Mobile Relay Configuration in Data-intensive Wireless Sensor Networks

Fatme El-Moukaddem; Eric Torng; Guoliang Xing; Sandeep Kulkarni
Department of Computer Science and Engineering
Michigan State University
East Lansing, MI, U.S.A.
{*elmoukad, torng, glxing, sandeep*}*@cse.msu.edu*

Abstract

Recently, Wireless Sensor Networks (WSNs) have become increasingly available for data-intensive applications such as micro-climate monitoring, precision agriculture, and audio/video surveillance. A key challenge faced by data-intensive WSNs is to transmit the sheer amount of data generated within an application's lifetime to the base station despite the fact that sensor nodes have limited power supplies such as batteries or small solar panels. In this paper, we propose to use low-cost disposable mobile relays to reduce the energy consumption of data-intensive WSNs. Different from previous work, our approach does not require complex motion planning of mobile nodes, and hence can be implemented on a number of low-cost mobile sensor platforms. Moreover, we integrate the energy consumption due to both mobility and wireless transmissions into a holistic optimization framework. The optimal relay configuration is shown to depend on both the positions of nodes and the amount of data to be sent. We develop two algorithms that iteratively refine the configuration of mobile relays and converge to the optimal solution. These algorithms have efficient distributed implementations that do not require explicit synchronization. Our simulation results based on realistic energy models obtained from existing mobile and static sensor platforms show that our algorithms significantly outperform the best existing solutions.

1. Introduction

Recent years have seen the deployments of WSNs in a variety of *data-intensive* applications including micro-climate and habitat monitoring [1], precision agriculture, and audio/video surveillance [2]. It is shown in [3] that a moderate-size WSN can gather up to 1 Gb/year from a biological habitat. Due to the limited storage capacity of sensor nodes, most data must be transmitted to the base station for archiving and analysis. However, sensor nodes must operate on limited power supplies such as batteries or small solar panels. Therefore, a key challenge faced by data-intensive WSNs is to minimize the energy consumption of sensor nodes such that the sheer amount of data generated within the lifetime of the application can be transmitted to the base station.

Recent work showed that the energy cost of WSNs can be significantly reduced by utilizing the mobility of nodes. Several different approaches have been proposed. A robotic unit may move around the network and collect data from static nodes through one-hop or multi-hop transmissions [4]–[8]. The mobile node may serve as the base station or a "data mule" that transports data between static nodes and the base station [9]–[11]. Mobile nodes may also be used as *relays* [12] that forward the data from source nodes to the base station. Several movement strategies for mobile relays have been studied in [12], [13].

Although the effectiveness of mobility in energy conservation is demonstrated by previous studies, the following key issues have not been addressed collectively. First, the movement cost of mobile nodes is not accounted for in the total network energy consumption. Instead, mobile nodes are often assumed to have replenishable energy supplies. For instance, a mobile node may periodically recharge its battery at a fixed charging dock [7]. However, energy replenishment is not always feasible due to the constraints of the physical environment. Second, complex motion planning of mobile nodes is often assumed in existing solutions which introduces significant design complexity and manufacturing costs. In [7], [8], [14], [15], mobile nodes need to compute optimal motion paths and continuously change their orientation and/or speed of movement. Such capabilities are usually not supported by existing low-cost mobile sensor platforms. For instance, Robomote [16] nodes are designed using 8-

978-1-4244-5113-5/09/$25.00

bit CPUs and small batteries that only last for about 25 minutes in full motion. Complex motion planning is not practical for such platforms due to the limited computational power and short battery lifetime.

In this paper, we use low-cost disposable mobile relays to reduce the total energy consumption of data-intensive WSNs. Different from mobile base station or data mules, mobile relays do not transport data; instead, they move to different locations and then remain stationary to forward data along the paths from sources to the base station. As a result, the communication delays can be significantly reduced compared with using mobile stations or data mules.

Our approach is motivated by the current state of mobile sensor platform technology. On the one hand, numerous low-cost mobile sensor prototypes such as Packbot [14], Robomote [16], Khepera [17], and FIRA [18] are now available. Their manufacturing cost is comparable to that of typical static sensor platforms. As a result, they can be massively deployed in a network and used in a disposable manner. Our approach takes advantage of this capability by assuming that we have a large number of mobile relay nodes. On the other hand, due to low manufacturing cost, existing mobile sensor platforms are typically powered by batteries and only capable of limited mobility. Consistent with this constraint, our approach only requires simple motions of mobile relays, i.e., one-shot relocation to designated positions after deployment. Compared with our approach, existing mobility approaches (such as mobile base station and data mule) typically assume a small number of powerful mobile nodes, which does not exploit the availability of massive low-cost mobile nodes.

We make the following contributions in this paper. (1) We formulate the problem of *Optimal Mobile Relay Configuration* in data-intensive WSNs. Our objective of energy conservation is *holistic* in that the total energy consumed by both mobility of relays and wireless transmissions is minimized, which is in contrast to existing mobility approaches that only minimize the transmission energy consumption. The tradeoff in energy consumption between mobility and transmission is exploited by configuring the positions of mobile relays. (2) We develop two algorithms that iteratively refine the configuration of mobile relays and converge to the optimal solution. We show that the optimal position for a mobile relay is not the midpoint of its neighbors, which is in contrast to the result of several previous studies [13], [19] that only account for transmission costs. Instead, the optimal relay configuration depends on both the initial positions of nodes and the amount of data to be transmitted. Our optimal algorithms have efficient distributed implementations that do not require explicit synchronization. (3) We conduct extensive simulations based on realistic energy models obtained from existing mobile and static sensor platforms. Our results show that our algorithms can reduce energy consumption by roughly 23% compared to the best existing solutions.

