MobiRoute: Routing towards a Mobile Sink for Improving Lifetime in Sensor Networks *

Jun Luo, Jacques Panchard, Michał Piórkowski, Matthias Grossglauser, and Jean-Pierre Hubaux

School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
`firstname.lastname@epfl.ch`

Abstract. Improving network lifetime is an important issue involved in every aspect of the design and deployment of wireless sensor networks. There is a recent research trend of studying the application of a mobile sink to *transport* (rather than let the sensor nodes *transmit*) data back. Whereas this approach trades data delivery latency for reduced (node) energy consumption (and thus an improved lifetime), our experience tells us that sacrificing latency to extend lifetime is not necessary. In this paper, in line with our previous work, we investigate the approach that makes use of a mobile sink for balancing the traffic load and in turn improving network lifetime. We engineer a routing protocol that effectively supports sink mobility. Through intensive simulations in TOSSIM with a real implementation, we prove the feasibility of our mobile sink approach by demonstrating the improved network lifetime in several deployment scenarios.

1 Introduction

Apart from their great advantages, wireless sensor networks (WSNs) are subject to many limitations, among which the energy constraint becomes an increasing concern [1]. In WSNs, the size of a node should be sufficiently small to avoid altering phenomena of interest, and nodes, in many cases, should operate for a long period without human attendance. These requirements imply a capacity-limited and (possibly) non-renewable power source for each node. As a result, the longevity of WSNs under energy reserve limitations is a major problem that should be addressed before making use of such networks.

One important approach to maximizing the lifetime of WSNs consists in balancing the data flow at each link; this can be achieved typically through linear programming [2, 3]. Since the solution to a linear programming problem is guaranteed to be a global maximizer [4], the achieved network lifetime is optimum in given settings (e.g., certain locations and transmission powers of

* The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322. (`http://www.terminodes.org`)

nodes). However, it is possible that dynamically changing the settings could result in a better solution (i.e., network lifetime) that averages the solutions to a set of the above problems. Among all the possible changes that could be brought to a network, using mobile sinks seems to be a feasible and beneficial one [5–9].

We focus on a scenario where all the sensor nodes are fixed and have limited energy reserves, and where a mobile node endowed with significantly more resources serves as the data sink. In this scenario, the sink mobility can increase network lifetime through two different methods, depending on the relationship between the *mobility time-scale* and the *delay time-scale*. The mobility time-scale refers to the time over which the movement of the mobile sink covers a significant portion of the entire network; the delay time-scale refers to the tolerable delay that sensor data is allowed to incur between its origin and the sink.

In the *fast mobility* regime, the mobility time-scale is of the order of the delay time-scale. The WSNs may then take advantage of *mobility capacity* [10], i.e., the ability to transport information in part by physically carrying it in mobile nodes, rather than transmitting it through wireless links. In this *mobile relay* approach [5, 6, 8], the mobile sink "picks up" data from nodes and transports the data back with mechanical movements. By "picking up" we mean that the sink should move as close to a node as possible before asking the node to transmit its data. This approach trades data delivery latency for the reduction of energy consumption of nodes. We refer to [8] for a field study in this regime.

In the *slow mobility* regime, the mobility time-scale is longer than the required delay time-scale. In this case, the network cannot benefit from mobility capacity, as the delay bounds of most data packets would be exceeded. Therefore, data packets have to be carried from their origin to the sink through multihop transmissions. However, it has recently been observed [7, 9] that sink mobility can still offer benefits in terms of network lifetime. This is because nodes close to the sink deplete their energy reserves more quickly than other nodes, as they relay more traffic on behalf of others. In a fixed network, these nodes then exhaust their energy reserves very quickly, limiting the network lifetime. By moving the sink[1] around, even very **infrequently**, a load-balancing effect arises, by averaging the role of relay over many nodes. Therefore, in the slow mobility scenario, the network lifetime can be longer than in an equivalent static scenario, with no need to sacrifice latency.

We argue that the slow mobility conditions exist in many realistic applications of WSNs. For example, suppose that a WSN is equipped with batteries that cannot be replaced, e.g., because the sensor nodes are not accessible, or because changing batteries would be hazardous or costly. This may be the case in sensors in smart buildings, where batteries might be designed to last for decades, and in environmental or military sensing under hostile or dangerous conditions (e.g., avalanche monitoring). In this case, it may be desirable and comparatively simple to move a sink very infrequently (e.g., once a day or a week) by a human or by a robot. For example, in the avalanche monitoring scenario, a sink may

[1] Under this circumstance, the sink might need long-range (wireless) communication facilities to transmit data out of the considered sensor network.

be deployed at the periphery[2] of the monitored area, and moved by helicopter once in a while. In the building scenario, the sink can be "virtually" moved: computers in different offices serve as the sink in shifts. In the environmental and military monitoring scenario, moving the sink may require some effort, but it can be acceptable if done infrequently.

In the spirit of [9], we further investigate the performance, with respect to both lifetime and reliability (measured by packet delivery ratio), of WSNs with a mobile sink (in the slow mobility regime). We consider a scenario where nodes of a WSN periodically sample data and transfer these data through multi-hop routes towards the sink. We propose a routing protocol, MobiRoute, dedicated to support sink mobility. By performing intensive simulations with this protocol, our investigations take into account realistic conditions such as control overhead with a routing protocol and collision/overhearing [11, 12] at the MAC layer. We use TOSSIM [13] as the simulator. Our simulation results demonstrate the efficiency of MobiRoute, in terms of both an improved network lifetime and an undegraded reliability, in several deployment scenarios.

