

Performance of Routing in Sensor Networks with a Mobile Access Point

Cédric Westphal

Abstract—Routing streaming data in a sensor network is a task made arduous by the resource constraints imposed on each 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 about the network.

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 protocol and show that it is optimal in its scaling behavior in one- and two-dimension graphs. 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 positively support the analysis with some simulations of the bread crumbs protocol in both a lattice and a torus scenario.

Index Terms—Ad-hoc networks, sensor network, routing.

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 become 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 ubiquitous, the mobile phone carried by everyone 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 nodes to the mobile sink creates new challenges. For instance, giving the nodes 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 where the sink is, then it will initiate a flooding of the network, causing unnecessary expenditure of bandwidth and, most importantly, power.

In this paper, we study a basic protocol, the so-called Bread Crumbs protocol, to route data to a mobile sink. 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.

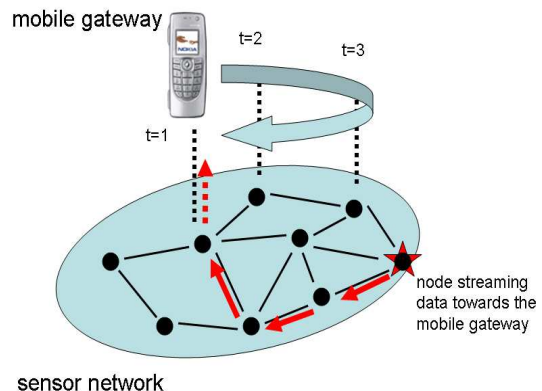


Fig. 1. Routing Data Streams in a Sensor Network with Mobile Access

The contribution of the paper is as follows: we show that in single dimension, the scaling behavior of the protocol is similar to that of flooding in terms of cost and delay, and is linear with respect to the number of nodes.

However, we show that in two dimensions, the Bread Crumbs (BC) routing still scales proportionally to the optimal path, that is proportional to the square root of the number of nodes, while flooding is linear with the number of nodes. This means that even a protocol as simple as the BC protocol provides a huge improvement in terms of economizing the sensor network's resources and

is asymptotically optimal.

In the next section, we describe the model and review some of the literature in the domain of wireless sensor networks with a mobile access point. In section III, we detail the protocol to route packets from the sensor network to a mobile sink. In section IV, we formally study the bandwidth gain from the protocol compared to basic flooding. In section V, we provide some numerical results as well as describe a one-step diffusion extension of the protocol.

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: it cannot attach to two sensors successively if there is no edge connecting the corresponding vertices. However, this restriction is rather limited: most sensor networks should be dense enough so that there is no hole in the covered area for the sink to cross. And 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.

We thus 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 $t \in \mathbf{Z}$.

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

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 [5] 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 [3], [4] in a more information theoretic context. We next study an algorithm which finds the sink all the time, but at a cost equal to a fraction of flooding the network.

III. FINDING THE MOBILE SINK: THE BREAD CRUMBS PROTOCOL

To route a packet to the sink, the nodes used a steepest descent algorithm. The gradient is created as follows: each time the mobile sink last attach to the wireless sensor, the sensor start a timer. This timer reflects the age of the last visit. The age of the visit is basically the pebbles or bread crumbs that our mobile sink leaves behind like in the Little Tom Thumb fairy tale. For this reason we denote our protocol the Bread Crumbs (BC) protocol.

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 that 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 [6], and the protocol can be considered as a version of last encounter routing protocols [7], [8] where only one node is mobile.

When a sensor receives an alert, it forwards it to the packet 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 with a more recent age forward it. In this scenario, packets might be send several times, so it is not the preferred solution. The other form requires knowledge of the age of the neighbor, for instance as a parameter included in a hello handshake. Since the delta between two vertices' ages stays constant, there is no need to periodically broadcast one's age, only to update it after another visit of the sink.

There are a few properties of this bread crumbs protocol:

- it is very simple, with no overhead added to locate the mobile node,
- 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 , and c^* , respectively, the cost (in number of messages sent) for the Bread Crumbs protocol, for the full flooding, and for the optimal cost. Similarly, we denote by d_{bc} , d_f and d^* the delay (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. However, in the remainder of the document, it is taken to be uniform over all vertices.

IV. ANALYSIS

We first state some simpler results to build some intuition.

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 IV.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 IV.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 IV.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 IV.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:

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

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:

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

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

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

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} :

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

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 a significant improvement over flooding the network. This discussion is the object of the next section. a forthcoming document.

B. 2 dimensional graphs

We now consider a 2-dimensional network with N still being the total number of nodes.

Complete Graph: A complete graph is a graph for which each node is one hop away from any other node. There are few practical networks with this property. However, we include the discussion here as this is the worst case of the bread crumbs protocol in terms of delay: an alert message has to follow the physical path of the sink, minus the loops, while in the shortest path, it only has to travel one hop. Using the same notations as before:

$$\begin{aligned} d_f &= 1 \\ d_{bc} &= O(N) \quad (11) \end{aligned}$$

The value of d_{bc} actually depends on the topology and the random walk driving the motion of the sink, so other values can be inserted here depending on the scenario.

