Improving the Data Delivery Latency in Sensor Networks with Controlled Mobility

Ryo Sugihara* and Rajesh K. Gupta

Computer Science and Engineering Department, University of California, San Diego
{ryo,rgupta}@ucsd.edu

Abstract. Unlike traditional multihop forwarding among homogeneous static sensor nodes, use of mobile devices for data collection in wireless sensor networks has recently been gathering more attention. It is known that the use of mobility significantly reduces the energy consumption at each sensor, elongating the functional lifetime of the network, in exchange for increased data delivery latency. However, in previous work, mobility and communication capabilities are often underutilized, resulting in suboptimal solutions incurring unnecessarily large latency. In this paper, we focus on the problem of finding an optimal path of a mobile device, which we call "data mule," to achieve the smallest data delivery latency in the case of minimum energy consumption at each sensor, i.e., each sensor only sends its data directly to the data mule. We formally define the path selection problem and show the problem is $\mathcal{NP}$-hard. Then we present an approximation algorithm and analyze its approximation factor. Numerical experiments demonstrate that our approximation algorithm successfully finds the paths that result in 10%-50% shorter latency compared to previously proposed methods, suggesting that controlled mobility can be exploited much more effectively.

1 Introduction

Exploiting mobility is gaining popularity as a means to solve several issues in traditional multihop forwarding approach in wireless sensor networks and mobile ad-hoc networks. Studies have shown that mobility significantly reduces energy consumption at each node, thus prolongs the network lifetime [1][2][3][4]. Controlled mobility, as opposed to random or predictable mobility by the classification in [5], refers to the case that the observers have the control on the motion of mobile devices, and has the biggest potential for improving the performance of the network. We focus on data collection application in sensor networks and use the term "data mules" to refer to such mobile devices from now on. There are some recent applications that employ data mules for data collection in sensor networks, e.g., a robot in underwater environmental monitoring [6] and a UAV (unmanned aerial vehicle) in structural health monitoring [7].

* R. Sugihara was supported by IBM Tokyo Research Laboratory.

S. Nikoletseas et al. (Eds.): DCOSS 2008, LNCS 5067, pp. 386–399, 2008.

There are some studies that analyze the use of data mules in terms of energy efficiency [2][5][8][9]. However, in these studies, mobility and communication capabilities are often underestimated, leading to suboptimal solutions that incur unnecessarily large latency. Some of the underestimations are: the data mule can only move at a constant speed, the data mule needs to go to each node's exact location to collect data from it, the data mule needs to stop during communication with each node, etc.

In this paper, we are interested in improving the data delivery latency in data collection using a data mule. We achieve this through a better formulation of the problem and an efficient approximation algorithm that finds near-optimal solutions. To capture the mobility capability precisely, we assume that the data mule can select the path to traverse the sensor field and also can change its speed under a predefined acceleration constraint. As for the communication capability, we assume that each node can send the data to the data mule when it is within its communication range, regardless of whether the data mule is stopped or moving. With all these assumptions together, we can formulate the problem as a scheduling problem that has both time and location constraints. We focus on the path selection problem in this paper. A heuristic algorithm for optimal speed control and job scheduling can be found in [10].

Our contributions in this paper are:

- Formulating the path selection problem for data collection in sensor networks with a data mule such that the mobility and communication capabilities are precisely captured,
- An efficient approximation algorithm that produces near-optimal paths that enable faster data delivery, and
- Demonstrating the validity and effectiveness of the formulation and the approximation algorithm through numerical experiments by comparing the data delivery latency with previous approaches.

This paper is structured as follows. In Section 2 we introduce the data mule scheduling problem and the related work. In Section 3 we give a formal definition of the path selection problem and describe the preliminary experiments to choose an appropriate cost metric. In Section 4 we present an approximation algorithm and analyze its computational complexity and approximation factor. Section 5 shows some results from numerical experiments and Section 6 concludes the paper.