The rest of this paper is organized as follows: Section 2 clarifies the metrics and methodologies we apply. Section 3 presents our MobiRoute protocol. Section 4 describes the algorithm that control the sink mobility in an adaptive way. Simulation results are reported in Section 5. Section 6 surveys related work. Finally, Section 7 concludes the paper.

2 Problem, Metrics and Methodology

The *network lifetime* can be defined in various ways; these definitions focus on either individual [2] or collective [14] behaviors of nodes. Because the individuality has an implication on the collectiveness (e.g., the death of a node is soon followed by the death of all nodes one-hop away [7]), we define the network lifetime as the time for the first node to die [2]. When evaluating this quantity, we convert the problem of maximizing network lifetime to a min-max problem in terms of the **radio** energy consumption of individual nodes. Another performance index we want to evaluate is the packet delivery ratio (or *reliability*). In fact, a possible side-effect brought by sink mobility could be an increase in packet loss due to occasional topology changes; the lifetime elongation resulting from sink mobility is justifiable only if the increase in packet loss is tolerable.

In our approach, the mobility pattern of a sink takes a *discrete* form: the movement trajectory consists of several *anchor point*s between which the sink moves and at which the sink sojourns. We require the sojourn time to be much longer than the moving time, such that the routing overhead introduced by sink mobility becomes negligible due to its amortization across the sojourn period. Imposing these anchor points simplifies the design of the mobile sink[3] and limits the extra overhead introduced to the routing protocol (see Section 3 for details).

[2] As we have shown in [9], the optimum trace (in terms of network lifetime) for a mobile sink is the network periphery.

[3] The mobile sink can simply be a laptop (moved occasionally by a human), rather than a sophisticate robot as used in [8].

In addition, a continuous movement is not necessary, as a granularity of (sink) displacement smaller than the magnitude of the effective radio range may not lead to any topological change (whereas topological changes are what we expect from the sink mobility). In order to better adapt to the topology and dynamics of a given network, we also intend to control the sink mobility *on-line* (based on the *off-line* optimization described in [9]).

Our experiment methodology involves simulations with TOSSIM [13]. The main benefit of using the TOSSIM simulator is that the protocol used for simulations can be directly adopted by real sensor nodes. We simulate a set of networks with nodes on 4×4, 5×5, and 7×7 point lattices; these scenarios represent outdoor WSNs in general. We also simulate a network that we intend to deploy as an in-building testbed.

3 MobiRoute: Routing Data Towards a Mobile Sink

According to the definition of discrete mobility pattern described in Section 2, the sink changes its location from time to time. A routing protocol that transfers data towards such a sink should perform the following operations that are not needed for traditional WSNs:

1. Notify a node when its link with the sink gets broken due to mobility.
2. Inform the whole network of the topological changes incurred by mobility.
3. Minimize the packet loss during the sink moving period.

Operation 1 seems to be encompassed by 2, but the level of urgency is different. Packets forwarded by a last-hop node will get lost if the node does not detect the link breakage, while a remote node can still send its data to the sink successfully without knowing the topological changes. However, the routing optimality is compromised without operation 2. It is not possible to avoid packet loss, because a realistic failure detector (which usually relies on a timer) always has some delay. Therefore, the goal of operation 3 is to minimize rather than eliminate packet loss. Possible scenarios related to these operations are illustrated in Fig. 1.

Our routing protocol, MobiRoute, is a superset of Berkeley MintRoute [15]. MobiRoute extends MintRoute by adding functions that perform the aforementioned operations. We first introduce MintRoute briefly in Section 3.1, then we describe the extended functions of MobiRoute separately in Section 3.2 – 3.4. The state diagram shown in Fig. 2 is used when we present MobiRoute.

3.1 MintRoute

Berkeley MintRoute [15] is a routing protocol designed specifically for the all-to-one data transmission style of WSNs. It takes a distributed distance-vector based approach: route messages (i.e., control packets) are exchanged periodically among neighbor nodes, and the next hop nodes (or *parent*s in MintRoute nomenclature) are chosen by evaluating the costs of routing data through different neighbors. The exchanged route messages not only help to measure the

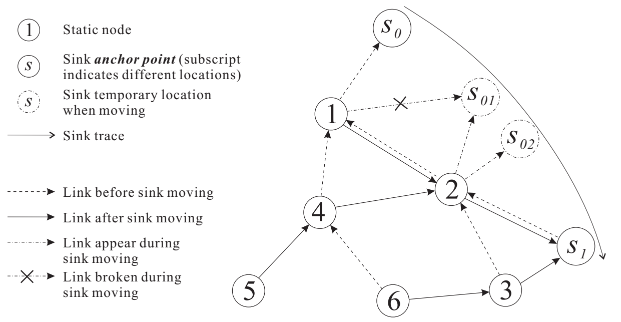

Static node
Sink *anchor point* (subscript indicates different locations)
Sink temporary location when moving
Sink trace
Link before sink moving
Link after sink moving
Link appear during sink moving
Link broken during sink moving
s_0
s_{01}
s_{02}
s_1
1
2
3
4
5
6

Fig. 1. This example illustrates possible scenarios where additional operations are necessary. Assuming the sink, after its (long) sojourn at s_0, moves to s_1, (1) the link breakage happening when the sink reaches intermediate location s_{01} (where it loses connectivity with node 1) should be notified to node 1, otherwise the node will have to drop packets sent from other nodes, (2) nodes 3, 4, and 6 should be informed about the topological changes at a proper time, otherwise, for example, 6 might take the following sub-optimal routing path: $6 \rightarrow 4 \rightarrow 1 \rightarrow 2 \rightarrow s$.

