, sensor networks, routing, broadcasting, physical layer.

\* Corresponding author: Stojmenovic@storm.ca.

## 1 Introduction

Ad hoc and sensor networks consist of autonomous nodes, which can be self-organized or attached to fixed infrastructures, such as a sink in sensor networks or the Internet in ad hoc networks. Due to possible node mobility and changes in sleep-active periods, the topology of these networks change very frequently. Since the majority of existing network layer protocols use information gathered from neighboring nodes to make communication decisions (e.g. forwarding), accurate neighborhood information is essential. The simplest method for collecting and providing neighborhood information is through the exchange of *hello* messages containing respective node information, such as a node's position, ID, and list of neighbors (if two-hop local information is required by given protocol). In ad hoc networks, all existing literature makes the assumption that all nodes within the transmission radius of a node will correctly receive its hello messages. However, this assumption does not hold if a realistic physical layer is applied. The shadowing model is an example of a model which applies a realistic physical layer. Applying the shadowing model, after a node sends a message, each neighboring node receives it with probability  $p(x)$ , where  $p(x)$  depends on the receiving node's distance  $x$ , as well as environmental conditions (this article, however, considers only dependence on distance). Therefore, not all neighbors receive the hello message.

The protocols and discussions presented in this paper are mainly concerned with the cost of sending additional hello messages with respect to the gains achieved by doing so. In particular, we examine the performance (in terms of success rates and expected hop counts) of position based localized routing protocols, whereby each node selects one neighbor to forward a message to among all neighbors from which hello messages were received (i.e. from neighbors the node is aware of). In order to collect accurate local knowledge information in ad hoc networks over a realistic physical layer, we propose two protocols. The first protocol is the *s-hello protocol*, which attempts to increase network density by asking every node in the network to send a fixed amount  $s$  of hello messages. The second protocol is the *target density protocol*, which realizes that not all nodes in the network need to send the same amount of hello messages to achieve a large enough network density (i.e. hearing from sufficient number of neighbors) to meet routing performance criteria; thus, it proposes to send hello messages until reaching a predefined target density providing the desired routing success rate. These two protocols are tested over static networks with no traffic collisions, and a MAC layer that performs according to the expected hop count formula. We consider a pure network layer problem and leave the investigation of the impact of medium access and transport layers for future investigations.

The rest of the paper is organized as follows. In Section 2, related work is presented, including a discussion of various propagation models and the localized routing protocol used in our experiments. In Section 3 and 4, the two protocols for collecting local information are described. Sections 5 and 6 present the results of our experimental study, which is followed by a brief discussion in Section 7. Conclusions and future work are provided in Section 8.

## 2 Related Work

### 2.1 Routing

There exists a vast amount of literature devoted to position based routing in ad hoc networks. A survey of position based routing schemes is given by [3]. In a localized routing algorithm, each node makes forwarding decisions based solely on the location of itself, its neighboring nodes, and the destination node. Finn [9] proposed a localized greedy scheme, where any node holding a message will forward it to the neighbor that is the closest to the destination node. Only neighbors that are closer to the destination than the current node are considered. In [10], a greedy and a face algorithm techniques are combined to achieve a localized greedy-face-greedy (GFG) algorithm, which guarantees delivery under an ideal MAC layer, and correct position information for the current node, its neighbors, and destination. CFG applies a greedy algorithm whenever possible, and face routing in recovery mode. Face routing makes use of a planar graph to route from face to face between source and destination nodes.

The impact of a realistic physical layer for different propagation models, when applying routing protocols, is investigated in [1, 2, 4, 5, 6, 8]. Routing protocols studied in [1, 2, 5, 6, 7] are based on a route discovery procedure, where neighborhood information is not required and each node that receives a route request packet will retransmit it once. However, in the case of localized position based routing protocols [3, 4, 10, 11], the correct geographic knowledge of the position of neighbors is essential for making the best routing decisions and to increase the probability of successful routing.

### 2.2 Realistic Physical Layer

Efficient protocols that consider realistic physical layers and correctly collect neighborhood information using techniques other than simply sending periodical hello messages (e.g. as specified by RFC 1256 for IEEE 802.11 protocol, with Agent advertisement and Agent solicitation protocols), have yet to be described in literature.

Most of the published reports on routing and broadcasting over ad-hoc networks make use of simplistic and idealistic models (e.g. unit disk graph) where the radio transmission range shapes a perfect circle. The free space propagation model assumes that the transmitter and the receiver have a clear line-of-sight. Therefore, the received signal strength depends on distance only. The two-ray ground reflection model considers both the direct path and a ground reflection path between the transmitter and receiver. This model is more accurate at long distances than the free-space propagation model. However, in real scenarios, the received signal strength is not only dependent on the distance between the transmitter and the receiver, but also on the environment. Moreover, the successful transmission of a packet does not guarantee that the next packet, over the same link,

with the same environmental conditions, will be successfully received. Rather, the reception depends on the reception threshold and some random events.

More realistic propagation models such as Rayleigh Fading [8], SIRCIM [6], and Log-Normal Shadowing [1] have been described in literature. Following [1] and [4], we adopt, in our evaluation, the Log-Normal Shadowing Model, where the noise is modeled by a Gaussian distribution. A brief description of the Log-Normal Shadowing Model is given below.