The rest of the paper is organized as follows. Section 2 reviews related work. In Section 3, we formally define our problem. We propose an iterative algorithm for the single-flow problem in Section 4. In Section 5, we extend our solution to the multi-flow problem. Section 6 describes our simulation results and Section 7 concludes this paper.

2. Related Work

We review three different approaches, mobile base stations, data mules, and mobile relays, that use mobility to reduce energy consumption. A mobile base station moves around the network and collects data from the nodes. In some work, all nodes are always performing multiple hop transmissions to the base station, and the goal is to rotate which nodes are close to the base station in order to balance the transmission load [4]–[6]. In other work, nodes only transmit to the base station when it is close to them (or a neighbor). The goal is to compute a mobility path to collect data from visited nodes before those nodes suffer buffer overflows [7], [8], [14], [15]. These approaches incur high latencies due to the low to moderate speed, e.g. 0.1-1 m/s [14], [16], of mobile base stations.

Data mules are similar to the second form of mobile base stations [9]–[11]. Data mules pick up data from the sensors and transport it to the sink. In [20], the data mule visits all the sources to collect data, transports data over some distance, and then transmits it to the static base station through the network. The goal is to find a movement path that minimizes both communication and mobility energy consumption. Similar to mobile base stations, data mules introduce large delays since sensors have to wait for a mule to pass by before starting their transmission.

In the third approach, the network consists of mobile relay nodes along with static base station and data sources. Relay nodes do not transport data; instead, they move to different locations to decrease the transmission costs. We use the mobile relay approach in this work. Goldenberg et al. [13] showed that an iterative mobility algorithm where each relay node moves to the midpoint of its neighbors converges on the optimal solution for a single routing path. However, they do not take into consideration the cost of moving the relay

nodes. In [19], mobile nodes decide to move only when moving is beneficial, (i.e., mobility costs are covered by the savings in transmission costs). However, only the cost of moving to the midpoint of neighbors is considered in [19].

Compared to the approaches of mobile base stations and data mules, our approach considers the energy consumption of both mobility and transmission, and requires much simpler motion planning. Specifically, after deployment, each mobile relay relocates only once and then remains stationary for data forwarding. The key difference between our approach and the mobile relay schemes in [13] and [19] is that we consider all possible locations as possible target locations for a mobile node instead of just its current position and the midpoint of its neighbors. Moreover, our approach accurately computes the optimal position of each mobile relay node.

3. Problem Definition

3.1. Energy Consumption Models

Nodes consume energy during communication, computation, and movement, but communication and mobility energy consumption are the major cause of battery drainage. Radios consume considerable energy even in an idle listening state, but the idle listening time of radios can be significantly reduced by a number of sleep scheduling protocols [21]. In this work, we focus on reducing the *total* energy consumption due to transmissions and mobility. Such a holistic objective of energy conservation is motivated by the fact that mobile relays act the same as static forwarding nodes after movement.

For mobility, we consider wheeled sensor nodes with differential drives such as Khepera [17], Robomote [16] and FIRA [18]. This type of node usually has two wheels, each controlled by independent engines. We adopt the distance proportional energy consumption model which is appropriate for this kind of node [22]. The energy $E_M(d)$ consumed by moving a distance d is modeled as:

$$E_M(d) = kd$$

The value of the parameter k depends on the speed of the node. In general, there is an optimal speed at which k is lowest. In [22], the authors discuss in detail the variation of the energy consumption with respect to the speed of the mote. When the node is running at optimal speed, $k = 2$ [22].

To model the energy consumed through transmissions, we analyze the empirical results obtained by two radios CC2420 [23] and CC1000 [24] that are widely used on existing sensor network platforms. For CC2420, the authors of [25] studied the transmission power level needed for transmitting packets reliably (e.g., above 95% packet reception ratio) over different distances. Let $E_T(d)$ be the energy consumed to transmit reliably over distance d. It can be modeled as

$$E_T(d) = m(a + bd^2)$$

where m is the number of bits transmitted and a and b are constants depending on the environment. We now discuss the instantiation of the above model for both CC2420 and CC1000 radio platforms. In an outdoor environment, for received signal strength of -80 dbm (which corresponds to a packet reception ratio higher than 95%), we obtain $a = 0.6 \times 10^{-7} J/bit$ and $b = 4 \times 10^{-10} Jm^{-2}/bit$ from the measurements on CC2420 in [25]. This model is consistent with the theoretical analysis discussed in [26]. We also consider the energy needed by CC1000 to output the same levels. We get lower consumption parameters: $a = 0.3 \times 10^{-7} J/bit$ and $b = 2 \times 10^{-10} Jm^{-2}/bit$. We will see in section 4.1 that we maintain this high packet reception ratio throughout our algorithm.

3.2. An Illustrative Example

We now describe the main idea of our approach using a simple example. Suppose we have three nodes s_1, s_2, s_3 located at positions x_1, x_2, x_3, respectively (Figure 1), such that s_2 is a mobile relay node. The objective is to minimize the total energy consumption due to both movement and transmissions. Data storage node s_1 needs to transmit a data chunk to sink s_3 through relay node s_2. One solution is to have s_1 transmit the data from x_1 to node s_2 at position x_2 and node s_2 relays it to sink s_3 at position x_3; that is, node s_2 does not move. Another solution, which takes advantage of s_2's mobility, is to move s_2 to the midpoint of the segment x_1x_3, which is suggested in [13]. This will reduce the transmission energy by reducing the distances separating the nodes. However, moving relay node s_2 also consumes energy. We assume the following parameters for the energy models: $k = 2, a = 0.6 \times 10^{-7}, b = 4 \times 10^{-10}$.

