

Little Tom Thumb Went Straight Home

Asymptotic Behavior of a Routing Protocol in Ad Hoc Networks with a Mobile Access Point

Cédric Westphal

Networking Laboratory, Nokia Research Center,
Mountain View, California.

E-mail: cedric.westphal@nokia.com

Abstract—The task of routing streaming data in a sensor network is made arduous by the resource constraints imposed on each node. The control overhead of a routing protocol has to be minimized in order to preserve limited resources at the node. Furthermore, it is widely expected that the gateway of the sensor network will not be static in many application scenarios, but will be moving around the network. The gateway could move to collect data, or because it is not associated with a specific location, but with a specific mobile user.

We study a simple protocol and consider the cost it imposes on the networks in terms of number of messages sent, or equivalently, bandwidth or power usage. We study the scaling behavior of the so-called bread crumbs (BC) protocol, and show that it is optimal in its scaling behavior in one- and two-dimensional graphs. The BC protocol is thus named as the mobile gateway leaves a trail in the network as it moves about, as Little Tom Thumb, or Hansel and Gretel do in the forest, in the well-known fairy tales. The BC protocol routes packets along the path followed by the mobile sink, taking short cuts whenever possible. We compare this protocol with the cost of routing on the shortest path from the sensor node to the mobile sink, and with the cost of flooding the network in the hope of finding the sink.

We show that the path attained with the BC protocol is asymptotically optimal, and scales as the shortest path. We support the analysis with some simulations and also consider a one step diffusion extension of the bread crumbs protocol as well.

I. INTRODUCTION

Sensor networks are becoming ubiquitous. Applications range from the military to infrastructure and wildlife monitoring to security to the support of the supply chain management. Commercial consortia such as Zigbee [1], or even Bluetooth [2], have been created to assist with the building of the sensor market and promote the use of the technology. As sensor networks become more and more available, network management issues becomes more and more important.

A key issue is how to extract information from the sensor network. Typical architectures include a gateway, or sink, and nodes forward their information streams to this gateway. In many applications, the sink has a fixed position. But in other applications, such as perimeter monitoring for physical intrusion detection, or for home automation, the sink should be mobile. In the perimeter monitoring case, the gateway should be the security guard doing his rounds. In a home automation scenario, the sink should be a universal remote control carried by the user living in the home. As sensor networks become widely deployed, the mobile phone, fitted with the proper short range antenna, becomes a central part of the network management: many applications will require the data they generate to be streamed to the hand-held device of the user, as illustrated in Figure 1.

Routing the packets created by the motes to the mobile sink creates new challenges. For instance, giving the motes location information about the mobile sink so that it knows at all time where the sink is, would introduce some unnecessary overhead. On the other hand, if the sensor node has no idea as to where the sink is, it will initiate a flooding of the network, causing unnecessary expenditure of bandwidth and, most importantly, power.

The same routing challenge exists in other networks where packets have to go through a fixed infrastructure which blankets a coverage area in order to find a mobile node. Similarly to a mobile sink in a sensor network, the mobile node never leaves the coverage area created by a set of nodes, in this case static access points. A large-scale metropolitan wireless mesh networks (see [3]–[6]) for instance faces the same trade-off: updating the location of a node at any given time finds the optimal path in hop count, or in number of packet retransmissions; yet it adds a management overhead by sending control packets in the network. The more control packets, the more accurate the routing protocol performance. Since such multi-hop metropolitan mesh networks see a strong

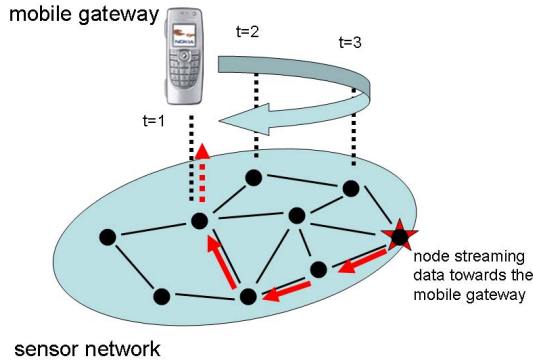


Fig. 1. Routing Data Streams in a Sensor Network with Mobile Access

performance drop-off after a few hops away from a wired internet gateway, the control packets may strongly affect the overall performance in the network and offset the gain of a *shorter* routing path.

In this paper, we study a well known and very basic protocol, the so-called Bread Crumbs protocol, to route data over a fixed infrastructure to a mobile node. We place ourselves in the context of a mobile sink in a sensor network, but the reader will keep in mind the wider application field of the results. We compare the cost imposed on the network by the protocol as it affects the power usage and the delay, and compare it with flooding the network and with the optimal shortest path.

The contribution of the paper is as follows: we show that in single dimension, the scaling behavior is linear with respect to the number of nodes. It thus scales as the shortest path. We compute the proportionality ratios between the BC protocol path and the shortest path.

Further, the main result of the paper is that in two dimensions, the Bread Crumbs (BC) routing still scales proportionally to the optimal path, namely it is proportional to the square root of the number of nodes, while flooding is linear with the number of nodes. We prove this result both in a lattice topology, and in a random geometric graph topology.

This means that even a protocol as simple as the BC protocol, which requires no overhead at all, provides a huge improvement in terms of economizing the sensor network's resources and is asymptotically optimal as the sensor network grows larger!