The Log-Normal Shadowing Model also known as the shadowing model is a statistical model. The mean received power  $Pr(d)$  at distance  $d$  is computed relative to  $Pr(d_0)$  (in dB) as:

$$\left[ \frac{Pr(d)}{Pr(d_0)} \right] db = -10\beta \log\left(\frac{d}{d_0}\right) + X_\sigma$$

This model consists of two main components. The first component is the path loss model, where  $d_0$  is the reference distance, and  $\beta$  is the loss exponent ( $2 \leq \beta \leq 6$ ). The second component is  $X_\sigma$ ; the variation of the power received at a certain distance.  $X_\sigma$  is a Gaussian random variable with zero mean and standard deviation  $\sigma$ , called the shadowing deviation. Both  $\beta$  and  $\sigma$  are obtained by measurements.

The probability of correct packet reception can be calculated as follows [1]. Based on the shadowing model, the average path loss for an arbitrary distance can be expressed as a function of distance  $d$ :

$$P_r(d)[dBm] = P_t[dBm] - \overline{PL}(d_0) - 10\beta \log\left[\frac{d}{d_0}\right] - X_\sigma$$

where  $X_\sigma$  is the zero mean random variable with standard deviation  $\sigma$  and  $P_t$  is the transmitted signal power. The shadowing model can be used for area coverage calculations, and the probability that the received power at a location  $d$  exceeds the reception threshold  $\gamma$  can be given as:

$$\Pr[P_r(d) > \gamma] = 0.5 \left( 1 - \operatorname{erf} \left[ \frac{\gamma - \overline{P_r(d)}}{\sqrt{2}\sigma} \right] \right)$$

This indicates that signals fade with distance and noise, and that there is a probability for proper signal reception at the receiver node. This is the probability  $b(x)$  of receiving a bit successfully, without assuming the existence of any error correcting scheme. The probability of packet reception  $p(x)$  depends on the length  $L$  of the packet and is given by  $p(x) = b(x)^L$ . The packet transmission radius  $R$  is defined as the distance for which  $p(R) = 0.5$  is satisfied [4].

The exact computation of  $p(x)$  is a time consuming process, therefore, the authors of [4] provide an approximation function for packet length  $L=120$  (with error within 4% for distances up to  $2R$ ) to enable faster calculation. The approximation function is:

$$p(x) = \left\{ 1 - \frac{\left(\frac{x}{R}\right)^{2\beta}}{2} \right\} \text{ for } x < R$$

$$p(x) = \left\{ \frac{\left(\frac{2R-x}{R}\right)^{2\beta}}{2} \right\} \text{ for } 2R \geq x \geq R$$

$$P(x) = 0 \text{ for } x > 2R.$$

From the above equation it can be noticed that the power attenuation factor is  $2\beta$  rather than  $\beta$ . This is due to the approximation of the packet probability rate rather than the bit probability rate, and the greater impact of packet length on packet reception at larger distances.

The expected hop count of a link on a route is estimated as follows. Suppose that a sender transmits a given packet repeatedly until the receiver acknowledges it. Each received packet is acknowledged  $u$  times [4]. In [4], it is observed that the optimal number of acknowledgements is  $u \approx 1/p(x)$ , which gives the best expected hop count as  $f(u, x) = 2/(p(x)(1-(1-p(x))^u))$ .

Let  $C$  be the node currently holding the message,  $D$  be destination node,  $A$  the considered forwarding neighbor,  $|CD| = c$ ,  $|AD| = a$  and  $|CA| = x$ . In the  $aEPR-u$  routing algorithm [4], node  $C$  currently holding the packet will forward it to a neighbor  $A$  (i.e. a node closer to the destination than itself) that maximizes the ratio of expected progress and cost for the progress made. Since the progress is  $c-a$ , and considered cost (i.e. expected hop count) is  $f(u, x)$ ,  $aEPR-u$  will select the neighbor  $A$  that maximizes  $(c-a)/f(u, x)$ .

### 3 *s-hello* Protocol

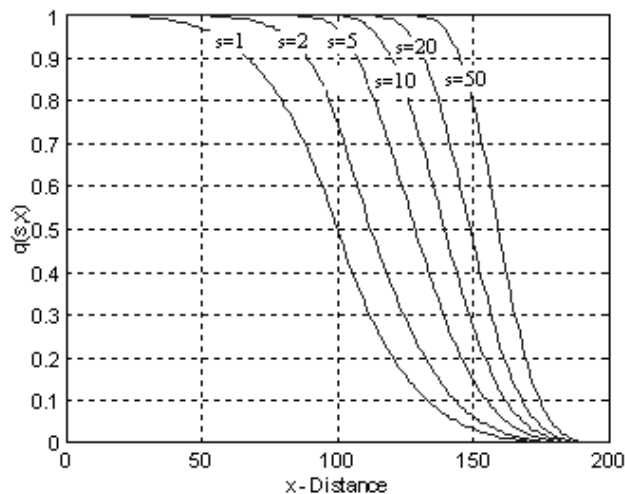
In order to find local knowledge information over a realistic physical layer, two protocols are proposed in this work. The first is the *s-hello protocol*. In this protocol, each node sends a fixed number  $s$  of hello messages to its neighbors.

