# Data Salmon: A Greedy Mobile Basestation Protocol for Efficient Data Collection in Wireless Sensor Networks

Murat Demirbas[1], Onur Soysal[1], and Ali Şaman Tosun[2⋆]

[1] Dept. of Computer Science & Engineering, University at Buffalo, SUNY
{demirbas, osoysal}@cse.buffalo.edu
[2] Dept. of Computer Science, The University of Texas at San Antonio
tosun@cs.utsa.edu

**Abstract.** Our work addresses the spatiotemporally varying nature of data traffic in environmental monitoring and surveillance applications. By employing a network-controlled mobile basestation (MB), we present a simple energy-efficient data collection protocol for wireless sensor networks (WSNs). In contrast to the existing MB-based solutions where WSN nodes buffer data passively until visited by an MB, our protocol maintains an always-on multihop connectivity to the MB by means of an efficient distributed tracking mechanism. This allows the nodes to forward their data in a timely fashion, avoiding latencies due to long-term buffering. Our protocol progressively relocates the MB closer to the regions that produce higher data rates and reduces the average weighted multihop traffic, enabling energy savings. Using the convexity of the cost function, we prove that our local and greedy protocol is in fact optimal.

## 1 Introduction

A wireless sensor network (WSN) consists of potentially hundreds of sensor nodes and is deployed in an ad hoc manner for collecting data from a region of interest over a period of time [1, 2]. Even though the technology is new, WSNs received an enthusiastic reception in the science community as WSNs enable precise and fine-grain monitoring of a large region in real-time. Some examples of successful large-scale deployments of WSNs to date are in the context of ecology monitoring (monitoring of micro-climate forming in redwood forests [3]), habitat monitoring (monitoring of nesting behavior of seabirds [4]), and military surveillance (detection and classification of an intruder as a civilian, soldier, car, or SUV [5, 6]).

In traditional WSN deployments (including all of the above deployments), the data collection is achieved by using a multihop data forwarding mechanism toward a static basestation (SB), which has the computational power to store and process all the collected data. A major shortcoming of this approach is that it neglects the *spatiotemporal nature of data generation in the WSN*. That the WSN data generation rates are local both in time and space has been observed

⋆ Partially supported by Center for Infrastructure Assurance and Security at UTSA

| | Mobility | Buffering | Latency | Energy consumed |
|---|---|---|---|---|
| Data mules | random | long-term | high | low |
| Predictable MB | periodic | long-term | high | low |
| MES | self-controlled | long-term | medium | low |
| Data salmon | network-controlled | short-term | low | medium |

**Table 1.** Comparison of MB protocols

in several WSN deployments. Investigations of natural phenomena in forest environments have validated spatiotemporal distribution of solar illumination, temperature, and humidity [3, 7, 8]. This effect is especially prevalent in intrusion detection applications [5, 6].

As the observed environmental phenomena changes inevitably with time, the performance of data collection in the WSN suffer using the SB approach. In typical WSN applications most of the time the network remains idle, and when there is an interesting event (such as a rapid change in ambient features or detection of an intruder), a bursty generation of traffic occurs at a region of the network for some time. Fixing the location of the SB as the center or one corner of the network penalizes these bursts of data generation. Multihop relaying of this high data-rate traffic from the originating region toward the SB results in the depletion of energy at the relaying nodes for the duration of the traffic. Also collisions and message losses occur as this high data-rate traffic contends with itself over multiple hops. Clustering and aggregation techniques [9, 10] help to alleviate these contention and surge conditions, but they fail to address the root cause of the problem.

In order to address the drawbacks of the SB approach, there has been a flurry of work on employing a mobile basestation (MB) for data collection. The *data mules* [11] work exploit random movement of MBs to opportunistically collect data from a sparse WSN. Here, the nodes buffer all their data locally, and upload the data only when the MB arrives within direct communication distance. Although this approach is energy-efficient (in that the nodes do not engage in multihop data forwarding), the tradeoff is the very high latency and buffering costs. A similar approach is investigated in the context of predictable movement of the MB. Here sensors are assumed to know the trajectory of the MB and predict when the data transfer will occur accordingly. This work also shares similar drawbacks as the data mules work: since the data rate may be varying in time among the regions, buffer overflows may occur due to high data-rate traffic.

Mobile element scheduling (MES) work [12] considers controlled mobility of the MB in order to reduce latency and serve the varying data-rates in the WSNs effectively. The MES work shows that the problem of planning a path for the MB to visit the nodes before their buffers overflow is NP-complete. Although some heuristic based solutions are proposed to address this problem [12–14], these solutions ignore the problem of communicating the status of the spatiotemporally varying data-rates to the MB.