In the next two sections, we describe the model, the routing protocol and review some of the literature in the domain of wireless sensor networks with a mobile access

point. In section IV, we formally study the asymptotic scaling behavior from the protocol compared to the shortest path. We then define a diffusion extension of the protocol in Section V. In section VI, we provide some numerical results for both the BC protocol and its diffusion extension. Section VII offers concluding remarks.

II. MODEL DESCRIPTION

We consider a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with a set \mathcal{V} of vertices and \mathcal{E} a set of edges connecting them. Each vertex corresponds to a sensor node. The mobile sink roams within the sensor network, and attaches to the sensor node closest to the sink. In our model, we consider that the sink is jumping along the edges of the graph, and is always situated at one of the vertices.

This restricts the movement of the mobile sink. We are imposing the following assumption: the mobile sink cannot attach to two sensors successively if there is no edge connecting the corresponding vertices.

This is not a constraining assumption, as most sensor networks should be dense enough so that there is no hole in the covered area for the sink to cross. In many applications, where it is critical that the information be received at the sink as fast as possible, the mobile gateway should never disconnect from the network anyway. Further, we will discuss later how to relax this assumption, as the results still hold.

We model the movement of the sink as a stationary random walk on the graph. We assume a slotted time t , and define by $S(t) = v$ the position of the sink at time $t \in \mathbf{Z}$.

$$P(S(t+1) = u | S(t) = v) = \begin{cases} q(u, v) & \text{if } u \leftrightarrow v \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The unit of time here is based on the movement of the mobile sink. t means the t -th movement of the sink, and it maps to an epoch in a time axis measured in minutes and seconds. With no loss of generality, we can assume the two time measures coincide, as will become clear when we detail the routing protocol. If the time is not uniform, namely if the sink attaches to one sensor longer than it attaches to the next, this does not impact the *length of the routing path* to the sink, nor the accuracy of the results presented here.

There are many graph topologies, and many ways to pick the next vertex for the random walk. An interesting graph to study is the 2-dimensional torus, associated with a random walk which moves at each time slot with the same probability to each neighbor. A nice property of

this scenario is that it converges to a uniform distribution: the mobile node has the same likelihood to be at any point in the graph.

We assume that an event is detected at one of the vertices, and a notification should be sent to the mobile sink. This is an event-triggered routing problem from the nodes to the sink. The main issue is how to send the alert packet to the sink efficiently. A broadcast packet would eventually find the sink, but at the expense of flooding the network. A gossip-like [12] probabilistic forwarding to an estimation of the sink’s random position would find it only within some probability bounds.

The issue of sensor networks with mobile access (SENMA) was described in [7], [8] in a more information theoretic context. [7] introduced the acronym SENMA. [10] or [9] also consider a mobile sink, but the mobility of the sink is introduced as a way to alleviate the inhomogeneous power consumption when data is funneled towards a static sink. [11] defines MobiRoute, a protocol to support sink mobility in a sensor network in order to reduce power consumption. Our work considers a mobility model for the sink which is user or application driven, as in [7], [8].

III. FINDING THE MOBILE SINK: THE BREAD CRUMBS PROTOCOL

In this document, to route a packet to the sink, the nodes use a steepest descent algorithm. The gradient is created as follows: each time the mobile sink last attaches to the wireless sensor, the sensor starts a timer. This timer reflects the age of the last visit. The mobile sink leaves a mark at each stop along its path. The age of the visit is basically the white pebbles or bread crumbs that our mobile sink leaves behind like in the *Little Tom Thumb* [13] or the *Hansel & Gretel* [14] fairy tales. For this reason the protocol is usually called the Bread Crumbs (BC) protocol, thus the title of our paper.

The gradient created by the protocol is plotted on Figure 2 (note that for clarity of the illustration, we inverted the age axis, and the mobile sink is located at the highest point in the figure).

The bread crumbs protocol is a simple and relatively obvious choice for a routing protocol and is not novel. What we are interested in this paper -and what is our main contribution- is to assess and quantify the performance of the protocol. A version of the protocol was used as an application example in [15], and the protocol can be considered as a version of last encounter routing protocols [16], [17] where only one node is mobile.

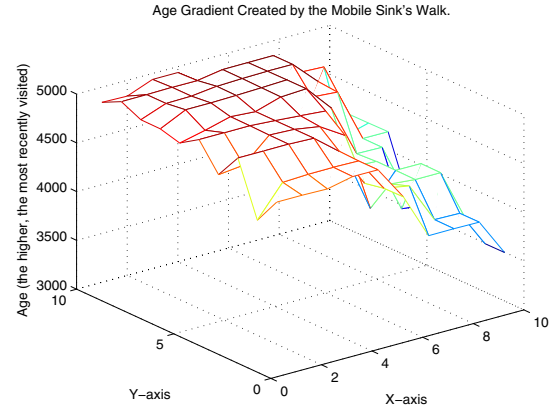


Fig. 2. Gradient Created by the Mobile Sink

Note that while the asymptotic results of this paper are similar to those in [16], namely that age routing approximates the shortest path, there are very significant differences. For once, [16] assumes –and makes use of– a uniform and independent movement of the nodes. In our setting, there is a *strong dependence* in the movement of the nodes if the reference axes are centered on the mobile point, and there is *no uniformity* if the referential is static, as one node is mobile and the other static. The framework here is fundamentally asymmetric. Furthermore, we will see that the tools and techniques utilized to prove the results are very different as well.

When a sensor receives an alert, it forwards it to its neighbor which has the lowest age, ie. the neighboring node last visited by the mobile sink.