Given that  $p(x)$  is the probability of packet reception between two nodes at distance  $x$ , the probability that a neighbor at distance  $x$  receives at least one of the  $s$  hello messages, is given by:

$$q(s, x) = 1 - (1 - p(x))^s$$

The *s-unit graph* can be considered as a graph with  $n$  nodes and edge weights  $q(s, x)$ , determined by a distance  $x$  between its neighbors and  $s$ . Theoretically, with  $s$  going to infinity, all nodes will get at least one hello message, but this is not realistic. According to conclusions made in [4], only nodes at a distance up to  $2R$ , where  $R$  is decided by  $p(R) = 0.5$ , are considered. The choice of

threshold  $t$  for considering neighbors at a distance up to  $tR$  is somewhat arbitrary. For example, [4] actually consider neighbors up to a distance of about  $1.4R$ , where the probability of receiving a packet is  $>5\%$ . We enlarge this threshold to study the impact of obtaining a packet with additional retransmissions. Moreover, when  $x > 2R$ , values  $q(s,x)=0$  are obtained in the  $s$ -unit graph. With this assumption, when  $s$  goes to infinity, a unit disk graph with transmission radius  $2R$  is acquired. For finite values of  $s$ , an  $s$ -unit graph (over realistic physical layer) may have directional links between nodes (e.g. node  $A$  hears one hello message from  $B$ , but node  $B$  may not hear any hello message from  $A$ ).



**Fig. 1.** Probability of receiving at least one hello message if the distance between two neighbours is  $x$  and  $R=100$ , for different values of  $s$

Figure 1 shows the probability of receiving at least one hello message for different values of  $s$ , when the distance between two nodes is  $x$ ,  $R=100$ , and  $\beta=2$ . For instance, the probability of receiving at least one hello message at a distance  $1.5R$  when  $s=20$  is about 0.5. This implies that approximately 20 hello messages are required to increase the transmission radius (i.e. the distance where packet reception probability is 0.5) by 50%, from  $R$  to  $1.5R$ . Another observation is that the slope of the curve increases sharply, dropping from a probability near 1 to a probability near 0 on a narrower interval. This can be interpreted as approaching the unit graph model with closer resemblance for larger values of  $s$ . Although, the transmission radius increases as  $s$  goes to infinity.

#### 4 Target Density Protocol

The second protocol proposed to acquire local knowledge information over a realistic physical layer is the *target density protocol*. In this protocol, the goal of each node is to learn about at least

$td$  (target density) neighbors. To achieve this goal, this protocol asks all the nodes in the network to keep sending a hello packet until a target density or a stopping criterion is reached. A stopping criterion could include a timeout or maximum allowed number of sent hello messages. The target density protocol assumes that no prior neighborhood knowledge is available to the nodes. Each node acts as follows:

```

Input: Desired target density  $td$ ; maximal number
      of hello packets  $nmax$ ;  $timeout$ 
Initiate local density to  $ld=0$ ;
Initiate number of sent packets to  $n=0$ ;
Initiate  $time=0$ ;
Repeat
  For each hello packet received from a
    neighbor do  $ld=ld+1$ ;
  Send one hello packet;  $n=n+1$ ;
  Increase accordingly  $time$ ;
Until  $ld \geq td$  or  $n \geq nmax$  or  $time \geq timeout$  .

```

The timeout and max number of hello messages sent are required as the network could be sparse, and therefore, it may not be possible to achieve the desired number of  $td$  neighbors. When using an ideal MAC layer, as in our experiments, a timeout constraint is not required, and simply including a limit on the total number of hello messages sent is sufficient. By applying the target density concept, the *average* number of hello messages sent by the nodes in the network is expected to decrease, when compared to the  $s$ -hello protocol. However, in the case of uniformly distributed networks, nodes located near the network boundary will need to send a larger amount of hello messages to achieve target density, as compared to nodes located inside the network, but the average number of messages is still expected to decrease.

## 5 Simulations of $s$ -hello Protocol

The motivation for testing the  $s$ -hello protocol is to determine what a reasonable choice for  $s$  would be in further applications, such that we decipher whether there is any significant benefit to increasing  $s$ . The experimental work was performed as follows. For each value of  $s$ , every node in the network sends  $s$  hello messages over a realistic physical layer, which generates an  $s$ -unit graph, where neighbors of a node  $A$  are nodes from which a hello message is heard.

**Table 1.** The distribution of neighbors in *s*-hello protocol

<i>x/R</i>	#										
	nodes	<i>s</i> =1	<i>s</i> =2	<i>s</i> =3	<i>s</i> =4	<i>s</i> =5	<i>s</i> =6	<i>s</i> =7	<i>s</i> =8	<i>s</i> =9	<i>s</i> =10
0.13	1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.25	3	2.99	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00
0.38	5	4.95	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
0.50	7	6.78	6.99	7.00	7.00	7.00	7.00	7.00	7.00	7.00	7.00
0.63	9	8.31	8.95	9.00	9.00	9.00	9.00	9.00	9.00	9.00	9.00
0.75	11	9.26	10.72	10.96	10.99	11.00	11.00	11.00	11.00	11.00	11.00
0.88	13	9.19	11.88	12.67	12.90	12.97	12.99	13.00	13.00	13.00	13.00
1.00	15	7.50	11.25	13.13	14.06	14.53	14.77	14.88	14.94	14.97	14.99
1.13	17	4.98	8.50	10.99	12.75	14.00	14.88	15.50	15.94	16.25	16.47
1.25	19	3.01	5.54	7.67	9.46	10.97	12.24	13.31	14.21	14.97	15.61
1.38	21	1.60	3.08	4.45	5.71	6.88	7.96	8.95	9.87	10.72	11.50
1.50	23	0.72	1.42	2.09	2.74	3.38	3.99	4.58	5.16	5.72	6.26
1.63	25	0.25	0.49	0.73	0.97	1.21	1.45	1.68	1.91	2.14	2.36
1.75	27	0.05	0.11	0.16	0.21	0.26	0.31	0.37	0.42	0.47	0.52
1.88	29	0.00	0.01	0.01	0.01	0.02	0.02	0.02	0.03	0.03	0.04
2.00	31	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
total	256	60.60	77.94	87.85	94.83	100.22	104.60	108.30	111.48	114.27	116.74
4%	4	0.95	1.22	1.37	1.48	1.57	1.63	1.69	1.74	1.79	1.82