**Contributions.** Our work addresses the spatiotemporally varying nature of data traffic in environmental monitoring and surveillance applications. In contrast to previous MB-based solutions, we avoid indefinite buffering of data at

the sensor nodes and let them forward the data toward the MB to avoid any latencies. In order to reduce the energy-consumption due to multihop data forwarding, our MB protocol, namely the Data Salmon, progressively relocates the MB to minimize the average weighted-multihop distance from the data producing nodes to the MB. To this end, our protocol directs the MB closer to the regions that produce higher data rates so that most of the traffic arrives to the MB via a small number of hops. Intuitively speaking our MB always tends toward the center of mass of the network based on the data rate distribution.

Secondly, we prove that it suffices to design a local and greedy protocol to achieve an optimal relocation of the MB, minimizing the average weighted-multihop distance in the WSN. This proof involves showing that the cost function is convex and a local minima is a global minima. Our Data Salmon protocol exploits this result in that at each position in the network it decides on the next position via a simple greedy decision, using only the information available at that node. Our protocol relocates the MB toward the edge with the largest flow. [3] Another implication of such a greedy approach is that it is easy to parallelize the solution via divide and conquer technique: Adding more MBs to the network is easy since the MBs do not need to coordinate, yet each by optimizing its own gain implicitly cooperates to achieve a desirable global behavior. We present an extension of our Data Salmon protocol to multiple MBs along these lines.

Thirdly, our work demonstrates a synergistic cooperation of the MB and the underlying WSN for achieving efficient and effective data collection. In our protocol, the MB uses an underlying spanning tree structure to receive the data and to decide which direction to move on this backbone tree. In return when the MB moves along one edge of the tree, it updates the direction of the edge to point to its new location to ensure that a dynamic tree is always rooted at the MB. This way it is possible to keep the MB always reachable from the backbone tree structure, and the movement of MB also becomes relatively simple (by following one edge on the backbone tree). Although previous work assumed that the data-rates in the WSN is known and fixed [12], our protocol addresses this problem explicitly and discovers the current data-rates on-the-fly by means of this distributed dynamic tree structure.

Finally, we simulate our Data Salmon protocol using real WSN data (collected from an intrusion detection application) and some synthetic data. Via these simulation results, we compare the improvements gained by using Data Salmon over using SB under various configurations.

**Applications.** Since the Data Salmon protocol action for the MB is simple and the MB is virtually controlled by the network, our protocol does not require a fully-autonomous robot to implement the MB. Thus, it is practical to implement and deploy Data Salmon in real-world environmental monitoring and surveillance applications using semi-autonomous MBs. A suspended cableway infrastructure for MB mobility would provide a suitable framework for the deployment of the Data Salmon protocol. For example, the Networked Infomechan-

[3] Due to this greedy behavior to move toward the largest flow, we name our protocol after the Salmon fish which swim upstream to lay eggs.

ical Systems (NIMS) architecture [7] successfully avoids surface-based obstacles found in natural environments by employing a horizontally mobile node suspended via an aerial cable, and achieves adaptive sampling and effective solar radiation mapping in microclimate monitoring applications. Another example of such a system is the SkyCam platform [15], which is suitable for intrusion detection and surveillance applications.

In our model, we have not included the energy required for moving the MB. The reason behind our willingness to generously tradeoff the energy required for relocating the MB with the energy gain in data collection is that it is much easier to replenish and maintain the batteries of one MB than those of the sensor nodes in the entire network. As it was observed through the NIMS deployment [16], by using a solar panel attached to the mobile node it is possible to harvest an average of 250 Watt hours of energy per day and sustain the mobility of the node. Such an alternative energy source creates a virtual flow of energy into the system, hence, the WSN lifetime is also elongated. Another benefit a network controlled MB provides is the increased traffic capacity and network throughput as mentioned in [17].

## 2 Model

We consider a dense, connected, multihop WSN. The sensor nodes are static after the initial deployment. There is a distinguished MB in the network whose current location (the node it resides on) is denoted by $m$.

We assume that a spanning tree structure is overlaid over the WSN during the network initialization phase. To reduce the height of the tree, the tree root (denoted as *root*) may be a node in the center of the network. By using a flooding protocol initiated by the *root* it is easy to construct this backbone tree structure [5, 6]. We denote the set of neighbors of node $i$ on the tree as $N(i)$, and use $d(i, m)$ to denote the hop distance over the tree structure between a node $i$ and the MB at node $m$. We denote the data rate generated by a node $i$ at a given time as $w_i$. For a node $m$ we define $M(m)$, the cost of forwarding all the data to $m$ from the entire network, as $M(m) = \sum_i w_i * d(i, m)$. Then the problem of finding the optimal location for the MB reduces to finding a node $m^*$ with minimal $M(m)$.