2 Data Mule Scheduling

The Data Mule Scheduling (DMS) problem is how to control a data mule so that it can collect data from the sensors in a sensor field in the shortest amount of time. As shown in Figure 1, we can decompose the DMS problem into the following three subproblems:

1. Path selection: determines the trajectory of the data mule; produces a set of location jobs

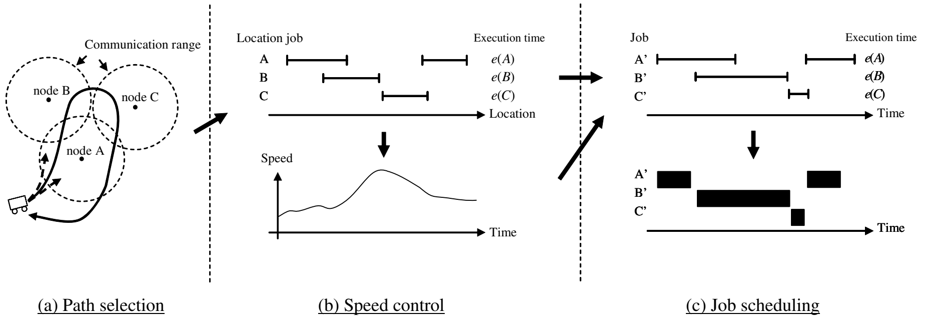


Fig. 1. Subproblems of the data mule scheduling (DMS) problem

2. Speed control: determines how the data mule changes the speed; produces a set of jobs
3. Job scheduling: determines the schedule of data collection jobs from individual sensors

The focus of this paper is on the path selection problem. The other two problems have been formulated as the 1-D DMS problem and we have presented an efficient heuristic algorithm that yields near-optimal solutions [10].

Path selection is to determine the trajectory of the data mule in the sensor field. To collect data from each sensor node, the data mule needs to go within the node's communication range at least once. Depending on the mobility capability of data mule, there can be some constraints on the path, such as the minimum turning radius.

2.1 Why Do We Minimize the Latency?

As mentioned earlier, data mules can be used as an alternative to multihop forwarding in sensor networks. The use of data mules in collecting data introduces the trade-off between energy consumption and data delivery latency. Our objective is to optimize this trade-off, so that energy consumption is minimized under some latency constraint or vice versa.

Protocol designers have tried to optimize the multihop forwarding in both energy and latency through sophisticated MAC protocols [11][12][13]. Data mule, or its combination with multihop forwarding, is a relatively nascent area. In this paper, we focus on the pure data mule approach, in which each node uses only direct communication with the data mule and no multihop forwarding. Energy consumption related to communication is already minimized in this case, since each node only sends its own data and does not forward others' data. Naturally, our objective is to minimize the data delivery latency by minimizing the travel time of the data mule.

2.2 Example Application: SHM with UAV

Our problem formulation is based on our experience with the example application described in [7]. It is a structural health monitoring (SHM) application to do post-event (e.g., earthquakes) assessments for large-scale civil infrastructure such as bridges. Automated damage assessment using sensor systems is much more efficient and reliable than human visual inspections.

In this application, the sensor nodes operate completely passively and do not carry batteries, for the sake of long-term measurement and higher maintainability. Upon data collection, an external mobile element provides energy to each node via microwave transmission, wakes it up, and collects data from it. The prototype system uses a radio-controlled helicopter as the mobile element that is either remotely-piloted or GPS-programmed. Each sensor node is equipped with ATmega128L microcontroller, a 2.4GHz XBee radio, antennas for transmission/reception, and a supercapacitor to store the energy. Each node has two types of sensors. One is a piezoelectric sensing element integrated with nuts and washers to check if the bolt has loosened. The other is capacitive-based sensors for measuring peak displacement and bolt preload. Since the size of data from these sensors are small, communication time is almost negligible; however, it takes a few minutes to charge a supercapacitor through microwave transmission in the current prototype. The team is currently investigating a new design to improve the charging time down to tens of seconds.

The data collected by the UAV is brought back to the base station and analyzed by researchers using statistical techniques for damage existence and its location/type. Since the primary purpose of this application is to assess the safety of large civil structures after a disaster such as an earthquake, every process including data collection and analysis needs to be as quick as possible for prompt recovery. Furthermore, shorter travel time is required in view of the limited fuel on the helicopter.

Thus the goal of our formulation is to achieve data collection from spatially distributed wireless sensors in the minimum amount of time. It also provides another reason for using controlled mobility instead of multihop forwarding approach: simply because the SHM sensors are not capable of doing multihop communication. Furthermore, use of UAVs implies the need for more precise mobility model that takes acceleration constraint into consideration, as opposed to the simple “move or stop“ model used in majority of the related work.

2.3 Related Work

The term “data mule” was coined by Shah et al. in their paper in 2003 [3]. They proposed a three-tier architecture having mobile entities called Data MULEs (Mobile Ubiquitous LAN Extensions) in the middle tier on top of stationary sensors under wired access points. As we have also assumed, Data MULEs collect data from sensor nodes when they are in close proximity and deposit it at the wired access points. The difference is that they assumed Data MULEs are not controllable and move randomly, so their routing scheme is rather optimistic.