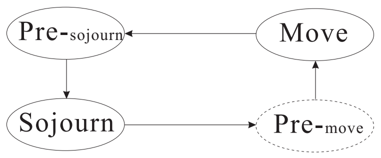


Fig. 2. States and transitions involved in MobiRoute. Note that only the protocol running at the sink side has the *pre-move* state.

distance (in terms of the number of possible transmissions) from the sink but also provide a way to evaluate the link qualities (from both directions) between nodes. As a result, MintRoute applies a **M**inimum **T**ransmission (MT) metric, where the goal is to minimize the total number of transmissions (including retransmissions). Since the data rate in WSNs is low, route messages do not need to be exchanged frequently (the rate is actually a multiple of the data rate in MintRoute). This helps MintRoute to greatly reduce its energy consumption. Although MintRoute does not explicitly apply a metric that considers load balancing, the protocol, according to our experience, does balance the traffic load with occasional switches of nodes' parents (which is a direct consequence of the MT metric). This feature makes MintRoute a leading candidate for supporting a mobile sink. Finally, we note that, in order to detect packet loss and thus evaluate link quality, MintRoute applies a sequence number for each packet; this sequence is shared by both control and data packets.

3.2 Detecting Link Breakage

In order to inform the nodes located close to the sink trace about the state of their links with the sink, MobiRoute applies a beacon mechanism. The sink, during the whole moving period, periodically broadcasts a beacon message (*s-beacon* hereafter). A node, upon receiving a s-beacon, sets (or resets) its **detecting** timer. If the timer times out before receiving the next s-beacon, the failure detector at this node indicates a link breakage and a new parent is chosen. We now discuss several crucial points of this seemingly simple mechanism.

First, we require the sink to transit from the **sojourn** state to the **pre-move** state before physically beginning to move. The sink begins to broadcast s-beacons under the pre-move state and evolves to the **move** state after a while. The sink moves while broadcasting s-beacons under the move state. A node, after receiving the first s-beacon under its current **sojourn** state, transits to the **move** state directly. The importance of the pre-move state can be straightforwardly seen: it guarantees the reception of s-beacons at the nodes' side before the link quality changes due to the sink mobility.

Secondly, although only the sink (who is assumed to have sufficient energy reserve) spends energy to send s-beacons, nodes also spend energy to receive these beacons. Therefore, the frequency of s-beacons should not be too high. On the other hand, low frequency sending retards failure detection, which in turn increases packet loss. We apply a simple heuristic: the frequency is set in the same order as the accumulative packet sending rate. For example, if the sending rate of each node is 1 packet/minute and there are 60 nodes in the network, the accumulative rate at a last-hop node is at most 1 packet/second, and the beacon frequency is set to 1Hz. A related parameter is the timeout value for the detecting timer. Fortunately, the value can be relatively small, because a node will detect a false-positive when receiving another s-beacon.

Finally, the beacon mechanism is a costly procedure, no matter which beacon frequency is chosen. Fortunately, since the moving period accounts only for a small fraction of the network lifetime, its costs will be amortized across the lifetime. A continuous sink movement, on the contrary, would incur such costs permanently.

3.3 Conveying Topological Changes

MobiRoute could have relied on MintRoute to propagate the topological changes resulting from sink mobility. However, the rate of route message exchanges in MintRoute is very low. Therefore, it takes a long time to convey the topological changes to the whole network; during this period, many packets are routed through sub-optimal paths, which consumes additional energy and thus offsets the benefit of sink mobility. As a result, MobiRoute needs a speed-up (route message exchange) rate for propagating the topological changes.

Propagating information throughout a network is a costly procedure (message complexity $O(n)$); it cannot be performed frequently. So MobiRoute only performs a propagation upon the sink reaching an anchor, and it tolerates a limited number of sub-optimal routing during the moving period. The sink enters the **pre-sojourn** state (see Fig. 2) when it stops moving; it then sends route messages with a speed-up rate, which causes their receivers to enter the same state. Nodes that receive messages **directly** from the sink also send speed-up route messages; they re-evaluate the quality of their links with the sink using these exchanges. A node receiving speed-up messages **indirectly** also enters the pre-sojourn state; it forwards the message only if its distance towards the sink changes significantly (e.g., node 6 in Fig. 1 might not forward messages received from node 3). The energy consumption of the propagation procedure is effectively reduced, because there are nodes that are not affected for a given move of the sink. Every node (including the sink) in the pre-sojourn state transits to the **sojourn** state after a short time span controlled by a timer.

3.4 Minimizing Packet Losses

Although packet loss cannot be avoided during the sink moving period due to the lag of the failure detector, there are ways to mitigate the losses. Taking advantage of having a very short moving period (which we would not have if the sink moved continuously), the protocol tries to reduce the sending rate of the last-hop nodes, by asking them to buffer data packets using the interface queue (`QueuedSend` module) in MintRoute. We also add the following command to `QueueControl` interface, such that the routing module can access the interface

```
command void QueueControl.setAddrInQueue(uint16_t parent) {
    uint16_t i;
    if (!fQueueIdle)
        for (i = dequeue_next; i != enqueue_next;
             i = (i + 1) % MESSAGE_QUEUE_SIZE)
            msgqueue[i].address = parent;
}
```

queue to change the next-hop address of the buffered packets upon detecting a link failure.