In this example, for a given data chunk m_i, the optimal solution is to move s_2 to x_2^i (a position that we can compute precisely). This will minimize the total energy consumption due to both transmission and mobility. For small messages, s_2 moves very little if at all. As the size of the data increases, relay node s_2 moves closer to the midpoint. Table 1 illustrates how much energy can be saved by using the optimal approach instead of the other two approaches. In this

example, for large enough messages (13 MB), one relay node can reduce total energy consumption by 10% compared to the other two approaches. In general, the energy consumption reduction is higher when there are multiple mobile relay nodes.

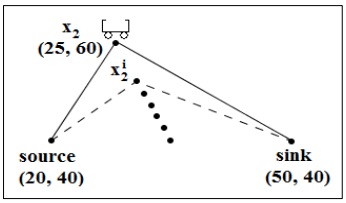

Figure 1. Reduction in energy consumption due to mobile relay. As the data chunk size increases, the optimal position converges to the midpoint of $s_1 s_3$.

Table 1. Energy consumption comparison

Data Size (MB)	Costs at Original Pos.	Costs at Midpoints	Costs at Optimal Pos.	Reduction
11.00	94.12	101.93	88.39	6.09%
12.00	102.68	107.13	94.71	7.75%
13.00	111.23	112.33	100.87	9.32%
14.00	119.79	117.53	106.89	9.06%
15.00	128.35	122.74	112.80	8.09%
16.00	136.90	127.94	118.62	7.28%
17.00	145.46	133.14	124.37	6.58%
18.00	154.01	138.34	130.06	5.98%

The above example illustrates two interesting results. The optimal position of a mobile relay is not the midpoint between the source and sink when both mobility and transmissions costs are taken into consideration. This is in contrast to the conclusion of several previous studies [12], [13] which only account for transmission costs. Second, the optimal position of a mobile relay depends on not only the network topology (e.g., the initial positions of nodes) but also the amount of data to be transmitted. Moreover, as the data chunk size increases, the optimal position converges to the midpoint of s1 and s3. These results are particularly important for minimizing the energy cost of data-intensive WSNs as the traffic load of such networks varies significantly with the sampling rates of nodes and network density.

3.3. Problem Formulation

In our definitions, we assume that all movements are completed before any transmissions begin. As we show in section 4.1, the distance moved by a mobile relay is no more than the distance between its starting position and its corresponding position in the evenly spaced configuration along the straight line between the source and the destination, which often leads to a short delay in mobile relay relocation. We assume that transmission routes from the sources to the sink are already determined. This is a first step towards solving the more general problem in which transmission routes are not known in advance. Furthermore, we assume that all mobile nodes know their locations either by GPS units mounted on them or a localization service in the network. We focus on the case where all nodes are in a 2-dimensional plane $\Re^2$, but the results apply to $\Re^3$ and other metric spaces.

Our problem can be described as follows. Given a network containing one or more static source nodes that store data gathered by other nodes, a number of mobile relay nodes and a static sink, we want to find the optimal positions to move the mobile relays to in order to minimize the total energy consumed by transmitting a data chunk from the source(s) to the sink and the energy consumed by the mobile relays to reach their new locations. The source nodes in our problem formulation serve as *storage points* which cache the data gathered by other nodes and periodically transmit to the sink, in response to user queries. Such a network architecture is consistent with the design of storage-centric sensor networks [27]. Our problem formulation also considers the initial positions of nodes and the amount of data that needs to be transmitted from each storage node to the sink. The formal definition of the problem is given below.

Definition 1 (Mobile Relay Configuration):
Input Instance: S, a list of n nodes $(s_1, \ldots, s_n)$ in the network; O, a list of n locations $(o_1, \ldots, o_n)$ where o_i is the initial position of node s_i for $1 \leq i \leq n$; $S_{sources}$, a subset of S representing the source nodes; r, a node in S, representing the single sink; $M_{sources} = \{M_i \mid s_i \in S_{sources}\}$, a set of data chunk sizes for all sources in $S_{sources}$; E, a set of directed arcs (s_i, s_j) that represents the directed tree in which all sources are leaves and the sink is the root.

We define m_i, which we compute later, to be the weight of node s_i which is equal to the total number of bits to be transmitted by node s_i. We define a configuration U as a list of n locations $(u_1, \ldots, u_n)$ where u_i is the transmission position for node s_i for $1 \leq i \leq n$. The cost of a configuration U is given by: $c(U) = \sum_{(s_i,s_j)\in E} am_i + b\|u_i - u_j\|^2 m_i + k\|o_i - u_i\|$
Output: U, an optimal configuration that minimizes the cost $c(U)$.

We note that if only a subset of the relay nodes is mobile, we can decompose the problem into subproblems in which all intermediate relay nodes in each subtree are mobile. Also, if there is one or more sources that are not leaves in the tree, we can decompose again

to subproblems in which each non-mobile intermediate source node is a sink. Our algorithm can be applied to each subproblem to yield an optimal solution for the main problem.

4. Mobile Relay Configuration for Single Data Flow

We first study the special case where there exists only one data flow. For example, storage nodes may be sparsely distributed and their paths to the sink may not share any intermediate nodes. The solution to this special case also provides important insights for the general case with multiple data flows (Section 5).