The use of controlled mobility in sensor networks has been studied in several papers. Kansal et al. [5] analyzed the case in which a data mule (which is called "mobile router" in the paper) periodically travels across the sensor field along a fixed path. Jea et al. [14] used similar assumptions but further assumed multiple data mules are simultaneously on the sensor field. In their models, they can only change the speed of data mule and path selection is out of scope. Path selection problem has been considered in several different problem settings. Somasundara et al. [8] studied path selection problem, assuming that each sensor generates data at a certain rate and that the data mule needs to collect data before the buffer of each sensor overflows. Gu et al. [15] presented an improved algorithm for the same problem settings. Xing et al. [9] presented path selection algorithms for rendezvous based approach. In these work, it is assumed that data mule needs to go to the sensor node's exact location to collect data (i.e., no remote communication)[1]. Although this assumption facilitates TSP-like formulation of the problem, the communication capability is underutilized, since the data mule can actually collect data from nodes without visiting their exact locations via wireless communications. Ma and Yang [2] also discussed the path selection problem but under different assumptions. They consider remote wireless communication and also multihop communication among nodes. However, they assumed the data transmission time is negligible. In a recent paper from the authors [16], they consider constant bit rate case, but they also assume the data mule stops during the communication, whereas we allow communication while in motion.

There are a number of studies on exploiting mobility also in mobile ad-hoc networks (MANETs) area. Among these, our work is most analogous to Message Ferrying [17]. They assume a controllable mobile node (called "ferry") that mediates communications between sparsely deployed stationary nodes. The speed of ferry is basically constant but can be reduced when it is necessary to communicate more data with a node. Further, they consider the extent of wireless communication range to optimize the movement. In our work, we employ a more precise mobility model with acceleration constraint and also realize a more optimized path selection where the data mule only needs to visit subset of nodes as long as it travels inside the communication ranges of all nodes.

3 Path Selection Problem

In this section we give a formal definition of the path selection problem. To make the problem tractable, we first simplify the problem, where the path consists of the line segments between the nodes. The problem is to find a minimum-cost path that intersects with the communication ranges of all nodes. We prove the problem is still $\mathcal{NP}$-hard. Then we do a preliminary experiment to choose an appropriate cost metric before proceeding to solve the problem.

[1] In [9], remote communication is used for gathering data at rendezvous points via multihop forwarding, but it is assumed that data mule needs to go to the exact location of these rendezvous points.

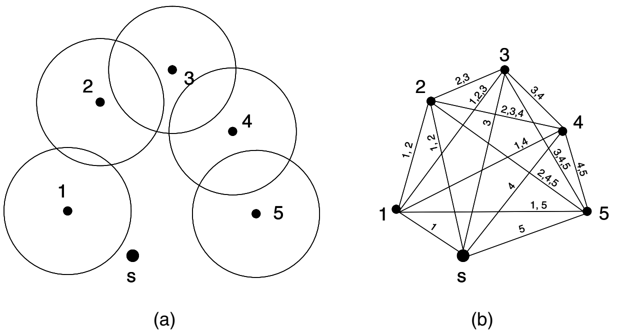


Fig. 2. Simplifying the path selection problem using a labeled graph representation: (a) Instance of path selection problem. (b) Corresponding labeled graph.

3.1 Problem Description

As we have discussed in the previous section, the ultimate objective of the path selection problem is to find a path such that the shortest travel time (= latency) can be realized in the corresponding 1-D DMS problem. However, it is not clear which path results in shorter travel time when there is an acceleration constraint. For example, even if the path length is short, the travel time would be long if the intersections of the path and communication range of each node are short, because the data mule needs to slow down to collect all the data and then accelerate again. Moreover, it is also difficult to search an optimal path in a brute-force manner when the data mule can freely move around within the space.

To deal with these issues, we simplify the path selection problem. To reduce the solution space, we consider a complete graph having vertices at sensor nodes' locations and assume the data mule moves between vertices along a straight line. Each edge is associated with a cost and a set of labels, where the latter represents the set of nodes whose communication ranges intersect with this edge. In other words, the data mule can collect data from these nodes while traveling along this edge. We want to find a minimum-cost tour that the data mule can collect data from all the nodes. We discuss later how we assign the cost to each edge so that a tour with smaller cost results in shorter travel time.

Figure 2 is an example that depicts the basic idea of the formulation. Figure 2(a) shows five nodes and their communication ranges, in addition to the starting point (shown as "s"). From this input, we construct a labeled undirected complete graph as shown in Figure 2(b). Each edge e has a set of labels $L(e) \subseteq L$ and cost $c(e)$, where $L = \{l_1, ..., l_n\}$ is the set of all labels and n is the number of sensor nodes. We determine $L(e)$ as follows: $l_i \in L(e)$ if node i's communication range intersects edge e. Intuitively, by moving along edge e, the data mule can collect data from the nodes whose labels are in $L(e)$.