## 3 The Data Salmon Protocol

After discussing how we maintain a dynamic tree rooted at the MB, we give our greedy MB relocation protocol and prove its optimality.

### 3.1 The Dynamic Tree Maintenance Protocol

Keeping the MB always reachable from the backbone tree structure is essential to guarantee always-on data forwarding to the MB. In order to maintain a dynamic

tree that is always rooted at the MB over the static backbone tree, we adopt the distributed arrow protocol [18].

After the backbone tree structure is set up as discussed in the Model section, we assume that the tree edges all point to the MB initially. As the MB moves over one of the tree edges, the arrow protocol prescribes flipping the direction of the edge. This way the tree is always rooted at the MB. By locally updating a tree edge, a dynamic tree rooted at MB can be thus maintained over the static backbone tree.

Of course, embedding a tree constrains how nodes can forward the traffic to the MB. For example, shortest path forwarding may not be achievable for some nodes as they are constrained to follow the tree while forwarding data to the MB. However using a backbone tree for forwarding of the traffic reduces the tracking cost of the MB drastically: In our scheme as the MB moves only one edge needs to be updated. Had we not used a tree backbone for data forwarding toward the MB, the tracking of the MB would incur an expensive (nonlocal) communication cost for updating the tracking structure as the MB relocates. Investigating update-efficient and local tracking structures is an active topic of research, and we give some pointers to this work in Section 5.

### 3.2 The Greedy Data Salmon Protocol

Our Data Salmon protocol for the MB runs on top of the dynamic tree structure, and uses the incoming data rates from neighboring nodes for deciding which neighbor to move the MB next. For each neighbor $i$ of the current node $m$, we denote the forwarded data rate from $i$ with $\varepsilon_i$. Note that, $\varepsilon_i$ corresponds to the cumulative weights of all nodes in the subtree rooted at $i$. We denote the total data rate in the WSN with $\varepsilon$, which is calculated at $m$ as $(\sum_{i \in N(m)} \varepsilon_i) \; + \; w_m$.

To minimize the cost function $M$, it is natural for the MB to move toward a neighbor $i$ with a lower cost function $M(i)$. We prove in Theorem 1 that in fact such a neighbor $i$ is unique since the $\varepsilon_i$ is at least more than half of the total data rate $\varepsilon$ in the WSN if and only if $M(i) < M(m)$.

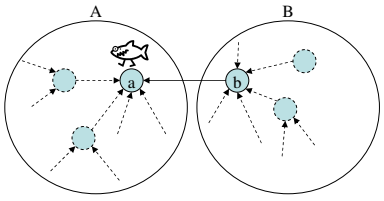



**Fig. 1.** Conceptual representation for proof of Theorem 1

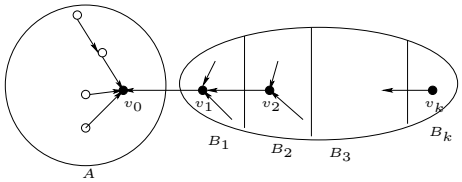



**Fig. 2.** Visual Representation of Theorem 2

**Theorem 1.** *Let MB be at node $v_a$, and $v_b \in N(v_a)$, then* $M(v_b) < M(v_a) \iff \varepsilon_{v_b} > \frac{\varepsilon}{2}$.

**Proof:** Consider Figure 1, where MB is at $v_a$. If MB is moved from $v_a$ to $v_b$, since this graph is a tree, all data generated at nodes in set $A$ has to be forwarded through edge $(v_b, v_a)$ toward $v_b$. That is, the distance to the MB increases by 1 for all nodes in set $A$, and the distance decreases by 1 for all nodes in set $B$. Thus, we can write the following:

$$M(v_b) = M(v_a) + \varepsilon_A - \varepsilon_B \tag{1}$$

Since $A \cup B$ contains all the nodes, the following also holds:

$$\varepsilon_A + \varepsilon_B = \varepsilon \tag{2}$$

**Case ($\Longrightarrow$):** Using the assumption $M(v_b) < M(v_a)$, from (1) we can write, $\varepsilon_A - \varepsilon_B = M(v_b) - M(v_a) < 0$. So subtracting (2) from this term we can write $-2\varepsilon_B < -\varepsilon$ and thus conclude $\varepsilon_B > \frac{\varepsilon}{2}$.