The forwarding can take two forms: either a node broadcasts the packet, including its age and only the one neighbor with a more recent visit of the sink forwards it. In this scenario, packets the sender might not be aware that its one receiver successfully received the packet, so it is not the preferred solution. The other form requires knowledge of the age of the neighbors, for instance as a parameter included in a hello handshake¹. The node thus unicasts the packet to its intended next hop neighbor. Since the age delta between two vertices stays constant until the next sink visit, there is no need to periodically broadcast one’s age, only to update it after another visit of the sink.

There are a few nice properties of this bread crumbs protocol:

- it is very simple, with no overhead added to locate the mobile node,

¹Knowing the age of the neighbors is particularly useful when the sink can attach to several nodes at the same time

- it is fully distributed and lightweight,
- making the obvious assumption that the physical speed of the moving sink is much slower than the forwarding speed of the packets, each packet will eventually reach the sink,
- since the sensor nodes are not moving, there are no loops or packets going indefinitely around the network: this routing is loop-free.

Also, the packet follows a line, whereas flooding covers a surface, so it can be expected to be more efficient in finding the mobile sink. On the other hand, because it follows a line, and is not forwarded multiple times as a broadcast packet, it might take a longer time to reach its destination.

We attempt to quantify these two performance metrics: how long it takes the event to reach the mobile sink, and how many total messages are being sent to reach the sink, in the following section. We denote by c_{bc} , c_f , c^* , d_{bc} , d_f and d^* respectively, the cost -in number of messages sent- for the Bread Crumbs protocol, for the full flooding, for the optimal cost, the delay -ie. the length of time for an alert to reach the mobile sink- for the Bread Crumbs protocol, for the full flooding and the optimal delay, respectively. The optimal cost or delay might not always be available for computation.

Alerts are events that occur randomly in the graph. The distribution of the node originating the event could be chosen in many different ways. For instance, one could distribute the triggering events only at the periphery of the graph, as in the perimeter monitoring scenario. However, in the remainder of the document, it is taken to be uniform over all vertices.

IV. ANALYSIS OF THE SCALING BEHAVIOR

We first state some simpler results to build some intuition. Note that in one-dimension, the assumption that there is no hole in the coverage of the sensor nodes is natural, as it implies that the network is connected, a requisite property to reach the mobile sink.

A. Linear topology

Cycle: We consider a 1-dimensional graph with N nodes, such that nodes i and $i+1$ are connected, as well as node N and 1, thus closing the cycle. We consider the mobile sink moving to either neighbor with equal probability at each step. We have the following two results:

Theorem 4.1: If the distribution of alerts is uniform, the cost c_{bc} of alerting the mobile sink using the BC

protocol is on average half that of the cost c_f for the full flooding, and is equal to $2c^*$.

$$c_{bc} = \frac{c_f}{2} = 2c^* \quad (2)$$

Theorem 4.2: The notification delay d_{bc} for the BC protocol is equal to twice the delay for the full flooding and to twice the optimal delay d^* .

$$d_{bc} = 2d_f = 2d^* \quad (3)$$

Proof: Denote by i the position of the alert, and j the position of the sink. Since the network is closed into a cycle, all vertices are equivalent, and we can take $i = 1$ with no loss of generality. Sending a message from the alert to the sink can take exactly two paths here, either clockwise, or counter-clockwise. Flooding will send exactly N messages around both directions of the cycle, while the BC protocol will send packets in only one direction, that of the lesser ages. Since j is uniformly distributed over $\{1, \dots, N\}$, the cost on average will be $N/2$.

The optimal cost is the shortest distance between 1 and j , ie. the smallest of $j-1$ and $N+1-j$. On average, the cost is thus equal to $N/4$.

For the delay, some packets in the flooding mechanism will take the optimal path, thus $d_f = d^*$. The BC algorithm might take a longer path. The cost computation translates into a delay of twice the optimal delay. ■

While the BC protocol is far from optimal in this scenario, it is a significant cost improvement over flooding.

Line segment: We consider now the same graph, but as a straight line, without the connection $N \leftrightarrow 1$. Again, we assume that the mobile sink moves left or right with equal probability, except at node 1 where it can only go right, and node N where it can only go left. It is easy to compute the steady state distribution of this random walk. Denote by π_k , $1 \leq k \leq N$ the probability that the mobile sink is at vertex k , then:

$$\begin{aligned} \pi_k &= \frac{1}{N-1} \text{ for } 2 \leq k \leq N-1 \\ \pi_1 &= \pi_N = \frac{1}{2(N-1)} \end{aligned} \quad (4)$$

Theorem 4.3: If the distribution of alerts is uniform, the cost c_{bc} of alerting the mobile sink using the BC protocol is on average half that of the cost c_f for the full flooding, and is equal to c^* .

$$c_{bc} = \frac{c_f}{2} = c^* \quad (5)$$

Theorem 4.4: The notification delay d_{bc} for the BC protocol is equal to the delay for the full flooding and to the optimal delay d^*

$$d_{bc} = d_f = d^* \quad (6)$$

Proof: The delay result is trivial: flooding floods from the alert in two directions, while the optimal solution and the BC protocol only flood in the right direction. In any case, all three reach the sink at the same time.

The main difference here is in cost. It is trivial that $c_{bc} = c^*$: the node which receives the alert knows on which side is the mobile sink using the freshest age of its two neighbors. We only need to compare c_f and c_{bc} .