Now we define the problem formally as follows:

LABEL-COVERING TOUR. Given an undirected complete graph $G = (V, E)$ where each vertex in $V = \{x_0, x_1, ..., x_n\}$ is a point in $\mathbf{R}^2$, a cost function on edges $c : E \rightarrow \mathbf{Q_0^+}$, a set $L = \{l_1, ..., l_n\}$ of labels, and a constant r. Each edge $e_{ij} \in E$ is associated with subset $L_{ij} \subseteq L$. For $k = 1, ..., n$, $l_k \in L_{ij}$ iff the Euclidean distance between x_k and an edge e_{ij} is equal to or less than r. A tour T is a list of points that starts and ends with x_0, allowing multiple visits to each point. A tour T is "label-covering" when it satisfies at least one of the followings for $k = 1, ..., n$: 1) $\exists e_{ij} \in T(E), l_k \in L_{ij}$, where $T(E)$ is the set of edges traversed by T, or 2) $dist(x_0, x_k) \leq r$, where $dist(x_i, x_j)$ is the Euclidean distance between x_i and x_j. Find a label-covering tour T that minimizes the total cost $\sum_{e_{ij} \in T(E)} c_{ij}$.

Unfortunately, this simplified problem is still $\mathcal{NP}$-hard.

Theorem 1. *LABEL-COVERING TOUR is $\mathcal{NP}$-hard.*

Proof. We show metric TSP is a special case. First we choose the cost function c to satisfy the triangle inequality (e.g., Euclidean distance). For a given set of points $V = \{x_0, ..., x_n\}$, by choosing a small r, we can make $dist(x_0, x_i) > r$ for all $i > 0$, $L_{ij} = \{l_i, l_j\}$ for all $i, j > 0$, and $L_{0j} = \{l_j\}$ for all $j > 0$ For such r, any label-covering tour must visit all the points. An optimal label-covering tour does not visit any point multiple times except x_0 at the start and the end of the tour, since in such cases, we can construct another label-covering tour with smaller total cost by "shortcutting". Therefore, an optimal label-covering tour is an optimal TSP tour for V. □

3.2 Choice of Cost Metric

In the definition of LABEL-COVERING TOUR, the cost c_{ij} is a critical parameter. In a restricted scenario, in which the data mule can either move at a constant speed or stop and no remote communication is used, Euclidean distance is the optimal cost metric in the sense that the shortest travel time is realized when the total path length is minimum. However, it is not clear for the general case in which the speed is variable under an acceleration constraint.

Since we minimize the total cost, and also we want to choose a tour that we can achieve the shortest travel time, a good cost metric should be strongly correlated with the travel time. Therefore we measure the goodness of cost metric by the correlation coefficient between cost and total travel time in the corresponding 1-D DMS problem. When the correlation is high, smaller cost implies shorter total travel time, and thus finding a minimum cost tour makes more sense.

We compare three different cost metrics that seem reasonable:

– Number of edges: $c_{ij} = 1$
– Euclidean distance: $c_{ij} = dist(x_i, x_j)$
– Uncovered distance: $c_{ij} = \sum_{s \subseteq e_{ij}, \forall k, dist(x_k, s) > r} |s|$, i.e., total length of intervals in edge e_{ij} that are not within the communication ranges of any nodes.

Uncovered distance is apparently a reasonable cost metric because it represents the total distance that the data mule "wastes", i.e., travels without communicating with any nodes.

Table 1. Correlation coefficients between total cost and total travel time for different cost metrics: 20 nodes, $a_{max} = 1$, $v_{max} = 10$

Radius (d)	150				500			
Comm. range (r)	10		100		10		100	
Exec. time (e)	2	20	2	20	2	20	2	20
Num. edge	0.992	0.987	0.982	0.850	0.984	0.982	0.988	0.988
Euclidean dist.	0.997	0.996	0.990	0.835	0.999	0.999	0.999	0.999
Uncovered dist.	0.992	0.993	—	—	0.999	0.999	0.935	0.935

Experimental Methods. We assume nodes are deployed in the circular area of radius d that has a start (i.e., point x_0) in the center. We randomly place other nodes within the circle so that they are uniformly distributed. For each edge connecting a pair of nodes, we assign a set of labels by calculating the distance from the line segment and each node.