Nodes can only buffer data packets; control packets should still be sent. However, if we simply picked up control packets from the interface queue and sent them, there would be gaps among the sequence numbers (remember that MintRoute applies the same sequence for both control and data packets). These gaps would mislead a neighbor node about a degradation in the link quality. Two solutions can be applied: (i) using separate sequences and queues for data and control packets or (ii) changing the sequence number within the queue, such that control packets sent have consecutive sequence numbers. In the short term, we adopt the second solution because it is easy to implement, but the first could be desirable in a long-run perspective.

4 Algorithm for Adaptively Controlled Mobility

According to our simulation results in Section 5, a mobility strategy that adapts to the network topology (for which no a priori knowledge exists) performs better than a static schedule. In this section, we describe the adaptive algorithm to control sink mobility. Our algorithm adaptively changes the sojourn time of the sink at each anchor point, according to the power consumption profile of the network. We derive the algorithm from a linear programming problem:

$$\text{Maximize lifetime } T = \sum_i T_i \tag{1}$$
$$\text{Constraints} \sum_i T_i \mathbb{P}_i \leq \mathbb{E}$$

where $\mathbb{P}_i$ and $\mathbb{E}$ are vectors that represent the power consumptions of each node (referred to as *P-profile* hereafter) when the sink sojourns at a certain anchor point i and the initial energy reserves of all nodes, respectively. This formulation basically means that we weigh, through the sojourn time T_i, the anchor points based on the corresponding P-profile $\mathbb{P}_i$s, in such a way that the $\mathbb{P}_i$s that complement each other are favored.

In practice, we propose the following 2-phase algorithm to approximate the above programming problem:

- **Phase I–Initialization**: The mobile sink visits the anchor points one by one and sojourns at each point for a short *sampling period.* During each sampling period, the sink collects the power consumption records from all nodes and builds a P-profile for that anchor point. At the end of this phase, the sink performs the programming (1) and drops an anchor point if its weight T_i is extremely low. It is not worth keeping such a point because its corresponding sojourn time is not long enough to amortize the routing overhead introduced by the sink mobility.
- **Phase II–Operation**: The mobile sink goes through the trajectory repetitively but only sojourns at those chosen anchor points. At a given point i, the sink again collects power consumption information and builds a profile $\mathbb{P}_i$. Based on the new profile and previous profiles for other chosen points, the programming (1) is re-solved to deduce T_i. The actual sojourn time is computed as $T_i^s = T_i/\delta$, where $\delta > 1$ is an integer. Applying the δ makes it possible for the sink to repeat the movement pattern several turns, which allows the sink to be more adaptive to the network dynamics.

We have the following remarks on the algorithm:

- The sink could have directly applied the results (i.e., T_is) of the first phase to the second phase if the routing topology were fixed. However, according to our experiences with real WSNs, the routing topology keeps evolving even though the nodes are static. As a result, the P-profiles obtained from the first phase can only be considered as estimations and should be updated if new profiles are available.
- If we make a discrete search over the whole surface covered by the network to obtain those anchor points, the time to finish Phase I could become comparable to the network lifetime; the algorithm would thus lose its adaptability (e.g., the sink might not even get a chance to enter Phase II). Alternatively, we can search over a “good” trajectory. A candidate of such a trajectory could be the periphery of the network, according to the theory of [9].

5 Simulations

We report two sets of simulations in this section. In one set, network nodes are located on point lattices; simulation results of this set represent outdoor WSNs in general. In another set, nodes form a ring; the simulations emulate our future field tests with an in-building WSN. All simulations are performed in TOSSIM.

5.1 Grid Networks

We arrange nodes on a point lattice of size 4×4, 5×5, and 7×7. For each network, we either (i) put the sink (node 0) at the network border (the midpoint of one side), or (ii) at the center, or (iii) let the sink move around the network periphery. There is a constant distance between any two consecutive anchor points; the sink pauses on an anchor point and moves in between two anchors according to the instruction from a Tython [16] code. The connectivity[4] of a node with other nodes is shown in Fig. 3. The transmission range is set to 1.2 times longer than

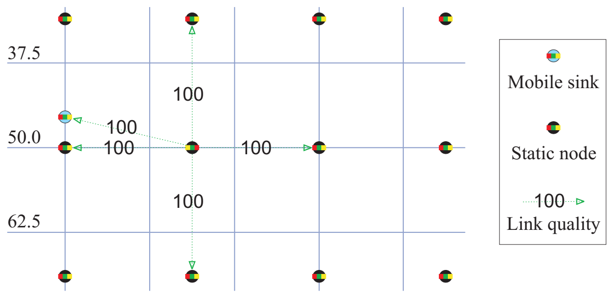


Fig. 3. Neighborhood graph in TinyViz [13]. The number beside a link states the link quality: 100 stands for a perfect link. The numbers on grid lines represent coordinates.

the distance between two neighbor nodes. Each node generates a data packet every 60 seconds. A control packet (route message) is sent every 120 seconds in the sojourn state and every 2 seconds (speed-up rate) in the pre-sojourn state. The s-beacon rate is one per second. The retransmission is disabled for all nodes if not stated otherwise. The sojourn time of non-adaptive mobility allows each