Define i to be the position of the alert, j the position of the sink. The cost c_{bc} is equal to $|i - j|$, while the cost c_f is given by:

$$\begin{aligned} c_f &= j \text{ if } j > i \\ c_f &= N - j \text{ if } j < i \\ c_f &= 0 \text{ if } i = j \end{aligned} \quad (7)$$

Using the distribution in (4) and the fact that the alert is uniformly distributed, one can compute the average cost. Since, for a fixed j , $P(i > j) = (N - j - 1)/N$, and $P(i < j) = (j - 1)/N$, the cost c_f is:

$$\begin{aligned} c_f &= \sum_{j=1}^N \pi_j \frac{j(j-1) + (N-j)(N-j-1)}{N} \\ &= \sum_{j=1}^N \pi_j \frac{2j^2 - (2N+1)j + N(N-1)}{N} \end{aligned} \quad (8)$$

For large N , we can approximate π with a uniform distribution, and we obtain the cost:

$$\begin{aligned} c_f &= \frac{1}{N^2} \left(2\sum j^2 - (2N+1)\sum j + N^2(N-1) \right) \\ &= N - 1 - \frac{(N+1)(2N+1)}{6N} \\ &\sim \frac{2N}{3} \end{aligned} \quad (9)$$

using, to get the 2nd line, the well known formulas for the sum of the integers and the sum of the squares.

We lastly need to compute the actual cost c_{bc} :

$$\begin{aligned} c_{bc} &= \sum_{j=1}^N \pi_j \left(\frac{\sum_{i=1}^{j-1} j-i}{N} + \frac{\sum_{i=j+1}^N i-j}{N} \right) \\ &\sim \frac{N}{3} \end{aligned} \quad (10)$$

where we again approximated the distribution of j as uniform and took the asymptotic behavior for large N . Comparing (9) and (10) yields the desired result. ■

In the case of the line segment, the BC protocol is optimal. Furthermore, we see that in the 1-dimension

case, the cost for all three protocols has the same scaling behavior as $O(N)$, so it is rather indifferent which one to use. We do not expect the behavior to be optimal in 2 dimensions, but we expect the scaling behavior to be asymptotically optimal as well, a significant improvement over flooding the network. This discussion is the object of the next section.

B. 2 dimensional graphs in $\mathbf{Z} \times \mathbf{Z}$

We now consider a 2-dimensional network with N still being the total number of nodes. The position of the sink $S(t)$ at time t is now a point in $\mathbf{Z} \times \mathbf{Z}$.

Torus: We now consider a $\sqrt{N} \times \sqrt{N}$ torus. Each node of coordinates $(n, m) \in \{1, \dots, \sqrt{N}\}^2$ thus has four neighbors, $(n-1, m)$, $(n+1, m)$, $(n, m-1)$, and $(n, m+1)$ with all the additions and subtractions modulo \sqrt{N} . For the uniform random walk on the torus, the distribution of the steady state position of the sink in the torus is uniform over all vertices.

In this situation, if one assumes that the alert are spaced far enough in time so that the distribution of the position of the sink is independent of the location of the alert, one can evaluate some specific aspects of the performance, namely the average cost and delay, as well as the worst case cost and delay.

For the flooding, the cost is equal to N messages to flood the whole network, and the delay is on average $\sqrt{N}/2$.

For the BC protocol, the delay and the cost are difficult to assess. One can state the following theorem:

Theorem 4.5: The cost of routing using the Bread Crumbs protocol is linear with \sqrt{N} and scales asymptotically as shortest path routing in the ratio:

$$\lim_{N \rightarrow \infty} \frac{c_{bc}}{c_{sp}} = 2\sqrt{\pi}. \quad (11)$$

This states that following the BC trail rather than the shortest path only implies a constant *stretch* factor. Whether or not such a stretched route is acceptable depends on different factors (number of packets, mobility of the sink) but knowing this ratio allows to perform the trade-off optimization.

Proof: To prove the result, we need to introduce some definitions. Define the Loop-Erased Random Walk (LERW, [18]) for a sequence of points $S(0), S(1), S(2), \dots, S(t)$ of the random walk S up to time t . For two points u and v on the graph, the Loop-Erased Random Walk $L(S)(u, v)$ is constructed as follows: take the highest index l_0 such that $S(l_0) = v$. Define l_u the highest index such that $S(l_u) = u$. We

denote by l_i the smallest index, such that $l_u \leq l_i < l_0$ and $S(l_i + 1) = S(l_{i-1})$. Iterate the construction until $l_j = l_u$.

$L(S)(u, v)$ constructs the BC trail left by the mobile sink and the path $S(l_j), S(l_{j-1}), \dots, S(l_1), S(l_0)$ is the route followed by a packet routed from node u to a sink located at node v .

Due to the symmetry of the torus, we can assume without loss of generality, that the sink is located at node $v = 0$ at time 0 and we can consider time indexed by the negative integers. For each node u_i in the graph, we can thus construct a path to the origin $L(S)(u_i, 0)$.

One key property to notice is, for two nodes x and y , if y is on the path $L(S)(x, 0)$ then $L(S)(y, 0) \subset L(S)(x, 0)$. This is a direct consequence of the construction of the LERW. This ensures that the construction of $L(S)(u_i, 0)$ for all nodes i construct a unique spanning tree.

Pemantle [19] has shown that the tree constructed this way is akin to choosing a spanning tree uniformly among all possible subgraphs that are spanning trees. This means that, to prove our result, we now only need to show that, for a uniform spanning tree, the average branch height is of the order \sqrt{N} .