**Case ($\Longleftarrow$):** Using the assumption $\varepsilon_{v_b} > \frac{\varepsilon}{2}$, from equation (2) we have $\varepsilon_A < \frac{\varepsilon}{2} < \varepsilon_B$. This entails $M(v_b) - M(v_a) = \varepsilon_A - \varepsilon_B < 0$. So $M(v_b)$ has smaller cost. □

**Algorithm 1** MB control action at $m$

1: $\varepsilon \leftarrow (\sum_{i \in N(m)} \varepsilon_i) \; + \; w_m$
2: **if** $\exists i \in N(m) : \varepsilon_i > \frac{\varepsilon}{2}$ **then**
3: move to $i$
4: **end if**
5: // else stay at $m$, since $m$ is optimal

Therefore, the MB control action at node $m$ is given as in Algorithm 1. We prove the optimality of this protocol in the next section.

### 3.3 Proof of Optimality

Our optimality discussion depends on some properties of the cost function over the backbone tree. We first show that the cost function is convex in Theorem 2, and that the rate of increase of the cost function is non-decreasing in Theorem 3. We use these two properties to conclude that the Data Salmon protocol is indeed optimal over the backbone tree.

**Theorem 2.** *Let $v_0$ be an optimal location for MB. Consider a path $v_0$, $v_1$,..., $v_k$ over the backbone tree. $M(v_0) \leq M(v_1) \leq \ldots \leq M(v_k)$ holds for the path.*

**Proof:** Consider Figure 2. Since $v_0$ is an optimal location of MB, $M(v_0) \leq M(v_1)$, and this proves the first inequality. If the MB is moved from $v_0$ to $v_1$, the distance increases by 1 for all nodes in set $A$, and the distance decreases by 1 for all nodes in the set $B = \cup_{i=1}^{k} B_i$. So,

$$M(v_1) = M(v_0) + \sum_{i \in A} w_i - \sum_{i \in B} w_i \tag{3}$$

Since we have $M(v_0) \leq M(v_1)$ we get

$$\sum_{i \in A} w_i - \sum_{i \in B} w_i \geq 0. \tag{4}$$

If the MB is moved from $v_1$ to $v_2$, similarly, the distance increases by 1 for all nodes in set $A \cup B_1$ and the distance decreases by 1 for all nodes in the set $B - B_1$. We can write $M(v_2)$ using $M(v_1)$ as follows:

$$M(v_2) = M(v_1) + \sum_{i \in A \cup B_1} w_i - \sum_{i \in B - B_1} w_i \tag{5}$$

This can be rewritten as

$$M(v_2) = M(v_1) + \sum_{i \in A} w_i + \sum_{i \in B_1} w_i - \sum_{i \in B} w_i + \sum_{i \in B_1} w_i$$

$$M(v_2) = M(v_1) + \sum_{i \in A} w_i - \sum_{i \in B} w_i + 2 \sum_{i \in B_1} w_i$$

Since all weights are non-negative, the last term is non-negative. First two terms are shown to be non-negative in equation 4, so we get $M(v_2) \geq M(v_1)$. This can be generalized to the following using the same approach:

$$M(v_k) = M(v_{k-1}) + \sum_{i \in A} w_i - \sum_{i \in B} w_i + 2 \sum_{j=1}^{k-1} \sum_{i \in B_j} w_i \tag{6}$$

Thus, we have $M(v_0) \leq M(v_1) \leq \ldots \leq M(v_k)$ $\square$

Since we use this result later we introduce $\varepsilon_S$, which corresponds to the sum of weights of all members of set $S$, as $\varepsilon_S = \sum_{i \in S} w_i$. Hence, equation (6) can be rewritten as:

$$M(v_k) = M(v_{k-1}) + \varepsilon_A - \varepsilon_B + 2 \sum_{j=1}^{k-1} \varepsilon_{B_j} \tag{7}$$

Theorem 2 shows that the cost function is convex, but in order to guarantee that there are no oscillations in the MB control protocol we need to show that the rate of increase is also non-decreasing.

**Theorem 3.** *Let $v_0$ be the optimal location of MB. Over the backbone tree consider a path $v_0, v_1, \ldots, v_a, \ldots, v_b, \ldots, v_k$, where $0 < a < b \leq k$.*
*$M(v_a) - M(v_{a-1}) \leq M(v_b) - M(v_{b-1})$ holds for the path.*

**Proof:** By using equation (7), we get the following:

$$M(v_a) - M(v_{a-1}) = \varepsilon_A - \varepsilon_B + 2 \sum_{j=1}^{a-1} \varepsilon_{B_j}$$

$$M(v_b) - M(v_{b-1}) = \varepsilon_A - \varepsilon_B + 2 \sum_{j=1}^{b-1} \varepsilon_{B_j} \tag{8}$$

$$[M(v_{b+1}) - M(v_b)] - [M(v_{a+1}) - M(v_a)] = 2 \sum_{j=1}^{b-1} \varepsilon_{B_j} - 2 \sum_{j=1}^{a-1} \varepsilon_{B_j} \tag{9}$$