Figure 1 already provides some basic properties of hello protocols, as it can be used to derive the average number and distance distribution of neighbors who received one of the *s* hello packets sent. Table 1 depicts our analysis. Neighbors are located at variable distances *x/R*, where *R* is the reference point ( $p(R)=0.5$ ). The total number of neighbors considered in this experiment is 256 (all up to distance  $2R$ ). Distances are normalized with respect to *R*, and are shown in the first column of Table 1. The second column shows the expected number of neighbors at each respective distance. It should be noted that column two formally represents the number of expected neighbors in a ring between two distances, but has been assumed as exact distances for ease of calculation. However, this assumption has no significant impact on the accuracy of the presented data. Therefore, there is 1 neighbor at distance 0.125, 3 neighbors at distance .25, 5 neighbors at distance .375 and etc. Moreover, the total number of neighbors at relative distance  $\leq 0.125t$  is  $t^2$  (located at a total of 16 rings). The remaining columns depict the expected number of neighbors that received at least one hello message after running the *s*-hello protocol for  $1 \leq s \leq 10$ . The second to last row shows the cumulative number of neighbors, while the last row shows the ratio of existing neighbors that learned about the node, multiplied by 4. We have restricted our density references to a distance of *R*, only considering ‘good’ neighbors to mimic a kind of ‘real’ density, where good neighbors are those that are most likely to receive a given message. Out of 256 considered

neighbors, a quarter of them are ‘good’ neighbors, which is why ratios are multiplied by 4. Furthermore, Table 1 shows that when  $s$  is one, approximately 60 neighbors receive a hello message, or close to a quarter of neighbors. Therefore, almost all ‘good’ neighbors have received a message and many others have not. As  $s$  increases, the ratio of neighbors that learn about their neighbors increases significantly, but is accompanied by decreasing amounts of gain for each increase in  $s$ . Table 1 also shows that increasing  $s$  much past 5 brings little benefit and a tendency for performance to approach that of the unit graph model.

The next simulation was performed over static networks with random node placements, assuming no collisions with other traffic. The simulation area for the placement of nodes was 300x300 (size has no impact on results). Each node was selected uniformly at random over the simulation area with equal power for transmission. The number of nodes  $n$  was tested for  $n=150, 200$  and  $250$ . The average node degree  $d$  is defined as the number of neighbors up to distance  $R$  from a given node, where  $R$  is the distance for which packet reception probability is  $p(R)=0.5$ . As mentioned in the introduction, nodes up to a distance  $2R$  may still potentially receive messages from a given node, implying that the average density, considering all neighbors up to distance  $2R$ , is then  $4d$ . The protocol was tested for  $d=6, 8, 10, 16, 20, 24, 32, 40$  and  $s=1, 2, 3, 4, 5, 10, 20, 50$ . The path loss exponent  $\beta$  was set to  $\beta=2$ .

We have verified the performance of s-hello protocol by running the *aEPR-u* routing protocol [4], using actual (discovered) neighbors as the only candidates for forwarding. The average expected hop count and the routing success rate, for each combination of parameters  $n$  and  $d$ , was measured over 30 graphs (connected graphs only) and 100 random source-destination pairs of nodes. We have tested several other routing schemes from [4], but results were very similar. The performance measures were expected hop count (EHC), measuring the expected number of transmissions, retransmissions, and acknowledgments (assuming that acknowledgements are of the same packet size as the transmitted packets), and success rates of routing tasks. Our primary concern was the impact of value  $s$  on the performance, and not the comparison between various routing methods, since they use different assumptions on neighbor knowledge (in [4], the assumption is close, but not the same, to assuming  $s=20$ ).

**Table 2.** Average EHC and routing success rate for  $d=40, n=250$  and different  $s$  values

	<b>d = 40</b>	<b>n = 250</b>
	<b>EHC</b>	<b>Success</b>
<b>s = 50</b>	8.175	0.997
<b>s = 20</b>	8.285	0.996
<b>s = 10</b>	8.232	0.994
<b>s = 5</b>	8.201	0.998
<b>s = 4</b>	8.246	0.993
<b>s = 3</b>	8.215	0.996
<b>s = 2</b>	8.178	0.996
<b>s = 1</b>	8.243	0.995

Table 2 shows that for large values of node average degree ( $d=40$ ), and a given network size ( $n=250$ ), increasing the number of hello messages  $s$  does not make a significant difference, neither

for the EHC nor the success rate, since both values remain almost the same for any value of  $s$ . Therefore, for very dense networks, the *1-hello* protocol is sufficiently effective.