For each of the node deployments, we randomly generate label-covering tours. A tour is generated by random walk which, at each point, chooses next point randomly, repeats this until all the labels are covered, and goes back to x_0. We measure the cost of the tour in three different cost metrics as listed above.

Using the tour, we transform the original problem to 1-D DMS problem and find a near-optimal latency by using the heuristic algorithm presented in [10]. We assume that each node has the same execution time e and the communication range r, and also that the speed of data mule needs to be zero at each point where it changes the direction[2]. For each node deployment, the cost and the total travel time are normalized among different random tours so that the mean is zero and standard deviation is one. For the collective set of data for same d, r, and e, we calculate the correlation coefficient between the normalized cost and the normalized total travel time for each cost metric.

For each (d, r, e), we generate 1000 examples, which consist of 20 random tours for each of 50 node deployments. We use $d = 150, 500$ and $r = 10, 100$ for LABEL-COVERING TOUR. For 1-D DMS, we use the execution time $e = 2, 20$, the maximum absolute acceleration $a_{max} = 1$, and the maximum speed $v_{max} = 10$. These parameter values roughly simulate data collection by a helicopter as in [7] by assigning units as follows: meters for d and r, seconds for e, m/s^2 for a_{max}, and m/s for v_{max}. Also the values of r are chosen to simulate the communication ranges of IEEE 802.15.4 and 802.11, respectively.

Results. Table 1 shows the correlation coefficients. Except one case, both number of edges and Euclidean distance had correlation coefficient above 0.98, and Euclidean distance had higher correlation than number of edges. In the exceptional case $(d, r, e) = (150, 100, 20)$, the correlation was weaker than other cases. This is most likely because the travel time is more influenced by the execution time rather than the moving time, since the deployment area is small relative to

[2] Otherwise, it would require infinite acceleration, since we assume a path consists of only line segments and not curves.

```
– Make a TSP tour $T$ using an exact or approximation algorithm for metric TSP
– Initialize $d[0] \leftarrow 0$, $d[1...n] \leftarrow +\infty$, $tour[0] \leftarrow \{T(0)\}$, and $tour[1...n] \leftarrow \emptyset$.
– For $i = 0$ to $n - 1$ do
    • For $j = i + 1$ to $n$ do
        ∗ Check if the line segment $T(i)T(j)$ is within the distance $r$ from each of
          the nodes $T(i + 1), ..., T(j - 1)$.
        ∗ If yes and $d[i] + |T(i)T(j)| < d[j]$, update the tables by $d[j] \leftarrow d[i] +$
          $|T(i)T(j)|$ and $tour[j] \leftarrow append(tour[i], T(j))$.
– Return $tour[n]$.
```

Fig. 3. Approximation algorithm for LABEL-COVERING TOUR: $T(i)$ is the i-th vertex that the tour T visits. $T(0)$ is the starting vertex.

the size of communication range and also the execution time is long. Uncovered distance had similar results but had no data when $(d, r, e) = (150, 100, 2)$ and $(d, r, e) = (150, 100, 20)$. These are the cases when the communication range is so broad that the total cost measured by uncovered distance is always zero.

These results suggest that number of edges and Euclidean distance are both appropriate metrics that precisely measure the goodness of paths. In the rest of the paper, we use Euclidean distance as the cost metric.

4 Approximation Algorithm for the Path Selection Problem

In the previous section we have formulated the path selection problem as LABEL-COVERING TOUR and shown its $\mathcal{NP}$-hardness. In this section, we design an approximation algorithm for this problem.

As discussed earlier, Euclidean distance is an appropriate cost metric. This enables us to design an approximation algorithm by using known algorithms for metric TSP where the triangle inequality holds. Figure 3 shows the approximation algorithm for LABEL-COVERING TOUR. It first finds a TSP tour T by using any algorithm (exact or approximate) for TSP. Then, using dynamic programming, it finds the shortest label-covering tour that can be obtained by applying shortcutting to T. For the dynamic programming, we use two tables $d[i]$ and $tour[i]$, where $tour[i]$ is the shortest path that is obtained by shortcutting T and covers the labels $T(0), ..., T(i)$, and $d[i]$ is the length of $tour[i]$.

4.1 Analysis

Computation time of the algorithm is $\mathcal{C}_{TSP} + O(n^3)$, where $\mathcal{C}_{TSP}$ denotes the computation time of the algorithm used for solving TSP.