[4] We make use of the *fixed radius* model, although it is less realistic than the *empirical* one (we refer to [13] for the definitions of these models). The reason is that, given a set of geo-distributed nodes, applying the empirical model usually leads to small network diameter due to the occasional existence of *shortcuts*. A relatively large network diameter (up to 10 hops) is essential to fully exhibit the benefit of using a mobile sink, but increasing the network size to achieve larger diameter results in a simulation time of unreasonable duration (e.g., 100 hours). By using the fixed radius model, we simply assume that, for a certain node, only nodes within its *effective region* [15] are considered as its neighbors.

node to send 10 data packets (i.e., 600 seconds)[5], and the moving time is 10 seconds for the 49 nodes network and 20 seconds for the other two. The sink moves at a speed of 1 ft/s in the move state. The full simulation time is just long enough to let the sink go through one round of its trip; the simulation for a given network is repeated 10 times. For the measurement of energy consumptions[6], we use the number of (both control and data) packets that a node is involved to characterize the energy consumption. By doing this, we implicitly assume that (i) radio communication is the dominating energy consumer, (ii) sending and receiving a packet consumes the same amount of energy, and (iii) control and data packets are of the same size.

Non-adaptive Mobility The spatial distributions of energy consumptions for networks with 25 and 49 nodes are shown in Fig. 4. According to the lifetime definition in Section 2, the smaller the maximum energy consumption in a network, the longer the network lifetime will be. By comparing the two cases with a static sink and the case with a mobile sink, we make the following observations:

- The load-balancing effect of using a mobile sink is evident. The network with a mobile sink always lives longer than the network with a static sink at its border and no shorter than the network with a static sink at the center.
- In the network of 49 nodes, using a mobile sink is the best choice, irrespective of whether the overhearing at the MAC layer exists or not. However, overhearing does offset the benefits of using a mobile sink: the 100% improvement on the lifetime (comparing the network having a mobile sink with the one having a centered static center) is reduced to 50% if overhearing exists.
- In smaller networks of 16 and 25 nodes (only the latter case is shown in Fig. 4 due to their similarity), using a mobile sink is not necessarily helpful, because it does not improve the lifetime compared with using a centered static sink while increasing the accumulative energy consumption of the network.

A straightforward conclusion is that using a mobile sink is more beneficial in large networks. Since the function of the mobile sink is to disperse the traffic flows, the network should be large enough to provide nodes with a sufficient number of alternative routing paths. However, since locating a sink at the network center is not always practical[7], using a mobile sink does help to improve the lifetime in most networks.

[5] This duration is way shorter than what could be in a real deployment, where it might last for days or even weeks. Therefore, the performance of MobiRoute is expected to be better in practice, thanks to a longer amortization period.

[6] Since TOSSIM uses a MAC that never switches off its radio, tools such as PowerTOSSIM [17] always report a flat energy consumption pattern of a network no matter where the sink is located. In reality, motes equipped with B-MAC [12] do switch off their radio when there is no transmission going on.

[7] For habitat and environment monitoring, unobtrusive observation is key for studying natural phenomena [19]. Although nodes are small enough for this purpose, a sink (especially when it has to transmit the collected data out of the network area) can hardly makes itself invisible in the environment.