We invoke the following theorem [21]:

Theorem 4.6: The average height of a planted plane tree with n nodes, considering all such trees to be equally likely, is

$$\sqrt{\pi N} - \frac{1}{2} + O(N^{-1/2+\epsilon}) \text{ for } \epsilon \rightarrow 0 \text{ and } N \rightarrow \infty. \quad (12)$$

Theorem 4.6 gives us the asymptotic behavior for $c_{bc}(N)$. As for c_{sp} , the shortest path on the $\sqrt{N} \times \sqrt{N}$ torus, it is equal on average to $\sqrt{N}/2$ when the distance from 0 to (x, y) is measure with the L1 norm, $|x| + |y|$. Taking the ratio and the limit as $N \rightarrow \infty$ concludes the proof. ■

This result states that on average, the path length will grow linearly with a constant stretch factor.

Worst Case Path: To see that the scaling behavior is a property of the loop-erasure built into the BC protocol, consider the cover time of the random walk on the same torus graph. The cover time is the longest path in the original sequence S between the sink at the origin, and any other node in the graph. This is the distance covered by the sink without taking the available short-cuts and without removing the loops. The cover time is given by a result in [20], which we state below with our notations:

Theorem 4.7: The cover time of the simple random

walk on the torus is given by:

$$\frac{4}{\pi} N (\log \sqrt{N})^2 \quad (13)$$

This is a behavior which is worst than flooding the network!

The following result is a direct consequence of LERWs, but gives an idea of the worst case behavior of the BC routing, namely the length of the longest path:

Theorem 4.8: The BC trail from the origin to the points at the edge of the torus $\sqrt{N} \times \sqrt{N}$ has length asymptotically equal to:

$$O(N^{\frac{5}{8}}) \quad (14)$$

Proof: The length of the BC trail from the origin to the side of the $\sqrt{N} \times \sqrt{N}$ square, is \sqrt{N}^α , where α is called the growth exponent of the LERW. Kenyon [22] recently proved that $\alpha = 5/4$. ■

In the interest of space, we focused on the torus in this section; however, the results can be extended to an $\sqrt{N} \times \sqrt{N}$ lattice and the corresponding random walk movement for the sink which ensures a uniform stationary distribution.

C. Random Geometric Graph

We have so far focused on topologies built upon \mathbf{Z} or $\mathbf{Z}^2_{\sqrt{N}}$. However, most networks of interest would follow a random spatial distribution of the nodes. In this section, we consider a random geometric graph, a class of graphs widely used in the ad hoc network literature. A random geometric graph (RGG) is typically defined as a graph $\mathcal{G}(N, r)$ obtained by distributing uniformly N nodes in the unit torus², and connecting two nodes if and only if their Euclidian distance is less than r .

We add the following assumption, consistent with our set-up so far: that r is large enough such that the graph is connected and such that the whole unit area is covered.

We define the corresponding movement of the random walk to be the usual canonical random walk on a RGG: the sink located at node u picks its next node to attach to uniformly amongst the neighbors of u .

Theorem 4.5 still holds, for the right relation between N and r :

Theorem 4.9: For a constant $c > 1$ and $r = \sqrt{2c \log(N)/N}$, the cost of routing using the Bread Crumbs protocol scales asymptotically as the shortest path.

²A unit square, or unit disk would exhibit the same asymptotic behavior as well.

Note that we are unable in this situation to provide a closed form expression for the stretch factor, ie. the proportionality ratio between the BC trail and the shortest path. Also, note that for r defined as in the statement of the theorem, the connectivity of the network and the coverage of the whole square is an immediate consequence (cf. [23], [24]).

Proof: We define a lattice on the unit area by paving it with squares with length $r/\sqrt{2}$. Since all the nodes in one square are within communicating radius of one another, once the sink visits one square, and attaches to one node, it is overheard by the other nodes in the square, and due to the density imposed by the parameters r and N , by nodes in all other neighboring squares with high probability. Thus the problem of routing a packet to the sink is equivalent to routing the packet from square to square until it reaches the square the sink is in.

Since the paving defines a lattice structure, we can define a path on the lattice and can call upon Theorem 4.5 once we prove that the path on the constructed lattice converges to a uniform random walk on the lattice.

We do this in two steps: (i) we construct a modified version of the sink random walk which converges to a uniform stationary distribution, yet yields the same LERW as the original version of the sink random walk. Then (ii) we show that this random walk maps to a random walk on the paving which converges to a uniform random walk.

(i) The random walk at node u at time t picks its next node at time $t+1$ by choosing with probability $1/\delta_u$ one of the neighbors of u , where δ_u is the degree of u . This yields a stationary distribution for which the probability π_u that the sink is at node u is $\delta_u/2|\mathcal{E}|$. As in section II, $|\mathcal{E}|$ is the cardinality of the set of edges in the graph.

Denote by δ_{max} the highest degree $\max_u \delta_u$. By adding $\delta_{max} - \delta_u$ virtual nodes at node u , and by setting the transition of the Markov process \hat{S} to be:

$$\begin{aligned} P(\hat{S}(t+1) = v | \hat{S}(t) = u) &= \frac{1}{\delta_{max}} \text{ for all } v \leftrightarrow u, \\ P(\hat{S}(t+1) = u | \hat{S}(t) = u) &= \frac{\delta_{max} - \delta_u}{\delta_{max}} \text{ otherwise,} \end{aligned}$$

one can see that all nodes, including the virtual nodes, have the same degree, and that as a consequence, the process \hat{S} converges to a uniform distribution over the set of nodes.