4.1. Iterative Algorithm

We propose a simple iterative approach to compute the optimal position u_i for each node s_i. To define our algorithm, we need the following notation. In $\Re^2$, let the original position of node s_i be $o_i = (p_i, q_i)$, and let $u_i^j = (x_i^j, y_i^j)$ be the position of node s_i after the jth iteration of our algorithm for $j \geq 0$. We define $u_i^0 = o_i$. Let $U^j = (u_1^j, \ldots, u_n^j)$ be the computed configuration of nodes s_1 through s_n after j iterations of our algorithm. Note that the mobile relay nodes typically do not move until the final positions are computed. Since there is a single source, the same data chunk size is transmitted from the source through intermediate nodes. We define m to be this value ($m = M_1$). According to our energy models, the total transmission and movement energy cost incurred by mobile relay node s_i assuming all the mobile relays move from the initial configuration U^0 directly to configuration U^j before performing any transmissions is $c_i(U^j) = k\|u_i^j - o_i\| + am + b\|u_{i+1}^j - u_i^j\|^2 m$, and the total cost of configuration U^j is $c(U^j) = \sum_{i=1}^{n-1} c_i(U^j)$. Finally, we define $C_i(U^j) = c_i(U^j) + am + b\|u_i^j - u_{i-1}^j\|^2 m$; this corresponds to the transmission cost of node s_{i-1} plus the total cost of node s_i.

In odd iterations j, the algorithm computes a position u_i^j for each odd-numbered node s_i that minimizes $C_i(U^j)$ assuming that $u_{i-1}^j = u_{i-1}^{j-1}$ and $u_{i+1}^j = u_{i+1}^{j-1}$; that is, node s_i's even numbered neighboring nodes are at the same positions in configurations U^{j-1} and U^j. In even-numbered iterations, the controller does the same for even-numbered nodes. The algorithm behaves this way because the optimization of u_i^j requires a fixed location for nodes $i-1$ and $i+1$. By alternating between optimizing for odd and even numbered nodes, the algorithm guarantees that the node s_i is always making progress towards the optimal position u_i.

The algorithm calculates position $u_i^j = (x_i^j, y_i^j)$ for node s_i by finding the values for x_i^j and y_i^j where the partial derivatives of the cost function $C_i(U^j)$ with respect to x_i^j and y_i^j become zero. Position u_i^j will be toward the midpoint of positions u_{i-1}^j and u_{i+1}^j. The partial derivatives $\frac{\delta C_i(U^j)}{\delta x_i^j}$, $\frac{\delta C_i(U^j)}{\delta y_i^j}$ at x_i^j and y_i^j, respectively are defined as follows.

$$\frac{\delta C_i(U^j)}{\delta x_i^j} = -2bm(x_{i+1}^j - x_i^j) + 2bm(x_i^j - x_{i-1}^j) + k\frac{(x_i^j - p_i)}{\sqrt{(x_i^j - p_i)^2 + (y_i^j - q_i)^2}}$$

$$\frac{\delta C_i(U^j)}{\delta y_i^j} = -2bm(y_{i+1}^j - y_i^j) + 2bm(y_i^j - y_{i-1}^j) + k\frac{(y_i^j - q_i)}{\sqrt{(x_i^j - p_i)^2 + (y_i^j - q_i)^2}}$$

Setting $\frac{\delta C_i(U^j)}{\delta x_i^j} = 0, \frac{\delta C_i(U^j)}{\delta y_i^j} = 0$, we get the following two cases. Suppose s_i needs to move left. This means p_i is to the right of the midpoint of nodes s_{i-1} and s_{i+1}. Let $Y_i^j = \frac{k}{4bm}\frac{1}{\sqrt{1+\frac{(y_{i-1}^j + y_{i+1}^j - 2q_i)^2}{(x_{i-1}^j + x_{i+1}^j - 2p_i)^2}}}$. The optimal position is then $x_i^j = \frac{1}{2}(x_{i-1}^j + x_{i+1}^j) + Y_i^j$. If s_i needs to move right, then p_i is to the left of the midpoint of nodes s_{i-1} and s_{i+1}. The optimal position is then $x_i^j = \frac{1}{2}(x_{i-1}^j + x_{i+1}^j) - Y_i^j$. The corresponding y_i^j in both cases is $\frac{(x_{i-1}^j + x_{i+1}^j - 2p_i)}{(y_{i-1}^j + y_{i+1}^j - 2q_i)}(x_i^j - p_i) + q_i$.

Figure 3 shows an example of an optimal configuration. Nodes start at configuration U^0. In the first iteration, odd nodes (s_3 and s_5) moved to their new positions (u_3^1, u_5^1) computed based on the current location of their (even) neighbors (u_2^0, u_4^0, u_6^0). In the second iteration, only even nodes (s_2 and s_4) moved to their new positions (u_2^2, u_4^2) computed based on the current location of their (odd) neighbors (u_1^1, u_3^1, u_5^1). Since s_3 and s_5 did not move, their position at the end of this iteration remains the same, so $u_3^1 = u_3^2$ and $u_5^1 = u_5^2$. In this example, nodes did two more sets of iterations, and finally converged to the optimal solution shown by configuration U^6.