**Table 3.** Average expected hop count and routing success rate for  $d=6$ ,  $n=250$  and different  $s$  values

	<b>d = 6</b>	<b>n = 250</b>
	<b>EHC</b>	<b>Success</b>
<b>s = 50</b>	42.365	0.982
<b>s = 20</b>	39.646	0.972
<b>s = 10</b>	42.136	0.940
<b>s = 5</b>	47.350	0.903
<b>s = 4</b>	39.714	0.890
<b>s = 3</b>	36.070	0.837
<b>s = 2</b>	32.857	0.769
<b>s = 1</b>	31.685	0.546

However, Table 3 shows that for small values of node average degree ( $d=6$ ), and a given network size ( $n=250$ ), increasing the number of hello messages clearly increases the routing success rate. The EHC increases as well because successful routing over longer routes is achieved. The success rate achieves 90% for  $s=5$ . Increasing  $s$  beyond 5 is not beneficial because no significant improvements, in terms of success rates, are achieved.

**Table 4.** Routing success rate for  $s=5$  and different values of  $n$  and  $d$

	<b>s = 5</b>	
	<b>n = 150</b>	<b>n = 250</b>
<b>d = 6</b>	0.928	0.920
<b>d = 8</b>	0.978	0.976
<b>d = 10</b>	0.986	0.988
<b>d = 16</b>	0.993	0.996
<b>d = 20</b>	0.992	0.995
<b>d = 24</b>	0.994	0.997
<b>d = 32</b>	0.994	0.994
<b>d = 40</b>	0.994	0.994

Table 4 shows that if  $s=5$  is chosen, regardless of the network size and the node average degree (for tested values), the routing success rate is over 92%. For node average degrees larger than  $d=7$ , the success rate is over 97%.

## 6 Simulations of the Target Density Protocol

Our main goal while investigating the target density protocol was to determine the average number of hello messages required to achieve the target density  $td$ , which is chosen according to the desired routing success rate, and to determine how the performance of the *target density protocol* compares against that of the *s-hello* protocol. The experimental work was performed as follows. The target density was selected to be between  $d$  and  $4*d$  (which is the maximum achievable density when considering neighbors up to distance  $2R$ ). The target density protocol was then applied

over a realistic physical layer, by all nodes in the network. Within the simulation, a randomly selected number is compared against the reception probability  $p(x)$  in order to determine whether or not the message was received. As in the previous section, neighbors of any node  $A$  are nodes from which at least one hello message was heard.

Simulation work was performed over static networks only, with no collisions with other traffic, as in the s-hello protocol simulation. Initially, nodes are not aware of any neighbors. The protocol was tested for  $n=250$  nodes, and for average node degrees (numbers of neighbors up to distance  $R$ )  $d=6, 8, 10, 12, 16$  and  $20$ . The target density  $td$  was tested for  $td = d, d+2, d+4, \dots, (2*d)+4$ . The maximum number  $m$  of hello messages was used as the stopping criterion, and the values used were  $m=1, 2, 3, 4, 5, 10, 20, 50$ . We measured the average number  $AvgHellos$  of hello messages sent to achieve the target density  $td$ , and the actual average network density  $AvgDen$  achieved, for each combination of parameters  $n, d, td$  and  $m$ . Each experiment was carried over 30 graphs (connected graphs only), and the path loss exponent  $\beta$  was set to  $\beta=2$ .

**Table 5.** Achieved average density ( $AvgDen$ ) after sending one hello message for  $n=250$  and different node degrees  $d$

	$n = 250$	
	$AvgDen$	$AvgHellos$
$d = 6$	6.340	1.000
$d = 8$	8.430	1.000
$d = 10$	10.510	1.000
$d = 12$	12.630	1.000
$d = 16$	16.763	1.000
$d = 20$	20.942	1.000



Table 5 shows the actual network densities  $AvgDen$  achieved after each node sends exactly one hello message. The experimental data shows the  $AvgDen$  as being slightly above the average network density  $d$ . This implies that after sending one hello message, the average number of neighbors learned is slightly above the number of existing neighbors up to distance  $R$ ,  $p(R)=0.5$ . Note that this conclusion differs from the one in Table 1, but not by much. This can be accounted for by the fact that all neighbours are actually closer to the sending node than was assumed in Table 1.

**Table 6.** Hello messages needed to achieve target density  $2d$ , when  $n=250$

$d$	$td$	$AvgDen$	$AvgHellos$
6	12	12.87	29.07
8	16	16.88	29.16
10	20	20.71	30.00
12	24	24.53	30.09
16	32	32.71	30.72
20	40	40.71	30.92

Table 6 shows the results of our study of achieving an actual density of  $2d$ , corresponding roughly to all neighbors up to a distance  $\sqrt{2} R$ . It can be noticed that in such a case, the average

number of hello messages per node is very high, about 30. Therefore, it is difficult to learn about almost all neighbors at distances between  $R$  and  $1.41R$ . Interestingly, the required number of hello messages almost did not depend on the initial network density. This is due to the proportionality of the problem studied, such that both the network density and desired actual density are expressed as multiples of the same number,  $d$ .