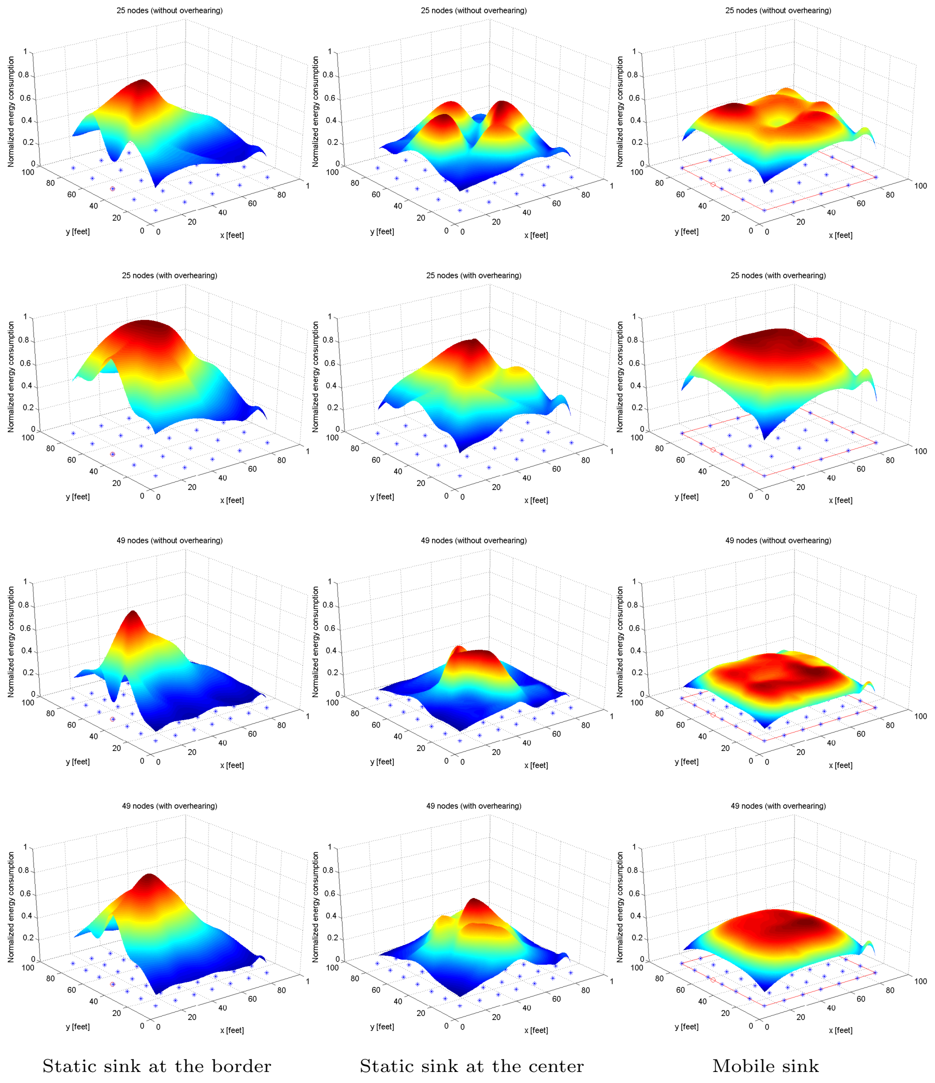


Fig. 4. Energy consumption of WSNs. Two networks with 25 and 49 nodes, respectively, are simulated. For each network, we either put the sink at the network border (the midpoint of one side), or at the center, or let the sink move around the network periphery. For each comparative case (i.e., one row in the figure), the energy consumptions are normalized to a common scale factor.

Another implication of our observations is that a MAC protocol free of overhearing is very important to improve the effectiveness of using a mobile sink. Unfortunately, the current MAC of motes (i.e., B-MAC [12]) suffers much from overhearing [19], and protocols with the potential to avoid overhearing (e.g., S-MAC [11]) do not necessarily have an overall performance better than B-MAC due to their burdensome synchronization schemes. So, we expect future technology to provide sensor nodes with overhearing-free MACs.

We plot the cumulative distribution functions of the packet delivery ratio in these two networks in Fig. 5. The comparisons are only made between a centered static sink and a mobile sink, because the ratios are quite similar for both networks with a static sink. The figures show that, without retransmission,

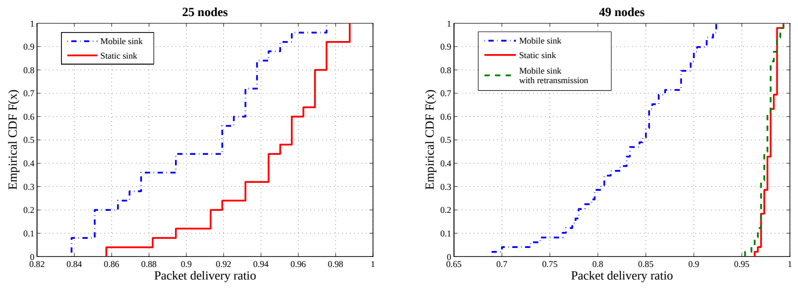


Fig. 5. Comparisons of packet delivery ratio

the packet delivery ratio is always lower in the case of a mobile sink, which is intuitive (see the reasons that we described in Section 3). Also, the difference between the two ratios increases with the network size. The reason is that using a mobile sink increases the worst-case routing path length (actually, a static sink located at one vertex of the network periphery achieves the same ratio). This is not a major problem, because we would expect a much higher reliability

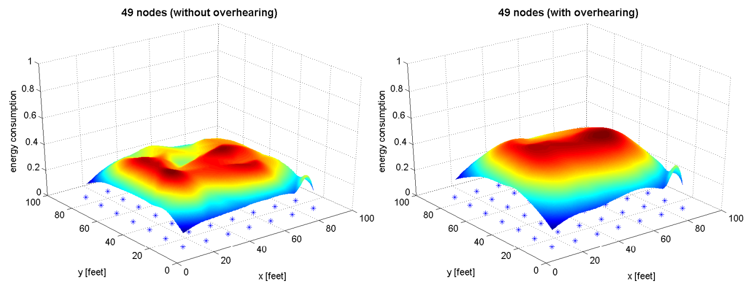
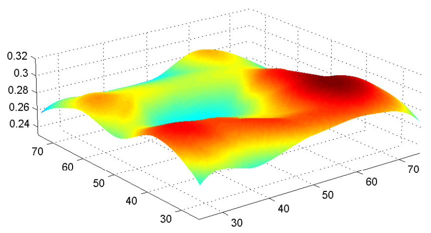


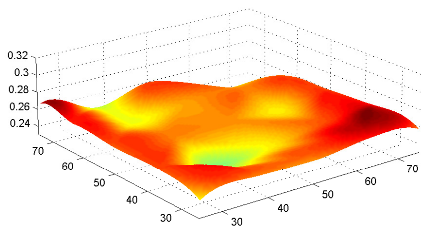
Fig. 6. Energy consumption of a WSN with a mobile sink and retransmission enabled. The scale factors take the same value as used for Fig. 4.

in reality, where a node typically sends data only every tens of minutes [19]. Actually, if we enable the retransmission, the packet delivery ratio in the case of a mobile sink can be as high as that in the case of a static sink, but at the cost of increased energy consumption (Fig. 6), whose maximum value is still low enough to justify the benefit of using a mobile sink.