Even though configurations change with every iteration, nodes only move after the final positions have been computed. So each node follows a straight line to its final destination. As the data chunk size increases, the optimal configuration gets closer to the straight line connecting the source and the sink and the nodes get more evenly spaced along that line. In fact, in any given configuration, the maximum distance travelled by a node is bounded by the distance between its starting position and its final position in the evenly spaced configuration.

```
procedure OPTIMALPOSITIONS(U^0)
    converged ← false;
    j ← 0;
    repeat
        anymove ← false;
        j ← j + 1;

        ▷ Start an odd iteration
        for i = 2 to n, i is odd do
            (u_i^j, moved) ← LOCALPOS(o_i, u^{j-1}, u_{i-1}^{j-1}, u_{i+1}^{j-1});
            ▷ Record if any node moved
            anymove ← anymove OR moved
        end for

        ▷ Start an even iteration
        for i = 2 to n, i is even do
            (u_i^j, moved) ← LOCALPOS(o_i, u^{j-1}, u_{i-1}^{j}, u_{i+1}^{j});
            ▷ Record if any node moved
            anymove ← anymove OR moved
        end for

        ▷ Update convergence status
        converged ← NOT anymove
    until converged
end procedure

function LOCALPOS(o_i, u_i^j, u_{i-1}^k, u_{i+1}^k)
    ▷ Consider case s_i moves right
    x_i ← (1/2)(x_{i-1} + x_{i+1}) − Y_i;
    if x_i > p_i then
        y_i ← ((x_{i-1}+x_{i+1}-2p_i)/(y_{i-1}+y_{i+1}-2q_i))(x_i − p_i) + q_i;

        u_i^{j+1} = (x_i, y_i);
        ▷ Record if new position is different from previous one
        if ||u_i^{j+1} − u_i^j|| < threshold then
                return (u^{j+1}, FALSE);
        else
                return (u^{j+1}, TRUE);
        end if
    end if

    ▷ Consider case s_i moves left
    x_i ← (1/2)(x_{i-1} + x_{i+1}) + Y_i
    if x_i < p_i then
        y_i ← ((x_{i-1}+x_{i+1}-2p_i)/(y_{i-1}+y_{i+1}-2q_i))(x_i − p_i) + q_i;

        u_i^{j+1} = (x_i, y_i);
        ▷ Record if new position is different from previous one
        if ||u_i^{j+1} − u_i^j|| < threshold then
                return (u^{j+1}, FALSE);
        else
                return (u^{j+1}, TRUE);
        end if
    end if

    ▷ not beneficial to move, stay at original position
    return (o_i, FALSE);
end function
```

Figure 2. Centralized Algorithm to Compute Optimal Positions

The above example shows another property of our algorithm. When a node s_i moves and its neighbors (s_{i-1} and s_{i+1}) remain in place, it moves in the direction of the midpoint of $s_{i-1}s_{i+1}$. This results in a reduction in the length of one of the transmission links. The other may increase in length but will never exceed the new length of the first link. So in any configuration U^{i+1}, the length of the largest link is at most the length of the largest link in the previous configuration U^i. So if we start with a route along links with good quality, this quality will be preserved in the optimal configuration (and throughout intermediate configurations).

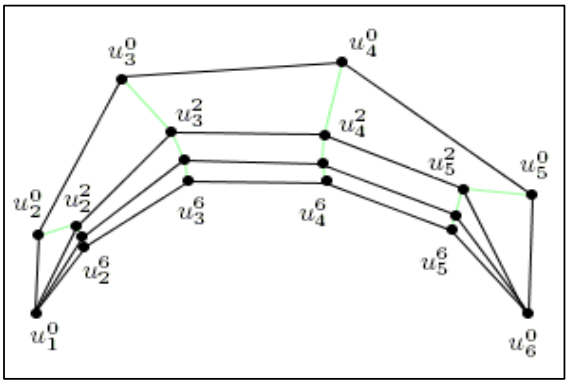

Figure 3. Convergence of iterative approach to the optimal solution. Each line shows the configuration obtained after 2 iterations. The optimal configuration is reached after 6 iterations.

4.2. Convergence and Optimality

Our algorithm continues iterating until the change in position for each node falls below a predefined threshold. Upon termination, no node can move by itself to improve the overall cost (within the threshold bound). We prove that such a configuration is globally optimal, and no simultaneous relocation of multiple nodes can improve the overall cost. The proof of optimality is omitted due to lack of space and can be found in [28]. The key intuition is that for a configuration in which no relay node can move and improve the cost by itself, the directional derivative [29] at that configuration is positive; this is a sufficient condition for the optimality of that configuration because the cost function is convex.

We have not completed a rate of convergence analysis for our algorithm. However, in our simulations, we reach our error threshold within 6 to 8 iterations. Since each iteration involves only half the nodes and each computation of u_i^j can be performed in constant time, the time complexity of our algorithm is $O(\lambda n)$, where λ is the number of iterations to reach convergence. Given that $\lambda \leq 8$ in our simulations, our observed time complexity is $O(n)$.

4.3. Centralized and Distributed Implementation

One implementation of our iterative approach is a centralized scheme in which one node has full

knowledge of the network including which nodes (s_1, ..., s_n) are on the transmission path, the order position i of each node s_i, the original physical position o_i of each node s_i, and the total message length m to be sent. This node computes u_i, the final position of each node s_i and sends u_i to node s_i. Each node s_i then moves to u_i before the transmission begins.

While the centralized algorithm computes optimal position u_i for each node s_i, it incurs prohibitively high overhead in large-scale networks. We now present a distributed and decentralized version of our optimal algorithm. The key observation is that computing each u_i^j for node s_i only depends on the current position of s_i's two neighbors, s_{i-1} and s_{i+1}. Thus, s_i can perform this computation.