Next we analyze the approximation factor of the algorithm. Let T_{OPT}, T_{APP} denote the optimal label-covering tour and the approximate label-covering tour,

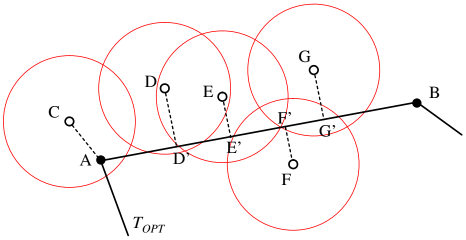


Fig. 4. Constructing a TSP tour from the optimal label-covering tour T_{OPT}: every non-visited point is within distance r from T_{OPT}

respectively. Total length of tour T is denoted as $|T|$. Also let α be the approximation factor of the TSP algorithm used in the first step of the approximation algorithm. Then we have the following theorem:

Theorem 2. $|T_{APP}| \leq \alpha(|T_{OPT}| + 2nr)$

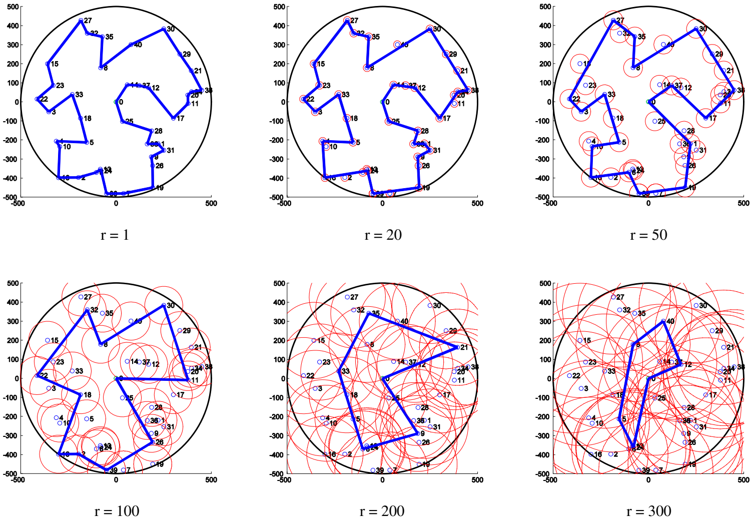
Proof. Clearly $|T_{APP}| \leq \alpha|T_{TSP}|$, where T_{TSP} is the optimal TSP tour. We give a lower bound to T_{OPT} by constructing another TSP tour by modifying T_{OPT}. Figure 4 shows the idea of construction. The points A and B (shown in filled circles) are visited by T_{OPT} and other points in the figure (shown in non-filled circles) are not. We call the former "visited points" and the latter "non-visited points". By the definition of label-covering tour, any non-visited points are within distance r from either a traversed edge or a visited point of a label-covering tour. For example in the figure, all of AC and DD', ..., GG' have the length less than r. Then we can construct a "tour"[3] that is identical to T_{OPT} but takes a detour to visit each non-visited point (e.g., ACAD'DD'...B). Since there are at most n non-visited points, total length of detour is at most $2nr$. This "tour" is easily converted to a shorter TSP tour by skipping all additional points (e.g., D', E', ...) and apply shortcutting so that each point is visited exactly once. Therefore, we have $|T_{OPT}| + 2nr \geq |T_{TSP}|$. The theorem follows by combining this and $|T_{APP}| \leq \alpha|T_{TSP}|$. □

5 Performance Evaluation

We evaluate the performance of the approximation algorithm by numerical experiments. We have implemented the algorithm in MATLAB. We use the same method and parameters as in Section 3.2 for test case generation, and use Concorde TSP solver[4] to find an optimal TSP tour. For each node deployment, we obtain a label-covering tour by running the approximation algorithm. Using the node deployment and the tour, we get a set of "location jobs", which is the

[3] This is not a tour in our definition because it does not consist of edges between the nodes.

[4] http://www.tsp.gatech.edu/concorde/index.html



r = 1

r = 20

r = 50

r = 100

r = 200

r = 300

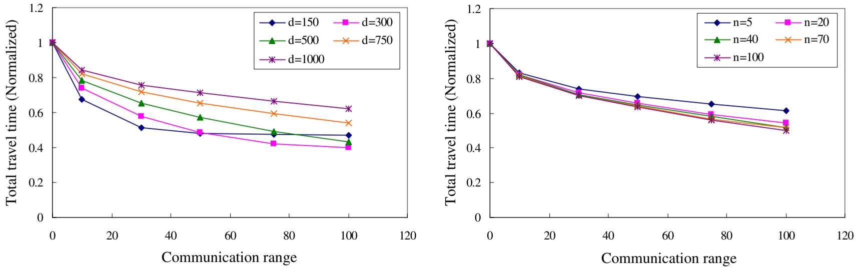
Fig. 5. Label-covering tours for different communication ranges: 40 nodes, $d = 500$; Path of data mule is shown in bold line

input to 1-D DMS problem. A location job is the notion defined in [10], and is intuitively understood as a real time job whose feasible interval is defined on 1-D location axis instead of time axis. Then we run the heuristic algorithm for 1-D DMS problem [10] and obtain the total travel time, which is near-optimal for the given tour. We use the total travel time as the evaluation metric.