Since $b > a$, and all weights are non-negative, we can rewrite above as:

$$[M(v_{b+1}) - M(v_b)] - [M(v_{a+1}) - M(v_a)] = 2\sum_{j=a}^{b-1} \varepsilon_{B_j} \geq 0$$

□

As a corollary to these theorems, we observe that when the data rates are stable for a sufficient enough period, the MB progressively relocates to the optimal location in the WSN. Our corollary follows by using $M(m)$ as the variant function. From Theorem 2, we know that there are no local optimum points, and $M(m)$ is non-increasing toward the direction of the optimal location. Furthermore, Theorem 3 states that equality among $M(m)$ values is only possible between optimal nodes, so at any suboptimal node we are guaranteed to have a neighbor with lower cost. The decrease of $M(m)$ is bounded below by $M(m^*)$, so the MB eventually reaches and comes to a rest at an optimal location $m^*$.

## 4 Simulation Results

In order to evaluate the performance of our protocol, we use real-world WSN deployment data from the "Catch Me if You Can" project [19]. This project implements a multiple-pursuer, multiple-evader tracking application by utilizing the WSN to help the pursuers in protecting an asset from the evaders. Figure 3 shows the topology of the 60 nodes deployed for this project. The distance between any neighboring nodes in the topology is 10 meters.

The *Catch Me if You Can* project collected data sets for over 50 experiments. Each data set contains onsets and offsets of detection for the nodes: during these detection periods, the sensor nodes generate detection data. In order to simulate our Data Salmon protocol for collecting the generated detection data, we overlay a randomly generated backbone tree over the WSN as shown in Figure 3. This way, we calculate the approximate energy consumption for the SB and MB approaches using the durations of detections and the distances on the backbone tree.

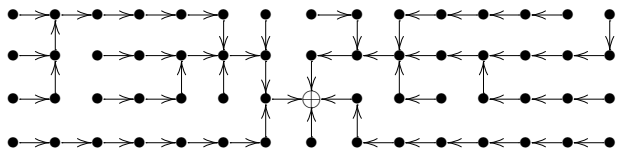
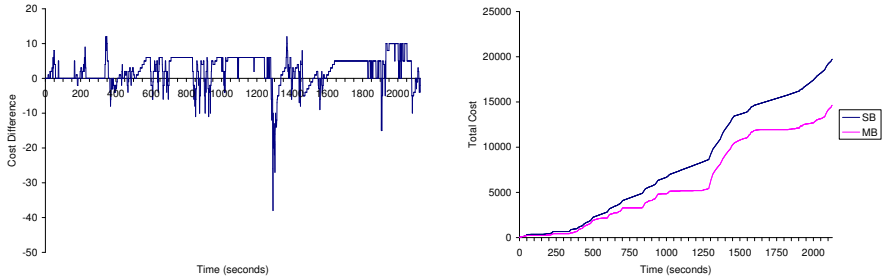


**Fig. 3.** The topology of sensors in *Catch Me if You Can* experiments. Dots denote the sensors. Embedded random backbone tree on the topology is shown by arrows and the *root* is indicated with a plus

The introduced locomotion model for the MB is a high-level abstraction of the mobile platform details. We assume that the MB moves following the tree edges, with constant speed. The MB only makes decisions at nodes, so it can not change direction during transitions between nodes.

We developed a Java application named *SalmonSim*[4] to interpret and emulate the data sets from the *Catch Me if You Can* experiments. The Java application uses constant time steps to measure the performance. During these time steps, the MB is simulated and the events from experiment logs are emulated. The Java application also provides a user interface to visualize the progress of the MB running the Data Salmon protocol. Using this simulator, we compare the performance of the SB positioned in the *root* of the tree with the MB using the Data Salmon protocol. For the comparisons, we use the same cost metric defined in the Section 3.