The distributed implementation proceeds as follows. First, there is a setup process where the sender s_1 sends a discover message that ends with the receiver s_n; the two purposes of this message are (1) to assign a label of odd or even to each node s_i and (2) for each node s_i to learn the current positions of nodes s_{i-1} and s_{i+1}. A node s_i sends its current position to node s_{i-1} when acknowledging receipt of the discover message. Second, there is a distributed process by which the nodes compute their transmission positions. We make each iteration of the basic algorithm a "round", though there does not need to be explicit synchronization. In odd rounds, each odd node computes its locally optimal position and transmits this new position to its two neighbors. In even rounds, each even node does the same. A node begins its next round when it receives updated positions from its two neighbors. The final step is to have the nodes move to their computed transmission positions, send messages to their neighbors saying they are in position, and finally perform the transmission. To ensure the second process does not take too long, we limit the number of rounds to 4; that is, each node computes an updated position two times. Simulation results show that this is enough to obtain costs close to optimal (see Section 6).

5. Mobile Relay Configuration for Multiple Data Flows

We extend our solution to the the more general multiple flows traffic pattern. We assume that we know the path from each source to the sink. The set of paths form a directed tree in which the leaves are sources and the root is the sink. We also assume that separate messages cannot be compressed through merging; that is, if two distinct messages of lengths m_1 and m_2 use the same link (s_i, s_j) on the path from a source to a sink, the total number of bits that must traverse link (s_i, s_j) is $m_1 + m_2$.

Before we can describe our modified iterative algorithm, we need to define the following notation. First, for a given node s_i, we define node s_d to be the parent node of s_i in the directed tree; that is, node s_i will transmit to node s_d. Furthermore, we use $S(s_i)$ to denote the set of nodes that have s_i as their parent in the tree. The cost incurred by s_i in a configuration U^j is $c_i(U^j) = k\|u_i^j - o_i\| + am_i + bm_i\|u_d^j - u_i^j\|^2$; remember that s_i transmits to s_d. The total cost of U^j is $c(U^j) = \sum c_i(U^j)$. Finally, $C_i(U^j) = c_i(U^j) + \sum_{s_l \in S(s_i)} am_l + b\|u_i^j - u_l^j\|^2 m_l$; this corresponds to the transmission cost of all nodes s_l that send messages to node s_i plus the total cost of node s_i.

The key modifications to our iterative algorithm are modified routines for labeling nodes as odd or even, a new routine to calculate m_i for each node s_i, and an updated routine to determine position u_i^j that takes into account multiple nodes transmitting to node s_i. To obtain consistent labels for nodes, we start the labeling process from the root using a breadth first traversal of the tree. The root gets labeled as even. Each of its children gets labeled as odd. Each subsequent child is then given the opposite label of its parent. We define m_i, the weight of a node s_i, to be the sum of message lengths over all paths passing through s_i. This computation starts from the sources or leaves of our routing tree. Initially, we know $m_i = M_i$ for each source leaf node s_i. For each intermediate node s_i, we compute its weight as the sum of the weights of its children.

The routine for calculating u_i^j still minimizes $C_i(U^j)$. The difference is that $C_i(U^j)$ includes the transmission cost of all nodes $s_l \in S(s_i)$ that transmit to node s_i. Setting $\frac{\delta C_i(U^j)}{\delta x_i^j} = 0$, $\frac{\delta C_i(U^j)}{\delta y_i^j} = 0$, the new position of node s_i can be computed as follows:

$$x_i^j = \frac{-B_x^j r^j + k x_i^j A}{A(r^j + k)}, \quad y_i = \frac{-B_y^j r^j + k y_i^j A}{A(r^j + k)}$$

where
$$A = m_i + \sum_{s_l \in S(s_i)} m_l$$
$$B_x^j = m_i x_d^j + \sum_{s_l \in S(s_i)} m_l x_l^j$$
$$B_y = m_i y_d^j + \sum_{s_l \in S(s_i)} m_l y_l^j$$
$$r = -k \pm \sqrt{(B_x^j + A x_i^j)^2 + (B_y^j + A y_i^j)^2}$$

The cost functions $c(U^j)$ and $C_i(U^j)$ are still convex which means that a configuration in which no node

can move to reduce the cost function is still an optimal configuration. The proof of optimality is similar to the proof for the single flow case. Details can be found in [28]. In our simulations, we observe that 10 iterations suffice to converge to our error threshold.

This algorithm again can be implemented as a centralized or a distributed algorithm. The basic mechanisms are identical. First a labeling process is needed which proceeds from the sink to the sources. Then the node weights are computed which proceeds from the sources to the sink. Finally, we perform the iterative algorithm to compute u_i^j for each node s_i. The only significant change is that each node has to wait for an update from all its neighbors before computing its next position. In the distributed implementation, we limit this process to 8 rounds. We observe 8 rounds is sufficient to obtain nearly optimal results.

6. Simulation Results

We carried out simulations on 100 network topologies for each of the single flow and multiple flows configurations, each topology consisting of 100 randomly placed nodes in a 150m by 150m area, with randomly selected sources and sink. To determine the transmission path between the sources and sink, any routing algorithm can be used. We arbitrarily picked greedy geographic forwarding in which each node forwards the packet to the neighbor that is closest to the sink. Of the 200 network topologies, only four networks resulted in a disconnected path between the sources and the sink, due to the existence of routing voids. We used 10^{-4} meters as our error threshold.

In the mobility energy consumption model, we use k values 0.25, 1, 2, and 4 with $k = 2$ as the standard setting because it models several platforms such as Robomote [16], [17]. For transmission, we use $a = 0.6 \times 10^{-7}$ and $b = 4 \times 10^{-10}$ as the standard setting which is consistent with the empirical measurements on CC2420 motes [25]. We also use other settings shown in Table 2 to consider other platforms. We set the maximum communication distance of a node to be roughly 30m, which was shown to result in a high packet reception ratio for CC2420 in [25].