Figure 5 shows some examples of label-covering tours for a node deployment with different communication ranges. As the communication range grows, the number of visited points becomes less and the path length becomes shorter.

5.1 Effect of Node Density and Network Size

Figure 6(a) shows the relation between the communication range and the total travel time for different node density. To see how broader communication range affects the travel time, we have normalized the total travel time by the one when the communication range is zero. The graph shows that the total travel time is reduced in all cases by up to 60% for this parameter set, suggesting the proposed problem formulation and algorithm altogether successfully exploit the breadth of communication range. The amount of reduction is bigger when the density is higher (i.e., smaller d), except the case of $d = 150$ for large communication ranges. This is because the total travel time is already very close to the lower bound, which is the product of the execution time and the number of nodes.



(a) Effect of node density

(b) Effect of number of nodes

Fig. 6. Comparison of total travel time for (a) different node density (40 nodes) and (b) different number of nodes ($d = 500$ for 20 nodes): $e = 10$, $a_{max} = 1$, $v_{max} = 10$

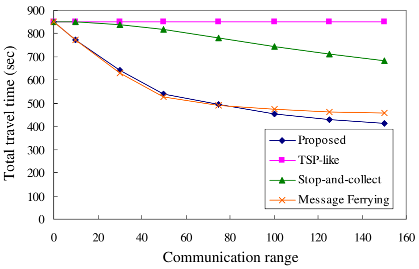


Fig. 7. Comparison of total travel time for different path selection algorithms: 40 nodes, $d = 500$, $e = 10$, $a_{max} = +\infty$, $v_{max} = 10$

Figure 6(b) shows the effect of number of nodes, varied from $n = 5$ to $n = 100$. We set d to 500 when $n = 20$, and changed d in proportion to $\sqrt{n}$ so that the density remains constant. The results show the reduction of total travel time for large communication ranges, but no big difference for different number of nodes.

5.2 Comparison with Other Strategies

Next we compare the travel time of our approximation algorithm with those of other algorithms as listed below.

- TSP-like: Based on the model used in [8]. Data mule visits all nodes. It stops at each node location to collect data and moves to the next node. While moving, the speed is constant at v_{max}. We use optimal TSP tours.
- Stop-and-collect: Based on the model used in [16]. Data mule takes a label-covering tour, as in our approximation algorithm. However, it stops to collect data when it is in the communication range of each node. While moving,

speed is constant at v_{max}. We find tours by using our approximation algorithm with optimal TSP tours[5].
- Message Ferrying: Based on the algorithm proposed in [17]. Data mule visits all nodes as in the TSP-like algorithm, but communication is also done while moving. Speed is variable between 0 and v_{max}. Speed and data collection schedule are determined by solving a linear program such that the total travel time is minimized. We use optimal TSP tours.

To allow direct comparison, we set $a_{max} = +\infty$ for our proposed approximation algorithm, since all other algorithms assume data mule can change its speed instantly. Note that when $a_{max} = +\infty$, we can obtain an exact solution for 1-D DMS problem by solving a linear program (see [10] for details).

Figure 7 shows the results for a representative case for 40 nodes. When the communication range is small, the travel time does not differ among the algorithms. As the communication range grows, Message Ferrying and the proposed algorithm show larger improvements than other two methods, and the proposed algorithm gets gradually better than Message Ferrying. When the communication range is 150, the proposed algorithm is nearly 10% better than Message Ferrying, 40% better than Stop-and-collect, and more than 50% better than TSP-like method.

When there is an acceleration constraint (i.e., $a_{max} \neq +\infty$), which none of these studies has addressed, the gaps between the proposed algorithm and others are expected to be larger. This is because all of these methods require the data mule to stop more frequently than the proposed algorithm does.

These results suggest that the proposed algorithm effectively exploits broader communication range for planning the path of the data mule.