(a) (b)

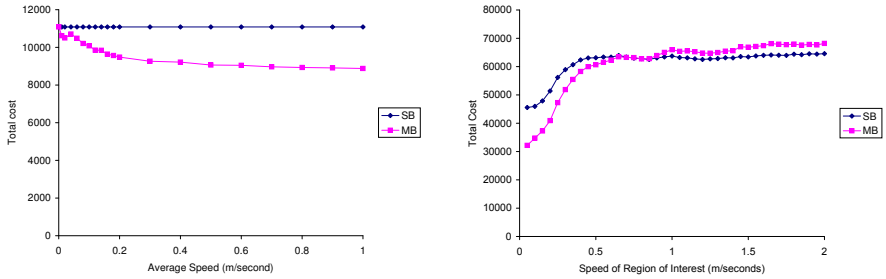
**Fig. 4.** (a) Difference between costs of SB and MB at any given time in a reference data set. (b) Total costs of SB and MB in a reference data set

We first compare the performance of the Data Salmon protocol with the SB using data from a reference data set. Figure 4(a) shows the instantaneous cost difference between the SB and the MB approaches. In the figure, the areas below the $y = 0$ baseline show that the MB may become disadvantageous (albeit, briefly) due to some abrupt changes in data rate. Since the cost difference stay above the $y = 0$ baseline most of the time, we observe that the cost of MB is less than that of SB. The cumulative of these differences, which gives us the total energy costs of each approach, are graphed in Figure 4(b).

Secondly, we investigate the effect of the MB speed. For chosen speed values the average of total cost for all data sets is shown in Figure 5(a). This graph shows that even with low speeds the MB approach can outperform the SB.

Since the emulation dataset does not lend itself well for controlling the data generation, we devised a second set of experiments using a synthetic data set.

[4] An applet version of the simulator is available in `http://www.cse.buffalo.edu/~osoysal/salmonSim/`.




(a) (b)

**Fig. 5.** (a) Total cost with respect to varying MB speed (b) Total cost with respect to varying region of interest speed

We chose the value of a normal distribution function, to represent the region of interest(ROI). The nodes have a threshold value to determine whether they should send messages or not depending on their interest level. Interest level of nodes correspond to the value of a normal distribution function at their position. We simulate the change in region of interest by moving the mean of a normal distribution function randomly. We start from a random point and choose another random point for mean. The mean is moved at a constant speed toward this random point until it reaches there. Then we choose another random point and repeat this process until the end of simulations.

We first investigate the speed of ROI by fixing the speed of MB to 0.4 m/second and varying the speed of ROI to obtain Figure 5(b). The graph shows that MB performs better than SB when the ROI moves up to two times faster than MB. After this point, MB cannot keep up with the sudden changes in ROI and falls beyond the SB case. Still the difference in performance is much less than the cases where ROI moves slowly.

We also replicated the experiments in Figure 4(a), Figure 4(b) and Figure 5(a) and obtained similar results. Synthetic results are more regular than regular data set which can be explained by the effect of the normal function used in modeling activation.

The Data Salmon protocol banks on the spatio-temporal nature of the data in WSNs. Through our experiments, we have seen that even for the rare cases this locality assumption does not hold, our protocol performs at least as good as the SB approach most of the time. For example, if the data rate is uniform throughout the network, our Data Salmon protocol fixes the location of the MB in an optimal location in the center of the WSN and acts more like an SB approach. The Data Salmon performs worse than the SB approach only in extremely pathological cases, where an adversary lures the MB toward one corner of the network only to follow it up with a large but short-lived surge of data from the opposite corner.

## 5 Discussion

**Fault-tolerance.** The backbone tree structure we use does not provide any redundancy to the face of node failures. When one node goes down, the result is a partitioned network. Fortunately, there has been a lot of work on self-stabilizing tree maintenance protocols [20–22] that enable the tree to recover itself upon node failure or corruption of the pointer structure at the nodes. (This is, of course, provided that the network is not physically partitioned.)

After the backbone tree is fixed as discussed above, we also need to consider the recovery of the distributed-arrow protocol. In [23], it is shown that by adding some self-stabilizing actions, it is possible to achieve self-stabilization of the distributed arrow protocol in a local and efficient way. Finally, when the underlying static tree and the distributed-arrow protocol stabilizes, the Data Salmon protocol stabilize trivially by virtue of being stateless.

Another issue for fault-tolerance is the collision-free collection of data packets. Since the tree structure is commonly used for data collection, there has been several work on collision-avoidance protocols for tree structures [24, 25].

**Load-balancing.** The static backbone tree imposes a strain on the static *root* and core of the tree as there is always considerable amount of traffic routed through the core toward the MB. Improving load-balancing in the network and reducing the hot-spot in the core would help elongate the network lifetime. Existing work in reducing uneven energy consumption in WSNs by using a MB [26,27] show that the optimum movement strategy for the MB is to follow the periphery of the network when the deployment area is circular. However, since these work assume uniform data generation by the sensor nodes every time unit, and reducing the average weighted multihop in the face of varying traffic is not a goal, these work are inapplicable in our context.