6.1. Single Data Flow

We first consider the single data flow scenario. We compare the minimum cost computed by our centralized algorithm to the approaches proposed in [13] and [19]. In the first approach, nodes move to the evenly spaced positions along a straight line from the source to the sink. This results in lower costs than those of both [13] and [19] since we assume that nodes follow a straight line to reach their final positions whereas in [13] and [19], nodes converge to their final destination by following a series of segments along a broken line. The second approach we compare to is transmitting data from the original positions, which is used in [19] when it is not beneficial for the nodes to move. In general, for each value of k, we get the same trend as shown in figure 4. When the data chunk is small, the optimal positions coincide with the original positions. As data size increases, the optimal positions diverge from the original positions towards the evenly spaced positions. For example, when $k = 2$, the optimal positions diverge from the original positions for data lengths as short as 5 MB (figure 4).

Our algorithms are most effective for data that is not too short to initiate a move or too long to cause a convergence to the evenly spaced positions. For these cases, our algorithm reduces energy consumption by up to 23% in comparison to [19] as shown in figure 4. This improvement is attained for all energy costs parameters. The data size at which this improvement is reached can be as low as 1 MB when the mobility cost is low compared to the transmission costs. When the mobility cost is high ($k = 4$) and the transmission costs are low ($b = 2 \times 10^{-10}$), the maximum improvement is reached for data chunks around 60 MB.

We now evaluate the performance of the distributed implementation. The energy consumed by the distributed approach is at most 1.5% more than the energy consumed by the optimal centralized version. This error margin is due to stopping the computation after only 4 rounds of updates (i.e. only two updates per node) as shown in figure 5. The distributed approach converges to the optimal positions if we allow 8 rounds of computation. We obtain the same results of quick convergence and small error margin after 4 rounds for all settings.

6.2. Multiple Data Flows

We now consider the multiple data flows scenario. We used between 4 and 12 sources. We used the same

Table 2. Improvement of optimal approach for various energy model parameters

a ($\times 10^{-7}$)	b ($\times 10^{-10}$)	k	Reduction	Data size (MB)
1.2	8	4	22.61%	32.5
1.2	8	1	22.96%	8
0.6	4	4	22.61%	65
0.6	4	0.25	22.96%	4
0.3	2	1	22.61%	32.5
0.3	2	0.25	22.96%	8
0.15	1	2	22.61%	130
0.15	1	0.25	22.96%	4

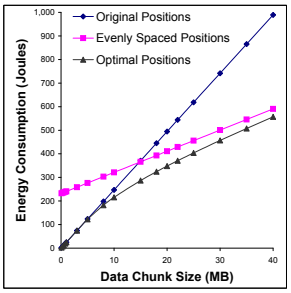

Figure 4. Comparison among the three approaches for a single data flow

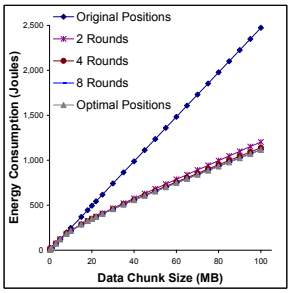

Figure 5. Convergence to optimal configuration for a single data flow

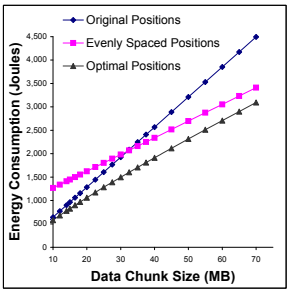

Figure 6. Comparison among the three approaches for multiple data flows

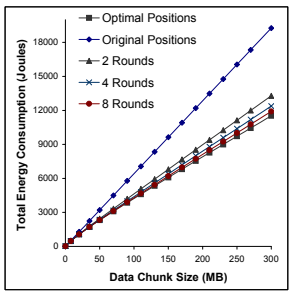

Figure 7. Convergence to optimal configuration for multiple data flows

message length at each source. Most of our results are similar to those from the single flow scenario.

In the centralized case, the cost of the optimal positions is equal to the cost of the original positions for small data chunks. It starts moving away as the data gets longer and finally meets with proportionally spaced positions for large data chunks as shown in Figure 6. Our approach performs best when the costs of the other two approaches are similar. In this example, this is reached for data chunks around 32 (MB), and the savings go up to 23%. We obtain similar results for all parameter values. Table 2 summarizes the reduction obtained and the data size that yields it for a variety of parameter settings. In all cases, similar reduction ratios ($\approx 23\%$) are reached, but the data size is different. It decreases as k decreases compared to a and b.

We then evaluate the distributed approach for multiple flows by comparing it to the optimal centralized approach. For all values of k, a and b, the distributed approach quickly converges to the optimal configuration. Figure 7 shows the average costs for $a = 0.6 \times 10^{-7}, b = 4 \times 10^{-10}, k = 2$; after eight rounds of updates, the distributed costs are insignificantly higher than the centralized costs. The extra cost incurred is less than 2% for data chunks shorter than 150MB and less than 3% for data chunks up to 300MB.