Adaptive Mobility Zooming into the spatial distribution of energy consumption in the network with a mobile sink (as shown in Fig. 7 (a)), we observe that



(a) Non-adaptive mobility



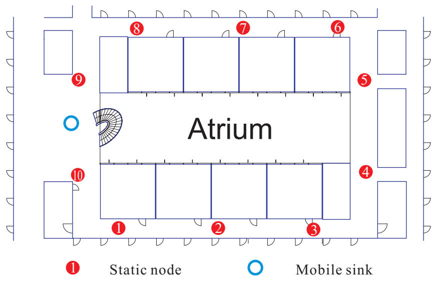
(b) Adaptive mobility

Fig. 7. Zooming in the distribution of energy consumption with a mobile sink.

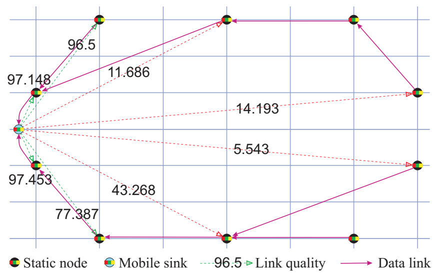
the load taken by nodes near the corner is heavier than that of other nodes. Applying the algorithm described in Section 4, we actually find that the sink should sojourn less time at those anchors near the corner. The resulting load, shown in Fig. 7 (b), is further balanced; which improves the network lifetime by about 10%. Note that the sink, in our simulations, only circles around the network twice: one in phase I and another in phase II (see Section 4); the network lifetime can be further improved with more rounds in phase II.

5.2 Ring Network

This section presents the simulation with a ring network. We use this simulation scenario to emulate a network deployed in our building, as shown in Fig. 8 (a). While a static sink[8] is located in-between nodes 9 and 10, a mobile sink moves



(a) The experimental network

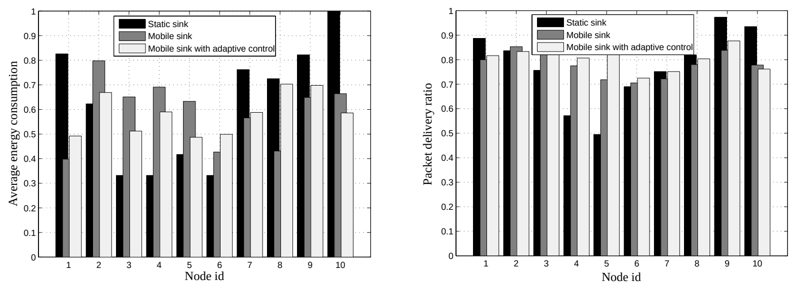


(b) Simulation scenario

Fig. 8. The plan of our network deployment (a) and the simulation scenario (b). Nodes are numbered the same way in (b) as in (a).

around the circle and sojourns in between two consecutive nodes. We use the *empirical* model [13] to characterize the connectivity in this set of simulations. As an example, the connectivity graph for the sink (node 0) is shown in Fig. 3 (b). Each node generates a data packet every 30 seconds. A control packet (route message) is sent every 60 seconds in the sojourn state and every 2 seconds (speed-up rate) in the pre-sojourn state. The s-beacon rate is one per second. The retransmission is disabled for all nodes. Each of the 10 simulations lasts for 17600 seconds and the sojourn time of non-adaptive mobility is 1760 seconds. The sink moves at a speed of 1 ft/s in the move state, and the moving time is 25 seconds. The measurement of energy consumptions is the same as for Section 5.1, and the overhearing is not taken into account.

We illustrate the simulation results with bar graphs in Fig. 9. As shown in Fig. 9 (a), the load balancing effect is already very evident by simply moving an uncontrolled sink, which improves the lifetime by 20%. Further improvement is achieved (an additional 15% of improvement on lifetime compared to the non-adaptive mobility) by controlling the mobile sink adaptively. The behavior in



(a) (b)

Fig. 9. Simulation results. (a) The energy consumptions are normalized by the largest energy consumption observed (i.e., node 10 in the case of a static sink). (b) The averaging effect arises also for the packet deliver ratio.

packet delivery, plotted in Fig. 9 (b), differs from that shown in Fig. 5; the averaging effect also arises due to the special network topology. In this specific scenario, the averaging effect makes a mobile sink beneficial not only to the network lifetime but also to the reliability, because nodes that are far away from the static sink perform poorly in terms of the reliability of packet delivery.

6 Discussion and Related Work

There are two main approaches, apart from our *mobile sink* approach, to use mobile devices for improving the lifetime of WSNs. One is the *mobile relay* ap-

[8] The atrium inside of our building prevents us from locating the sink at its optimum position (i.e. the center of the network). This indeed corroborates our claim in Section 5.1 that locating a sink at the network center is not always practical.

proach [5, 6, 8], where a mobile sink "picks up" data from nodes and transports the data back with mechanical movements. The other is the *mobile node*[9] approach [20]; the basic idea is that a few powerful mobile nodes can be deployed to replace different (heavily loaded) static nodes. In the latter approach, a mobile node inherits the responsibilities of a static node with which it is co-located, such that the static node can shut down for energy saving. In [20], the authors claim that, given a sufficient number of mobile nodes ($O(\sqrt{N})$ for an N-nodes network), their approach (with a static sink) can achieve a lifetime that is in the same order as our mobile sink approach.