In terms of bandwidth cost, $c_f = N$ as it takes one message to the N neighbors to propagate the alert message. The cost in bandwidth is strictly less with the BC protocol as $c_{bc} = d_{bc} < N$ on average. The BC protocol trades of bandwidth for delay.

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 IV.5: The cost of routing using the Bread Crumbs protocol is linear with \sqrt{N}

Proof: Due to space constraints, we omit the proof of the theorem. The proof can be found in a longer technical report version of this paper [9]. However, it is a consequence of properties of self-avoiding random walks as the bread crumbs trail is in essence a loop-erased random walk [10]. ■

This result states that on average, the path length will grow linear. A bound on the worst case worst case is given by the next result:

Theorem IV.6: The *worst case cost* to find the mobile sink scales with N as:

$$c_{bc} = d_{bc} = \frac{4}{\pi} N (\log \sqrt{N})^2 \quad (12)$$

Proof: The worst case cost is bounded by the cover time of the lattice by a random walk. It is bounded above since the cover time include the loops in the path, and the actual bread crumbs path is a loop-erased random walk. From [11] one can find the asymptotical behavior of the cover time on the 2-dimensional torus. Again, the details of the proof are omitted. ■

V. 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 2. 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!

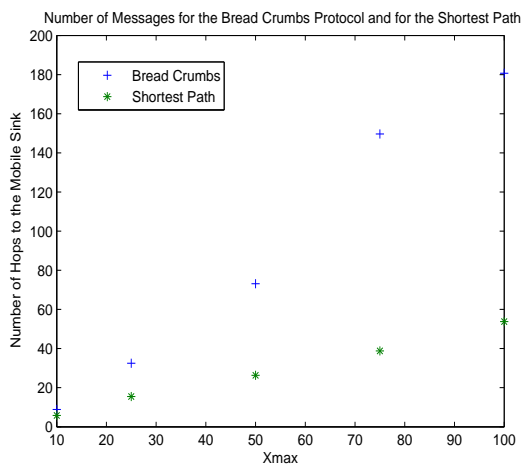


Fig. 2. Bandwidth Gain for Bread Crumbs protocol on a Torus

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, which Figure ?? describes. The shortest path is a little bit longer and the bread crumbs trail is a bit shorter, but the asymptotical behavior is similar.

VI. 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,

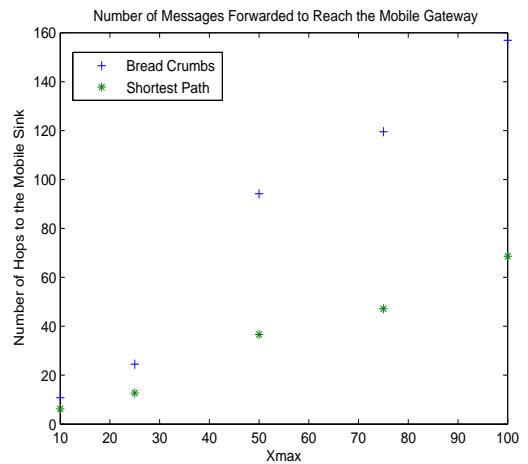


Fig. 3. Bandwidth Gain for Bread Crumbs protocol on a Lattice

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 one dimension, the gain of such, or any, protocol is rather limited, as flooding is within a constant factor of the cost optimal routing scheme. The scaling behavior of the BC routing cost, as the number of nodes N grows to ∞ , is $O(N)$, same as flooding the network, and same as using the shortest path.

However, in two dimension, the gain of such protocol is dramatic, and the scaling behavior is now similar to that of the optimal shortest path. It scales as $O(\sqrt{N})$ as opposed to $O(N)$ for flooding, where N is still the total number of nodes. We have provided bounds on the worst case cost to find the mobile sink, as well as some simulation results which validate the analysis.

Future research would focus on how to improve the performance of the protocol without incurring overhead. Diffusing the age of the last visit to a node's neighbors seems to expand the trail left by the mobile node without adding much overhead, and could be a potentially rich research direction.

REFERENCES

- [1] ZigBee Alliance, www.zigbee.org
- [2] Bluetooth Wireless Technology, <http://www.bluetooth.com/bluetooth/>
- [3] 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.
- [4] 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.

- [5] 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.
- [6] K. Whitehouse, C. Sharp, E. Brewer, D. Culler, *Hood: a Neighborhood Abstraction for Sensor Networks*, in Proc. of MobySys 2004, Boston, USA.
- [7] 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.
- [8] H. Dubois-Ferrière, M. Grossglauser, M. Vetterli, *Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages* in Proc. MobiHoc 2003, Annapolis, USA.
- [9] C. Westphal, *Little Tom Thumb Went Straight Home: Asymptotic Behavior of a Routing Protocol in Ad Hoc Networks with a Mobile Access Point* to appear in Proc. of IEEE Infocom 2007, Anchorage, USA.
- [10] G. Lawler, *Loop-Erased Random Walk* in Perplexing problems in Probability, pp. 197-217. Birkhauser Boston, Boston, MA, 1999.
- [11] A. Dembo, Y. Peres, J. Rosen, O. Zeitouni, *Cover Time for Random Walk and Brownian Motion in Two Dimensions* Preprint, submitted.