7. Conclusion and Future Work

In this paper, we proposed a holistic approach to minimize the total energy consumed by both mobility of relays and wireless transmissions. Most previous work ignored the energy consumed by moving mobile relays. We developed an iterative approach to compute the optimal positions of relay nodes that can be implemented in a centralized or distributed fashion. Our algorithms are appropriate for a variety of data-intensive wireless sensor networks. In both single flow and multiple flow patterns, we show that our holistic approach can reduce total energy consumption by up to 23% compared to previous approaches. The optimal position for a mobile relay is not the midpoint of its neighbors; instead, it converges to this position as the amount of data transmitted goes to infinity. Our algorithms allow some nodes to move while others do not because any local improvement for a given mobile relay is a global improvement. This allows us to potentially extend our approach to handle additional constraints on individual nodes such as low energy levels or mobility restrictions due to application requirements.

Our algorithms are optimal when the transmission routes are predetermined and participating nodes remain the same throughout different configurations. In some cases, it may be beneficial to insert a new relay along the transmission path after several rounds if the new route moves into its proximity, or alternatively, drop a node if the route moves away from it. For future work, we plan on studying the problem of finding the optimal routes along with the optimal locations of

nodes on those routes.

Acknowledgement

The authors thank Philip McKinley and Chiping Tang for their help.

References

[1] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler, "An analysis of a large scale habitat monitoring application," in *SenSys*, 2004.

[2] L. Luo, Q. Cao, C. Huang, T. F. Abdelzaher, J. A. Stankovic, and M. Ward, "Enviromic: Towards cooperative storage and retrieval in audio sensor networks," in *ICDCS*, 2007, p. 34.

[3] D. Ganesan, B. Greenstein, D. Perelyubskiy, D. Estrin, and J. Heidemann, "An evaluation of multi-resolution storage for sensor networks," in *SenSys*, 2003.

[4] S. R. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, "Energy efficient schemes for wireless sensor networks with multiple mobile base stations," in *Globecom*, 2003.

[5] J. Luo and J.-P. Hubaux, "Joint mobility and routing for lifetime elongation in wireless sensor networks," in *INFOCOM*, 2005.

[6] Z. M. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli, "Exploiting sink mobility for maximizing sensor networks lifetime," in *HICSS*, 2005.

[7] A. Kansal, D. D. Jea, D. Estrin, and M. B. Srivastava, "Controllably mobile infrastructure for low energy embedded networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 8, 2006.

[8] G. Xing, T. Wang, W. Jia, and M. Li, "Rendezvous design algorithms for wireless sensor networks with a mobile base station," in *MobiHoc*, 2008, pp. 231–240.

[9] D. Jea, A. A. Somasundara, and M. B. Srivastava, "Multiple controlled mobile elements (data mules) for data collection in sensor networks," in *DCOSS*, 2005.

[10] R. Shah, S. Roy, S. Jain, and W. Brunette, "Data mules: Modeling a three-tier architecture for sparse sensor networks," in *IEEE SNPA Workshop*, 2003.

[11] S. Jain, R. Shah, W. Brunette, G. Borriello, and S. Roy, "Exploiting mobility for energy efficient data collection in wireless sensor networks," *MONET*, vol. 11, no. 3, pp. 327–339, 2006.

[12] W. Wang, V. Srinivasan, and K.-C. Chua, "Using mobile relays to prolong the lifetime of wireless sensor networks," in *MobiCom*, 2005.

[13] D. K. Goldenberg, J. Lin, and A. S. Morse, "Towards mobility as a network control primitive," in *MobiHoc*, 2004, pp. 163–174.

[14] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling with dynamic deadlines," *IEEE Transactions on Mobile Computing*, vol. 6, no. 4, 2007.

[15] Y. Gu, D. Bozdag, and E. Ekici, "Mobile element based differentiated message delivery in wireless sensor networks," in *WoWMoM*, 2006.

[16] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme, "Robomote: enabling mobility in sensor networks," in *IPSN*, 2005.

[17] http://www.k-team.com/robots/khepera/index.html.

[18] J.-H. Kim, D.-H. Kim, Y.-J. Kim, and K.-T. Seow, *Soccer Robotics*. Springer, 2004.

[19] C. Tang and P. K. McKinley, "Energy optimization under informed mobility," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 9, pp. 947–962, 2006.

[20] C.-C. Ooi and C. Schindelhauer, "Minimal energy path planning for wireless robots," in *ROBOCOMM*, 2007, p. 2.

[21] L. Wang and Y. Xiao, "A survey of energy-efficient scheduling mechanisms in sensor networks," *Mob. Netw. Appl.*, vol. 11, no. 5, 2006.

[22] G. Wang, M. J. Irwin, P. Berman, H. Fu, and T. F. L. Porta, "Optimizing sensor movement planning for energy efficiency," in *ISLPED*, 2005, pp. 215–220.

[23] "Cc2420 datasheet," http://inst.eecs.berkeley.edu/cs150/Documents/CC2420.pdf.

[24] "Cc1000 single chip very low power rf transceiver." http://focus.ti.com/lit/ds/symlink/cc1000.pdf.

[25] M. Sha, G. Xing, G. Zhou, S. Liu, and X. Wang, "C-mac: Model-driven concurrent medium access control for wireless sensor networks," in *INFOCOM*, 2009.

[26] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *HICSS*, 2000.

[27] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, "Data-centric storage in sensornets with ght, a geographic hash table," *MONET*, vol. 8, no. 4, pp. 427–442, 2003.

[28] F. El-Moukaddem and E. Torng, "Energy efficient routing in mobile wireless sensor networks," Computer Science and Engineering, Michigan State University, East Lansing, Michigan, Tech. Rep. MSU-CSE-09-4, February 2009.

[29] J. M. Borwein and A. S. Lewis, *Convex Analysis and NonLinear Optimization. Theory and Exmaples*. Springer, 2000.