All these approaches have their pros and cons. The mobile relay approach definitely achieves the largest improvement in lifetime, because static nodes take no or a very light forwarding load. However, only applications that tolerate large delays may choose this approach, because the resulting data delivery latency is significant. Our mobile sink approach achieves less network lifetime, but it does not sacrifice latency. In addition, the requirement for discrete mobility, as well as the simple mobility strategy (i.e., the network periphery), greatly facilitate the design of routing protocol. The mobile node approach tends to be even less demanding in terms of required facilities; the authors of [20] contend that, in hostile terrains, moving the sink is not always possible. However, it seems to us that moving several nodes could potentially increase the complexity of the routing protocol design and the overhead of running the resulting protocol. Considering that both mobile relay and mobile sink approaches have already been supported by practical routing protocols (see [8] and this paper), it would be interesting to see a practical routing protocol devised to support the mobile node approach.

7 Conclusion

In this paper, we have presented a routing protocol, MobiRoute, to support wireless sensor networks (WSNs) with a mobile sink. This is a follow-up of our previous work [9] where we theoretically prove that moving the sink can improve network lifetime without sacrificing data delivery latency. By insentively simulating MobiRoute with TOSSIM (in which real implementation codes are running), we have demonstrated the benefit of using a mobile sink rather than a static one. We have simulated both general networks with nodes located in point lattices and a special in-building network with nodes forming a ring. The results are very promising: a mobile sink, in most cases, improves the network lifetime with only a modestly degraded reliability in packet delivery.

We are in the process of performing full-scale field tests with the in-building network. We will also improve MobiRoute based on the experience obtained from our field tests.

[9] The authors also term their approach "mobile relay". In order to be consistent with the terminology used in [9] (where "mobile relay" was given to the first approach), we give another name to this approach.

References

1. Akyildiz, I., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless Sensor Networks: A Survey. Elsevier Computer Networks Journal **38** (2002) 393–422
2. Chang, J.H., Tassiulas, L.: Energy Conserving Routing in Wireless Ad-hoc Networks. In: Proc. of the 19th IEEE INFOCOM. (2000)
3. Sankar, A., Liu, Z.: Maximum Lifetime Routing in Wireless Ad-hoc Networks. In: Proc. of the 23rd IEEE INFOCOM. (2004)
4. Nash, S., Sofer, A.: Linear and Nonlinear Programming. McGraw Hill Companies, Inc. (1996)
5. Shah, R., Roy, S., Jain, S., Brunette, W.: Data MULEs: Modeling a Three-tier Architecutre for Sparse Sensor Networks. In: Proc. of the 1st IEEE SNPA. (2003)
6. Chakrabarti, A., Sabharwal, A., Aazhang, B.: Using Predictable Observer Mobility for Power Efficient Design of Sensor Networks. In: Proc. of the 2nd IEEE IPSN. (2003)
7. Gandham, S., Dawande, M., Prakash, R., Venkatesan, S.: Energy Efficient Schemes for Wireless Sensor Networks with Multiple Mobile Base Stations. In: Proc. of IEEE Globecom. (2003)
8. Kansal, A., Somasundara, A., Jea, D., Srivastava, M., Estrin, D.: Intelligent Fluid Infrastructure for Embedded Networks. In: Proc. of the 2nd ACM/USENIX MobiSys. (2004)
9. Luo, J., Hubaux, J.P.: Joint Mobility and Routing for Lifetime Elongation in Wireless Sensor Networks. In: Proc. of the 24th IEEE INFOCOM. (2005)
10. Grossglauser, M., Tse, D.: Mobility increases the capacity of ad hoc wireless networks. IEEE/ACM Trans. on Networking **10** (2002) 477–486
11. Ye, W., Heidemann, J., Estrin, D.: An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In: Proc. of the 21st IEEE INFOCOM. (2002)
12. Polastre, J., Hill, J., Culler, D.: Versatile Low Power Media Access for Wireless Sensor Networks. In: Proc. of the 2nd ACM SenSys. (2004)
13. Levis, P., Lee, N., Welsh, M., Culler, D.: TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. In: Proc. of the 1st ACM SenSys. (2003)
14. Blough, D., Santi, P.: Investigating Upper Bounds on Network Lifetime Extenstion for Cell-based Energy Conservation Techniques in Stationary Ad Hoc Networks. In: Proc. of the 8th ACM MobiCom. (2002)
15. Woo, A., Tong, T., Culler, D.: Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks. In: Proc. of the 1st ACM SenSys. (2003)
16. Demmer, M., Levis, P.: Tython: A Dynamic Simulation Environment for Sensor Networks. `http://www.tinyos.net/tinyos-1.x/doc/tython/tython.html`.
17. Shnayder, V., Hempstead, M., Chen, B., Allen, G., Welsh, M.: Simulating the Power Consumption of Large-Scale Sensor Network Applications. In: Proc. of the 2nd ACM SenSys. (2004)
18. MPR/MIB User's Manual. `http://www.xbow.com/Products/Wireless_Sensor_Networks.htm`.
19. Szewczyk, R., Mainwaring, A., Polastre, J., Anderson, J., Culler, D.: An Analysis of a Large Scale Habitat Monitoring Application. In: Proc. of the 2nd ACM SenSys. (2004)
20. Wang, W., Srinivasan, V., Chua, K.C.: Using Mobile Relays to Prolong the Lifetime of Wireless Sensor Networks. In: Proc. of the 11th ACM MobiCom. (2005)