Since, by performing this construction, we only have extended the stay of the random node at nodes with a lower degree, and since a longer stay at any given point is removed by the loop-erasure procedure, we have not

modified the LERW created from the sample path of the random walk.

(ii) For $c > 1$ and $r = \sqrt{2c \log(N)/N}$, it is known (cf. for instance Lemma 3.4 in [25]) that the number of nodes in any square of side length $r/\sqrt{2}$ is w.h.p. $\Theta(c \log N)$.

Thus the movement of the random walk \hat{S} maps into a random walk on the lattice by assigning $\tilde{S} \in \mathbf{Z} \times \mathbf{Z}$ to be the coordinate of the square in which the process finds itself in.

For large enough N , this converges to a uniform random walk on the lattice, and we can apply theorem 4.5 to see that the path length is within a constant factor of the shortest path. ■

V. BC PROTOCOL WITH DIFFUSION

In this section, we explore the behavior of the BC protocol when the nodes visited by the sink are allowed to *diffuse* the time of the visit to their neighbors. The visits of the sinks create an age gradient in the line visited by the sink. Since we assume a random walk with uniform stationary distribution, all the nodes end up visited, and an age gradient is formed over the whole graph.

The BC trail is the line of steepest descent on this gradient, with the sink at age 0 being the lowest point. As we have seen, this line is asymptotically equivalent to the shortest path. However, it is tempting to improve the diffusion of the age gradient to all nodes in the network, by having a node visited by the sink share its sink visit time with its neighbors.

We consider again the random walk on the $\sqrt{N} \times \sqrt{N}$ torus. An intuitive diffusion process would be for a node u visited by the sink at time t to alert its neighbors. The three neighbors not visited immediately prior to u can set the age of the sink visit to $t - (1 + \epsilon)$ with $0 < \epsilon \ll 1$, and thus create a *virtual visit* by the sink. We call this the Bread Crumbs plus Diffusion (BC+D) protocol. The diffusion can be extended over k hops, where nodes k hops away set their age to $t - k(1 + \epsilon)$, where now $k\epsilon \ll 1$.

The packets is routed in exactly the same manner. Intuitively, setting k to some constant value thickens the trail left by the sink from a line into a ribbon. In the case of BC+D with $\epsilon = 0$, there might be several candidates to forward a node, ie. several nodes with a label $t+1$ neighboring the node with label t . The role of the ϵ is thus to break these ties, and to ensure that the packet stays on the outside edge of the "diffusion ribbon", as the reader can easily convince his/herself that: this is

where the more opportunities for short-cuts will be, and that in the absence of short-cuts, the path will be of the same length than the central line of the ribbon. This can be proved formally, if tediously.

The parameter k captures the trade-off between accuracy of the sink position, and cost of the overhead: for $k = 0$, this is the BC protocol we have been studying so far. For k of the order of \sqrt{N} , all nodes are informed of the new sink position after each sink movement. This corresponds to having perfect knowledge of the location of the sink at all nodes at all time.

Since the ribbon includes the line, the path defined by the BC+D is at most as long as the BC trail. Since it is by definition longer than the shortest path, the BC+D will have the same scaling behavior as the shortest path and the BC trail, but with a different stretch factor.

We will look at the stretch factor value in the simulation evaluation, but we cannot compute its value based on k . For now, we can state the result that we can attain, namely:

Theorem 5.1: The BC+D route is strictly shorter than the BC route when averaged for all nodes in the $\sqrt{N} \times \sqrt{N}$ torus.

Proof: We only highlight the steps of the proof, as the result is pretty intuitive, and the detailed proof is space consuming.

The main insight of the proof is as follows: it is equivalent to (a) compute the path directly using the age gradient implied by BC+D, or to (b) first compute the LERW implied by BC and add the diffusion along this LERW path only.

(a) and (b) yield the same path. However, from (b) one sees easily that the path cannot be longer than the LERW path. We need to prove a strict inequality on the average length of the path.

To see it is strictly shorter, one can see that the BC and the BC+D trail differ in length only if there is a new loop removed in the BC+D. One possible loop is removed from BC+D if two nodes in the LERW $L(S)$ satisfy:

$$\exists s, t \in \mathbf{Z}, |t - s| > k \text{ and } |L(S)(t) - L(S)(s)| \leq k \quad (15)$$

The LERW $L(S)$ is self-avoiding, but if two points come near enough from each other, then the diffusion finds a short cut. This happens with a probability which is strictly positive. To see this, one can draw a $k + 2$ hops pattern which satisfies (15). Since it involves only a finite numbers of moves of the LERW, each with a positive transition probability, the pattern has a strictly

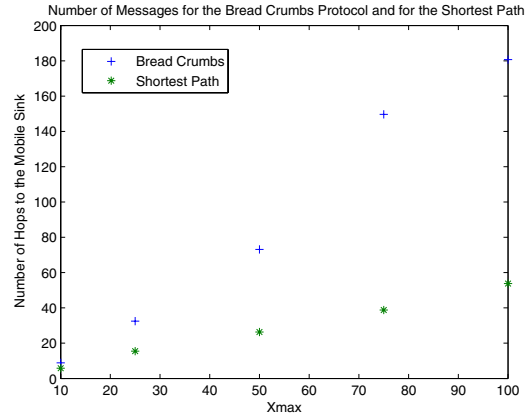


Fig. 3. Bandwidth Gain for Bread Crumbs protocol on a Torus

positive probability of happening, and thus the shortcut as well. ■

VI. SIMULATION

In order to assess the dynamics of the protocol in a general setting, we conducted some simulations performed using Matlab.