**Table 7.** Maximum amount of hello messages  $m$  required for achieving the chosen target density  $td$  for  $n=250$  and  $d=12$

td	n=250	d=12	
	m	AvgDen	AvgHellos
12	2	12.07	1.37
16	5	15.91	3.20
20	20	20.16	11.42
24	50	23.53	30.09

Table 7 shows the results of achieving actual densities  $d$ ,  $4/3d$ ,  $5/3d$  and  $2d$ , reporting average numbers of hello messages to be approximately 1, 3, 11 and 30, respectively. It also reflects that the stopping number  $m$  of hello messages should change appropriately.

Table 8 shows that if  $m=5$  and  $td=2d$  then a network density slightly larger than  $1.5d$  is achieved, which gives a reference point for choosing target density and timeout parameters.

**Table 8.** Achieved network densities for  $td=2d$  and  $m=5$

d	td	m	AvgDen	AvgHellos
6	12	5	9.468	4.413
8	16	5	12.619	4.483
10	20	5	15.881	4.517
12	24	5	18.991	4.580
16	32	5	25.370	4.622
20	40	5	31.572	4.677

Furthermore, we have verified the performance of the target density protocol by running the *aEPR-u* routing protocol [4], as was done for the *s-hello* protocol. In addition to the measures already discussed in this section, we included the average expected hop count and the routing success rate, for each combination of parameters  $n$ ,  $d$ ,  $td$ , and  $m$ , measured over 30 graphs (connected graphs only) and 100 random source-destination pairs of nodes. Our primary concern was to determine how the performance of the target density protocol compares against that of the *s-hello* protocol. Particularly, our aim was to determine the number of hello messages that could be saved using the target density protocol, while still maintaining the desired success rate achieved by the *s-hello* protocol.

In section 5, the results of the simulations utilizing the *s-hello* protocol on very dense networks ( $d=40$ ,  $n=250$ ), showed that the *l-hello* protocol is sufficiently effective, providing routing success rates above 99%. Since in this case all nodes must send exactly one hello message, it is clear that the target density protocol has little or no affect on very dense networks. However, quite the opposite is true in less dense networks. For example, the *s-hello* protocol simulations showed that in-

creasing  $s$  beyond 5 is not beneficial in terms of improving the success rate for small values of node average degree ( $d=6$ ) and a given networks size ( $n=250$ ). With this knowledge in mind, we tested the target-density protocol by setting the maximum number of hello messages  $m$  to 5, and observing how various target densities affect the expected hop count and success rate.

**Table 9.** Average EHC and routing success rate for  $d=6$ ,  $n=250$ ,  $m=5$  and different  $td$  values

	$m = 5$		$d = 6$	$n = 250$
	AvgDen	AvgHello	EHC	Success
<b>td = 18</b>	10.056	4.98	37.170	0.900
<b>td = 16</b>	9.945	4.93	55.783	0.889
<b>td = 14</b>	9.862	4.754	35.700	0.913
<b>td = 12</b>	9.513	4.389	34.751	0.904
<b>td = 10</b>	8.816	3.835	37.489	0.846
<b>td = 8</b>	7.859	3.184	40.119	0.794
<b>td = 6</b>	6.525	2.452	39.933	0.622
<b>td = 4</b>	4.895	1.756	55.842	0.284

Table 9 shows that the target density protocol is able to achieve an expected hop count of 35.7 with a success rate of 0.913 utilizing an average of 4.754 hello messages per node and a target density of 14. This is an average savings of 0.246 hello messages per node and a total savings of approximately 62 hello messages, when compared against the performance of the  $s$ -hello protocol (i.e.  $s=5$ ,  $EHC=47.35$ ,  $Success=0.903$ ).

Table 10 shows the effect of varying the maximum number of messages  $m$  while fixing the target density ( $td=14$ ). Although increasing the maximum number of hello messages while holding the target density allows more nodes to reach target density, it has the same effect seen with the  $s$ -hello protocol, such that increasing  $m$  much beyond 5 is not beneficial since no significant improvement in success rate is achieved.

**Table 10.** Average EHC and routing success rate for  $td=14$ ,  $d=6$ ,  $n=250$  and different  $m$  values

	$td = 14$		$d = 6$	$n = 250$
	AvgDen	AvgHello	EHC	Success
<b>m = 50</b>	12.819	34.266	42.658	0.980
<b>m = 20</b>	11.928	15.973	40.780	0.970
<b>m = 10</b>	10.986	8.831	40.300	0.959
<b>m = 5</b>	9.804	4.747	39.200	0.896
<b>m = 4</b>	9.459	3.858	36.348	0.890
<b>m = 3</b>	8.818	2.934	34.434	0.852
<b>m = 2</b>	8.003	1.982	35.937	0.760
<b>m = 1</b>	6.330	0.999	33.891	0.576

Further simulations show that the true performance gains of the target density protocol occur when attempting to attain very high success rates in low density networks. For example, in order to achieve a success rate of 99% for small values of node average degree ( $d=6$ ) and a given networks size ( $n=250$ ), the  $s$ -hello protocol requires approximately 125 hello messages per node

(Table 11). However, as Table 12 shows, the target density protocol can achieve a success rate of 99% using only an average of 86.078 hello messages per node. This is an average savings of 38.922 hello messages per node and a total savings of approximately 9,731 hello messages.