Modifying the backbone tree as the MB relocates may help reduce hot-spots in the tree. It is important to keep such modifications to be as localized around the MB as possible in order not to introduce excessive communication, hence, excessive energy-consumption into the WSN. A relatively local tree reconfiguration algorithm for bounded-length (4-5 hops) trees is presented in [28]. However, local reconfigurations alone are insufficient for maintaining a globally desirable tracking tree structure, and in the worst case local modifications may—over time—result in pathological cases where the height of the tree can be several orders of magnitude larger than the diameter of the network.

Relaxing the backbone tree structure by replacing it with a more permissive and load-balanced topology, such as a grid topology, may alleviate the hot-spot issue, as this allows multiple forwarding paths between any two points in the structure. Unfortunately, such a replacement introduces the problem of efficient tracking of the MB over the structure. Except for simple structures, such as a linear topology or a tree structure—as in our case—, designing update-efficient and local tracking protocols is a challenging problem. A tracking protocol for grid topology is investigated in [29] and several tracking protocols for more general network topologies are proposed in the literature [21,30]. However, when adopt-

ing such an approach, it is unclear whether the overhead involved in tracking would be commensurate with the gains achieved from using a MB.

**Multiple MB extension.** An implication of our greedy protocol is that it is easy to parallelize the solution via divide and conquer: Adding more MBs to the network is easy since the MBs do not need to coordinate, yet each by optimizing its own gain implicitly cooperates to achieve a desirable global behavior. As a demonstration of this claim, we present a simple scheme to extend our Data Salmon protocol to support multiple MBs. This scheme is based on the observation that when there are multiple MBs on the backbone tree, the arrow protocol maintains a dynamic directed acyclic graph (DAG) structure with multiple sinks instead of a tree structure with one root. A DAG structure implies that some nodes in the backbone tree now have multiple outgoing edges. Our modification, then, is to divide the incoming traffic at a node in an equal manner among the outgoing edges of the node. The MBs decide on their relocation in a local, greedy manner as before and, as before, an edge direction is reversed when a MB traverses the edge. This simple protocol leaves it solely to the discretion of the MBs to sort out how to share the network traffic and is not optimal. Devising optimal solutions for the multiple MB case is part of our ongoing work.

## 6 Concluding Remarks

We presented a simple, low-latency, and energy-efficient protocol for data collection in WSNs using a network controlled MB. In contrast to the existing MB-based solutions where WSN nodes buffer data passively until visited by an MB, our protocol overlays a spanning backbone tree and maintains an always-on multihop connectivity to the MB by employing the distributed-arrow tracking protocol on top of this tree. This enables the nodes to forward their data to the MB anytime, in a timely, and efficient fashion avoiding latencies due to long-term buffering. Our protocol achieves energy-efficiency for the WSN by greedily relocating the MB toward the direction of the tree that produce higher data rates and, hence, reducing the average weighted multihop traffic. Using the convexity of the cost function in this problem, we were able to prove that our local greedy protocol also optimizes the network-wide energy-efficiency metrics. An implication of this local, greedy, and optimal protocol is that it is easy to parallelize the data collection by adding more MBs to the WSN.

Devising and proving optimal solutions for the multiple MB case is part of our ongoing work. In future work we will focus on relaxing the underlying static backbone tree structure by replacing it with a less restrictive variant, such as a grid structure. Another extension we will pursue for alleviating the hot-spots problem is the inclusion of the energy constraints and the remaining lifetime of nodes (in addition to the data rates) for the calculation of the cost function.