Torus simulation: We consider a square torus with $X_{max} = \sqrt{N}$ to be the number of nodes along one side of the square. We ran a random walk to simulate the movement of a sink. Once the mobile sink is attached to one node, it chooses with equal probability one of its neighbor to attach to at the next time slot.

We generated randomly distributed alert and computed the shortest path and the path taken using the BC routing. The shortest path is also proportional to the delay for the sink to receive the alert when using flooding. The cost of flooding is equal to $N = X_{max}^2$ in terms of number of messages.

We simulated the system for X_{max} varying between 10 and a 100, which is plotted on Figure 3. The first observation is that both the Shortest Path and the Bread Crumbs routing behave linearly in terms of number of messages. This is the reason why we do not plot the cost of the flooding: it grows much faster and dwarves the Shortest Path and the Bread Crumbs. The respective cost for $X_{max} = 100$ is about 60 for the shortest path, 180 for the Bread Crumbs, and 10,000 for the flooding!

Lattice Simulation: Since the torus is not a practical topology in real life, we also simulated a finite square lattice with size X_{max} with reflections at the boundary for the random walk of the mobile sink, shown in Figure 4. The shortest path is a little bit longer and the bread crumbs trail is a bit shorter, but the asymptotical behavior is similar.

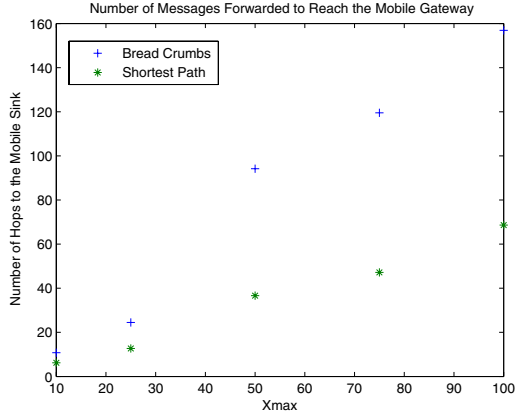


Fig. 4. Bandwidth Gain for Bread Crumbs protocol on a Lattice

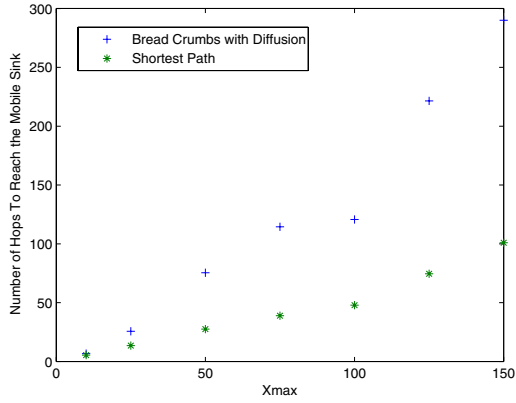


Fig. 5. Bandwidth Gain for Bread Crumbs + Diffusion protocol on a Torus

Bread Crumbs plus One-Step Diffusion: We consider here the improvements brought by the diffusion extension of the BC protocol, with $k = 1$. We simulated the BC+Diffusion protocol on the torus and the result is displayed in Figure 5. The scaling behavior is similar with and without diffusion, but the one-step diffusion bring about a slower slope of the routing cost.

In Figure 6, we compare the gradient profile obtained by the BC and the BC+D protocols, where again $k = 1$. One can see that the profile of the BC+D is much flatter, meaning that there are fewer valleys for the packets to get side-tracked.

VII. CONCLUDING REMARKS

We mentioned the assumption that the mobile sink never leaves the coverage area, and would like to add that, in the case there are holes in the coverage, protocols can be designed to notice when the sink leaves the coverage and to find the next hop accordingly. For

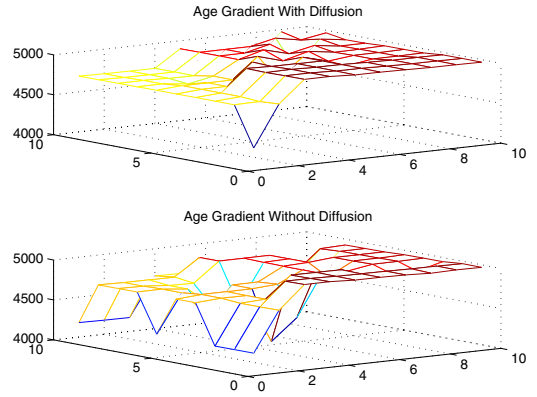


Fig. 6. Gradient Created by the Mobile Sink with and without Diffusion

instance, the one-hop diffusion extension implies that each node visited at time t should receive a diffusion message at time $t + 1$. Failing to receive this message would indicate a hole in the coverage, and could trigger an expanded ring search for the next hop.

In conclusion, we have studied some properties of the simplest protocol to route data from a static sensor network to a mobile gateway. The so-called Bread Crumbs protocol is simple, and very attractive as it adds very little overhead to the network management control plane in terms of building route tables or routing gradients, and as it is totally distributed. It is even robust to the failure of a few nodes.