**Table 11.** Average EHC and routing success rate for  $d=6$ ,  $n=250$ , and different  $s$  values (*s-hello protocol*)

	<b>d = 6</b>	<b>n = 250</b>
	<b>EHC</b>	<b>Success</b>
<b>s = 200</b>	44.155	0.994
<b>s = 175</b>	44.883	0.995
<b>s = 150</b>	48.718	0.987
<b>s = 125</b>	43.679	0.990

**Table 12.** Average EHC and routing success rate for  $d=6$ ,  $n=250$ ,  $m=125$  and different  $td$  values

		<b>m = 125</b>	<b>d = 6</b>	<b>n = 250</b>
	<b>AvgDen</b>	<b>AvgHello</b>	<b>EHC</b>	<b>Success</b>
<b>td = 18</b>	15.028	100.334	45.296	0.988
<b>td = 16</b>	14.502	86.078	44.336	0.990
<b>td = 14</b>	13.543	73.331	45.598	0.987
<b>td = 12</b>	12.408	59.258	52.892	0.980
<b>td = 10</b>	11.095	47.361	46.110	0.963
<b>td = 8</b>	9.524	36.460	55.942	0.903
<b>td = 6</b>	7.745	27.078	79.305	0.718
<b>td = 4</b>	5.784	18.644	102.845	0.392

The main advantage of the target density protocol is in saving unnecessary hello messages after a desired number of neighbors (e.g. for good performance of a routing scheme) is learned, while the s-hello protocol blindly continues sending messages possibly with too many or too little of them at end.

## 7 Discussion

The previous sections defined, and presented simulation results of, two protocols (i.e. s-hello protocol and target density protocol) designed to address the problem of gathering neighbor information when a realistic physical layer is applied. Since, when using a realistic physical layer, the reception of a hello message from a neighbor is no longer guaranteed, but probability-based, the s-hello protocol increases the likelihood of neighbors successfully receiving hello messages by having each node send  $s$  hello messages. Results showed that such a scheme does positively impact routing success rates by improving the amount of information that can be gathered in the network. However, results also showed that the s-hello protocol is better suited for less dense networks. This can be attributed to the fact that the probability of two neighbors successfully exchanging a hello message decreases as their distance from each other increases. In dense networks, the average distance between neighbors is minimal, and therefore, hello messages are successfully exchanged with a higher probability. Moreover, gathering information in dense networks can be

satisfactorily achieved using an  $s$  value of 1. Furthermore, results indicated that an optimal  $s$  value exists, such that increasing  $s$  past this optimal value does not significantly improve routing success rates. Although simulation results confirmed that the s-hello protocol is able to improve the gathering of information when using a realistic physical layer, the algorithm itself is less than ideal, reason being that similar routing success rates can be achieved using fewer hello messages. The s-hello protocol wastes hello messages by having all nodes send an equal number of hello messages. A node will keep sending hello messages even if its neighbors have successfully received the hello messages. In order to improve this scheme, the target density protocol was presented. The target density protocol attempts to save wasted hello messages by having each node keep track of the number of hello messages it receives from neighboring nodes. As soon as a specified target density  $td$  is reached, the node will stop sending hello messages, such that it only sends the ideal amount of hello messages required to reach the target density. Although this requires additional effort by the nodes of the network, simulation results showed that many of the hello messages wasted by the s-hello protocol can be saved using the target density protocol, especially when trying to achieve high routing success rates in sparse networks. Results also indicated that similar to the  $s$  value in the s-hello protocol, there exists an optimal  $td$  value, such that increasing  $td$  past this optimal value does not significantly improve routing success rates.

## 8 Conclusions

Accurate local knowledge information is extremely important for localized algorithms that make decisions based on neighborhood information. When a realistic physical layer is applied, hello message protocols become non-trivial. Therefore, robust protocols for collecting neighborhood information need to be designed. In this work, two first such protocols are proposed. These protocols are intended for scenarios where nodes are initialized (e.g. sensor networks starts to operate) or are mobile (entering new neighborhoods frequently). In scenarios where the network is relatively stable or slow moving, periodic hello messages, or the 1-hello protocol, can be applied successfully, since missing a hello message once in a while is not detrimental in such situations.

The ideas described in this paper may be further applied in the related problem of route discovery in reactive routing schemes. The common problem in all of the existing attempts [2, 1, 11] is that each node still retransmits the received packet (at most) once. While blind flooding is an acceptable protocol in the unit disk graph model, guaranteeing connectivity and optimality, it may not lead to an optimal or even any solution under the given realistic physical layer. A single transmission may not reach a particular neighbor, which then could disconnect the destination from the source, or fail to offer a good existing route. A link that fails in only one attempt between two nodes (during route discovery process) may in fact be part of a good link and contribute toward an optimal route under the considered metric.