6 Conclusions and Future Work

Controlled mobility for data collection in wireless sensor network provides flexibility in the trade-off between energy consumption and data delivery latency. With the goal of optimizing this trade-off, in this paper we focused on improving the latency in the pure data mule approach with the minimum energy consumption. The formulation of path selection problem, together with speed control and job scheduling problems, enables us to capture two-dimensional data mule scheduling problem under precise mobility and communication models. We have designed an approximation algorithm for the problem and experimentally demonstrated that it finds near-optimal paths and achieves much smaller latency compared to previously proposed algorithms.

Our ongoing work includes design of a hybrid approach that optimizes the energy-latency trade-off. The idea is to extend the problem framework shown in Figure 1 by adding "forwarding" subproblem before the path selection. In

[5] We could not use the path selection algorithm proposed in [16], since it has a restriction on the configuration of data mule and deployment area. Specifically, it assumes the data mule starts from the left end of the deployment area, travels toward the right end, and comes back to the initial position.

forwarding problem, we determine how each node forwards the data to other nodes within given energy consumption limit. Then the data mule collects data only from the nodes that have data after the forwarding. Since the results in this paper suggest the problem formulation along with the approximation algorithm allows significant reduction in latency, the hybrid approach is likely to achieve better trade-off between energy and latency compared to previous work.

References

1. Chakrabarti, A., Sabharwal, A., Aazhang, B.: Using predictable observer mobility for power efficient design of sensor networks. In: IPSN, pp. 129–145 (2003)
2. Ma, M., Yang, Y.: SenCar: An energy efficient data gathering mechanism for large scale multihop sensor networks. In: DCOSS, pp. 498–513 (2006)
3. Shah, R.C., Roy, S., Jain, S., Brunette, W.: Data MULEs: modeling a three-tier architecture for sparse sensor networks. In: SNPA, pp. 30–41 (2003)
4. Wang, W., Srinivasan, V., Chua, K.C.: Using mobile relays to prolong the lifetime of wireless sensor networks. In: MobiCom., pp. 270–283 (2005)
5. Kansal, A., Somasundara, A.A., Jea, D.D., Srivastava, M.B., Estrin, D.: Intelligent fluid infrastructure for embedded networks. In: MobiSys., pp. 111–124 (2004)
6. Vasilescu, I., Kotay, K., Rus, D., Dunbabin, M., Corke, P.: Data collection, storage, and retrieval with an underwater sensor network. In: SenSys., pp. 154–165 (2005)
7. Todd, M., Mascarenas, D., Flynn, E., Rosing, T., Lee, B., Musiani, D., Dasgupta, S., Kpotufe, S., Hsu, D., Gupta, R., Park, G., Overly, T., Nothnagel, M., Farrar, C.: A different approach to sensor networking for SHM: Remote powering and interrogation with unmanned aerial vehicles. In: Proceedings of the 6th International workshop on Structural Health Monitoring (2007)
8. Somasundara, A.A., Ramamoorthy, A., Srivastava, M.B.: Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines. In: RTSS, pp. 296–305 (2004)
9. Xing, G., Wang, T., Xie, Z., Jia, W.: Rendezvous planning in mobility-assisted wireless sensor networks. In: RTSS, pp. 311–320 (2007)
10. Sugihara, R., Gupta, R.K.: Data mule scheduling in sensor networks: Scheduling under location and time constraints. UCSD Tech. Rep. CS2007-0911 (2007)
11. Polastre, J., Hill, J., Culler, D.: Versatile low power media access for wireless sensor networks. In: SenSys, pp. 95–107 (2004)
12. Rhee, I., Warrier, A., Aia, M., Min, J.: Z-MAC: a hybrid MAC for wireless sensor networks. In: SenSys, pp. 90–101 (2005)
13. Ye, W., Heidemann, J., Estrin, D.: An energy-efficient MAC protocol for wireless sensor networks. In: INFOCOM, pp. 1567–1576 (2002)
14. Jea, D., Somasundara, A.A., Srivastava, M.B.: Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks. In: Prasanna, V.K., Iyengar, S.S., Spirakis, P.G., Welsh, M. (eds.) DCOSS 2005. LNCS, vol. 3560, pp. 244–257. Springer, Heidelberg (2005)
15. Gu, Y., Bozdağ, D., Brewer, R.W., Ekici, E.: Data harvesting with mobile elements in wireless sensor networks. Computer Networks 50(17), 3449–3465 (2006)
16. Ma, M., Yang, Y.: SenCar: An energy efficient data gathering mechanism for large-scale multihop sensor networks. IEEE Trans. Parallel and Distributed System 18(10), 1476–1488 (2007)
17. Zhao, W., Ammar, M.: Message ferrying: Proactive routing in highly-partitioned wireless ad hoc networks. In: FTDCS, pp. 308–314 (2003)