We have shown that, in two dimension, the gain of such protocol is dramatic, and the scaling behavior is now similar to that of the optimal shortest path. Little Tom Thumb went home optimally. The path scales as $O(\sqrt{N})$ as opposed to $O(N)$ for flooding, where N is still the total number of nodes. In the case of a lattice structure, we have computed the *stretch factor* for the path of the BC protocol, when compared to the shortest path, and have shown that it is equal to $2\sqrt{\pi}$. We have provided bounds on the worst case cost to find the mobile sink, as well as some simulation results which validate the analysis.

We also have presented a diffusion extension of the protocol which widens the trail left by the mobile sink and have shown by simulation that it strictly reduces the path length, but exhibits the same scaling behavior, linear with $O(\sqrt{N})$.

Further research areas would involve designing and implementing actual protocols which allow routing in the case of holes in a non-continuous coverage, or in the

case where some level of mobility is allowed among the sensor nodes. Even in a static networks, nodes disappear or connectivity evolves, and there are technical challenges in designing a protocol which takes this into account.

ACKNOWLEDGMENTS

We would like to thank Prof. Yuval Peres for answering our questions and mentioning [22] to us.

REFERENCES

- [1] ZigBee Alliance, www.zigbee.org
- [2] BlueTooth Wireless Technology, <http://www.bluetooth.com/bluetooth/>
- [3] Tropos Networks, *Ubiquitous, metro-scale Wi-Fi mesh network systems*, <http://www.tropos.com/>
- [4] Bel Air Networks, *Scalable Wide-Area Wi-Fi for data, voice and video*, <http://www.belairnetworks.com/>
- [5] Strix Systems, *Metropolitan Mesh Networks for communities across the globe* <http://www.strixsystems.com/>
- [6] Skypilot Networks, *Synchronous Mesh Infrastructure* <http://www.skypilot.com/>
- [7] P. Venkatasubramaniam, Q. Zhao and L. Tong, *Sensor Networks with Multiple Mobile AccessPoints*, in Proc. of the 38th Annual Conference on Information Sciences and Systems (CISS'04), Princeton, NJ, March 2004.
- [8] P. Venkatasubramaniam, S. Adireddy and L. Tong, *Sensor Network with Mobile Access: Optimal Random Access and Coding* in IEEE Journal on Selected Areas in Communications, vol.22, No. 6, August 2004.
- [9] S. Basagni, A. Carosi, E. Melachrinoudis, C. Petrioli, M. Z. Wang, *A new MILP formulation and distributed protocols for wireless sensor networks lifetime maximization*, in Proc. of IEEE ICC 2006.
- [10] J. Luo, J.-P. Hubaux *Joint Mobility and Routing for Lifetime Elongation in Wireless Sensor Networks*, in Proc. of IEEE INFOCOM 2005.
- [11] J. Luo, J. Panchard, M. Piorkowski, M. Grossglauser, J.-P. Hubaux, *MobiRoute: Routing towards a Mobile Sink for Improving Lifetime in Sensor Networks* in the 2nd IEEE/ACM DCOSS, 2006.
- [12] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, *Analysis and Optimization of Randomized Gossip Algorithms*. In Proc. of the 43rd Conference on Decision and Control (CDC 2004), 2004, Paradise Island, Bahamas.
- [13] C. Perrault, *Les Contes de Perrault, (ou Conte de ma Mère l'Oye)*, 1697.
- [14] J. Grimm, W. Grimm, *Fairy Tales*. (1812) (re-edited in 1982: Julian Messner. ISBN 0-671-45648-2 AACR2.)
- [15] K. Whitehouse, C. Sharp, E. Brewer, D. Culler, *Hood: a Neighborhood Abstraction for Sensor Networks*, in Proc. of MobySys 2004, Boston, USA.
- [16] M. Grossglauser, M. Vetterli, *Locating nodes with EASE: Last Encounter Routing for Ad Hoc Networks through Mobility Diffusion*. in Proc. of Infocom 2003, San Francisco, USA.
- [17] H. Dubois-Ferrière, M. Grossblauser, M. Vetterli, *Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages* in Proc. MobiHoc 2003, Annapolis, USA.
- [18] G. Lawler, *Loop-Erased Random Walk* in Perplexing problems in Probability, pp. 197-217. Birkhauser Boston, Boston, MA, 1999.
- [19] R. Pemantle, *Choosing a spanning tree for the integer lattice uniformly*, Annals of Prob. 19, 1559–1574, 1991.
- [20] A. Dembo, Y. Peres, J. Rosen, O. Zeitouni, *Cover Time for Random Walk and Brownian Motion in Two Dimensions* Preprint.
- [21] N. De Bruijn, D. Knuth, and S. Rice, *The average height of planted plane trees*, In Graph Theory and Computing (1972), R.C. Read, Ed., Academic Press, 15–22.
- [22] R. Kenyon, *The Asymptotic Determinant of the Discrete Laplacian*, Acta Math 185 (2000) no. 2, 239-286.
- [23] P. Gupta, P.R. Kumar, *Critical Power for Asymptotic Connectivity in Wireless Networks*, In Stochastic Analysis, Control, Optimization and Applications, (1998) pp. 547-66.
- [24] M. Penrose, *The Longest Edge of the Random Minimal Spanning Tree*. in Annals of Applied Probability 7 (1997) pp. 340-61.
- [25] C. Avin, G. Ercal, *On the Cover Time of Random Geometric Graphs*. Submitted to J. Theoretical Computer Science, Nov 2005.