In [12, 13], we proposed a modification to the route discovery process, as follows. A single packet retransmission by a node considered for inclusion, may not be received by relatively distant

neighbors that may be good choices for final route construction. Therefore, each node may retransmit the given route discovery packet *several times* rather than once as in the unit disk graph model. Such multiple retransmissions may also serve to measure or re-evaluate the packet reception probability. What is the optimal number of retransmissions? A simple option is to retransmit a fixed number of times. In dense neighborhoods, a few retransmissions may suffice. Better trade-offs between messages sent and gains made could be achieved by a node retransmitting until a certain number of packets with the same content have been received, before or after the first retransmission (subject to a timeout) [12, 13]. Nodes may also overhear further packets for the same route requests, and make decisions to stop based on satisfactory progress of the route request. If a received message contains information about a better route, then the retransmission of the previously known best route (if any is ongoing) stops and the retransmission of the new route starts with a fresh counter. One particular issue is the existence of long ‘bridge’ edges between two sub-networks. The link may not be discovered unless a maximal number of attempts are made. To increase the chance of discovering such bridges, nodes that are not able to extend a currently offered partial path will continue retransmitting until the maximum allowable number of packets has been sent. These general descriptions may be custom completed toward particular protocols in a variety of ways. The optimal number of retransmissions, and therefore, the cost of applying this concept, primarily depends on the network density. In very dense networks, a single retransmission of a route request may find a sufficient number of neighbors to provide a near best route. However, the more sparse the network, the more retransmissions might be needed, especially for discovering important bridges in the network. Network traffic characteristics (e.g. congestion) also impact the cost needed for finding near optimal routes.

The metric used during the route discovery process depends on the network assumptions. For instance, if a route with hop by hop acknowledgements is searched for, the appropriate metric is the expected hop count (EHC) on the route, being the sum of expected hop counts on each hop. The calculation of EHC further depends on medium access assumptions. In [4], a message is divided into packets of fixed size, and each packet is then independently routed. Acknowledgements, if sent, are of the same size as the packet, and are counted as part of the expected hop count. EHC then depends on the expected number of retransmissions and the expected number of acknowledgements.

The cost of a link on a route is estimated as follows. In [13, 4], it is observed that the optimal number of acknowledgements is  $u \approx 1/p(x)$ , which gives the best EHC  $2/(p(x)(1-(1-p(x))^u))$ . This cost can be included in the transmitted packets for addition to the considered route cost. Note that [4] actually derives the ideal hop count (if additional nodes can be placed in ideal positions) and uses it in timeout decisions for retransmissions. This allows better routes to be advertised faster than others, and consequently reduces overall traffic, since nodes that overhear neighbors already announcing themselves as better ‘forwarders’ may cancel their own retransmissions.

We hope that this research direction will attract more attention, and that more aspects of a realistic physical layer will be considered. This includes variable packet sizes [11], variable bit rates, applying channel modulation and error correction schemes instead of assuming independent bit

reception, and etc.

## Acknowledgements

This research is supported by NSERC Collaborative Research Grant and UK Royal Society Research Merit Award.

## References



1. L. Qin, T. Kunz: On-demand Routing in MANETs: The Impact of a Realistic Physical Layer Model. *Proceedings of the Second International Conference ADHOC-NOW* (2003) 37-48.
2. D. De Couto, D. Aguayo, J. Bicket, R. Morris: A High-throughput Path Metric for Multi-hop Wireless Routing. *Proceedings of ACM Mobicom*, San Diego. (2003)
3. I. Stojmenovic: Position Based Routing in Ad Hoc Networks. *IEEE Communications Magazine* 40(7) (2002) 128-134.
4. J. Kuruvila, A. Nayak, I. Stojmenovic: Hop Count Optimal Position Based Packet Routing Algorithms in Ad Hoc Wireless Networks with a Realistic Physical Layer. *IEEE Journal of Selected Areas in Communications* 23(6) (2005) 1267-1275.
5. S. Banerjee, A. Misra: Energy Efficient Reliable Communication for Multi-hop Wireless Networks. *Wireless Networks (WINET)*, to appear.
6. M. Takai, R. Bagrodia, K. Tang, M. Gerla: Efficient Wireless Network Simulations with Detailed Propagation Models. *Wireless Networks* 7(3) (2001) 297-305.
7. D. Niculescu: Positioning in Ad Hoc Sensor Networks. *IEEE Networks* 18(4) (2004) 24-29.
8. M. Haenggi: Wireless Sensor Networks with Rayleigh Fading Channels. *NSF/ECS Workshop*, University of Notre Dame, Indiana, September. (2003)
9. G. G. Finn: Routing and Addressing in Large Metropolitan-scale Internetworks. *ISI Research Report*, ISU/RR-87-180, March. (1987)
10. P. Bose, P. Morin, I. Stojmenovic, J. Urrutia: Routing with Guaranteed Delivery in Ad Hoc Wireless Networks. *ACM DIAL M Workshop*, Seattle, August. (1999); *ACM Wireless Networks* 7(6) (2001) 609-616.
11. T. Nadeem, A. Agrawala: IEEE 802.11 Fragmentation-aware Energy-efficient Ad-hoc Routing Protocols. *Proc. 1<sup>st</sup> IEEE Int. Conf. on Mobile Ad-hoc and Sensor Systems (MASS)*, Fort Lauderdale, October. (2004)
12. Ivan Stojmenovic, Amiya Nayak and Johnson Kuruvila: Design Guidelines for Routing Protocols in Ad Hoc and Sensor Networks with a Realistic Physical Layer. *IEEE Communications Magazine (Ad Hoc and Sensor Networks Series)* 43(3) (2005) 101-106.
13. I. Stojmenovic, A. Nayak, J. Kuruvila, F. Ovalle-Martinez, E. Villanueva-Pena: Physical Layer Impact on the Design and Performance of Routing and Broadcasting Protocols in Ad Hoc and Sensor Networks. *Computer Communications* 28(10) (2005) 1138-1151.