## References


1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine **38** (2002) 393–422
2. Estrin, D., Govindan, R., Heidemann, J.S., Kumar, S.: Next century challenges: Scalable coordination in sensor networks. In: Mobile Computing and Networking. (1999) 263–270
3. Tolle, G., Polastre, J., Szewczyk, R., Turner, N., Tu, K., Buonadonna, P., Burgess, S., Gay, D., Hong, W., Dawson, T., Culler, D.: A macroscope in the redwoods. In: Proceedings of the Third ACM Conference on Embedded Networked Sensor Systems. (2005) 51–63
4. Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D., Anderson, J.: Wireless sensor networks for habitat monitoring. In: ACM Int. Workshop on Wireless Sensor Networks and Applications. (2002) 88 – 97
5. Arora, A., et. al.: A line in the sand: A wireless sensor network for target detection, classification, and tracking. Computer Networks (Elsevier) **46**(5) (2004) 605–634
6. Arora, A., et. al.: Exscal: Elements of an extreme scale wireless sensor network. In: 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications. (2005) 102–108
7. Batalin, M., Rahimi, M., Yu, Y., Liu, D., Kansal, A., Sukhatme, G., Kaiser, W., Hansen, M., Pottie, G., Srivastava, M., Estrin, D.: Call and response: experiments in sampling the environment. In: Proceedings of the 2nd international conference on Embedded networked sensor systems. (2004) 25–38
8. Yu, Y., Ganesan, D., Girod, L., Estrin, D., Govindan, R.: Synthetic data generation to support irregular sampling in sensor networks. In: Geo Sensor Networks, Taylor and Francis Publishers (Oct 2003)
9. Pattem, S., Krishnamachari, B., Govindan, R.: The impact of spatial correlation on routing with compression in wireless sensor networks. In: Proceedings of the third int. symposium on Information processing in sensor networks. (2004) 28–35
10. Krishnamachari, B., Estrin, D., Wicker, S.B.: The impact of data aggregation in wireless sensor networks. In: Proceedings of the 22nd International Conference on Distributed Computing Systems. (2002) 575–578
11. Shah, R.C., Roy, S., Jain, S., Brunette, W.: Data mules: modeling a three-tier architecture for sparse sensor networks. In: Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications. (2003) 30–41
12. Somasundara, A., Ramamoorthy, A., Srivastava, M.: Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines. In: Proceedings of the 25th IEEE International Real-Time Systems Symposium. (2004) 296–305
13. Gu, Y., Bozdag, D., Ekici, E., Ozguner, F., Lee, C.: Partitioning based mobile element scheduling in wireless sensor networks. In: IEEE SECON. (2005) 386–395
14. Zhao, W., Ammar, M.: Message ferrying: Proactive routing in highly-partitioned wireless ad hoc networks. In: Proceedings of the The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems. (2003) 308– 314
15. Cone, L.L.: Skycam: An aerial robotic camera system. Byte **10** (1985) 122–132
16. Pon, R., Batalin, M., Gordon, J., Kansal, A., Liu, D., Rahimi, M., Shirachi, L., Yu, Y., Hansen, M., Kaiser, W., Srivastava, M., Sukhatme, G., Estrin, D.: Networked infomechanical systems: a mobile embedded networked sensor platform. In: Information Processing in Sensor Networks. (2005) 376–381

17. Kansal, A., Rahimi, M., Estrin, D., Kaiser, W.J., Pottie, G., Srivastava, M.: Controlled mobility for sustainable wireless sensor networks. In: Sensor and Ad Hoc Communications and Networks. (2004) 1–6
18. Demmer, M.J., Herlihy, M.: The arrow distributed directory protocol. In: Proceedings of the 12th International Symposium on Distributed Computing. (1998) 119–133
19. Cao, H., Ertin, E., Kulathumani, V., Sridharan, M., Arora, A.: Differential games in large-scale sensor-actuator networks. In: Proceedings of the fifth international conference on Information processing in sensor networks. (2006) 77–84
20. Dolev, S.: Self-Stabilization. MIT Press (2000)
21. Demirbas, M., Arora, A., Gouda, M.: Pursuer-evader tracking in sensor networks. Sensor Network Operations, IEEE Press (2006)
22. Chen, N., Huang, S.: A self-stabilizing algorithm for constructing spanning trees. Information Processing Letters (IPL) **39** (1991) 147–151
23. Herlihy, M., Tirthapura, S.: Self-stabilizing distributed queueing. In: Proceedings of 15th International Symposium on Distributed Computing. (oct 2001) 209–219
24. Woo, A., Culler, D.E.: A transmission control scheme for media access in sensor networks. In: Proceedings of the 7th annual international conference on Mobile computing and networking. (2001) 221–235
25. Kulkarni, S.S., Arumugam, M.: Ss-tdma: A self-stabilizing mac for sensor networks. In: Sensor Network Operations, IEEE Press. (2005)
26. Gandham, S., Dawande, M., Prakash, R., Venkatesan, S.: Energy-efficient schemes for wireless sensor networks with multiple mobile base stations. In: Proceedings of IEEE GLOBECOM. (2003) 377– 381
27. Wang, Z., Basagni, S., Melachrinoudis, E., Petrioli, C.: Exploiting sink mobility for maximizing sensor networks lifetime. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences. (2005) 287a– 287a
28. Zhang, W., Cao, G.: Dctc: Dynamic convoy tree-based collaboration for target tracking in sensor networks. IEEE Transactions on Wireless Communication **3**(5) (2004) 1689–1701
29. Ye, F., Luo, H., Cheng, J., Lu, S., Zhang, L.: A two-tier data dissemination model for large-scale wireless sensor networks. In: Proceedings of the 8th annual international conference on Mobile computing and networking. (2002) 148–159
30. Demirbas, M., Arora, A., Nolte, T., Lynch, N.: A hierarchy-based fault-local stabilizing algorithm for tracking in sensor networks. In: 8th International Conference on Principles of Distributed Systems (OPODIS). (2004) 299–315