Data Gathering by Mobile Mules in a Spatially Separated Wireless Sensor Network

Fang-Jing Wu[1], Chi-Fu Huang[2], and Yu-Chee Tseng[1]
[1]Department of Computer Science, National Chiao Tung University, Taiwan
[2]Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan
Email:{fangjing, cfhuang, yctseng}@cs.nctu.edu.tw

Abstract

While wireless sensor networks (WSNs) are typically targeted at large-scale deployment, due to many practical or inevitable reasons, a WSN may not always remain connected. In this paper, we consider the possibility that a WSN may be spatially separated into multiple subnetworks. Data gathering, which is a fundamental mission of WSN, thus may rely on a mobile mule ("mule" for short) to conduct data gathering by visiting each subnetwork. This leads to the problem of minimizing the path length traversed by the mobile mule. We show that minimizing the path length, which may reflect the data gathering latency and the energy consumption of the mule is a generalization of the traveling salesman problem and is NP-complete. Some heuristics based on geometrical properties of node deployment are proposed. Our simulation results show that these heuristics perform very close to optimal solutions in most practical cases.

Keywords: computer geometry, communication holes, data gathering, mobile mule, traveling salesman problem (TSP), wireless sensor networks.

1. Introduction

The progress of embedded micro-sensing MEMS and wireless communications has made *wireless sensor networks (WSNs)* feasible. A WSN normally consists of many inexpensive sensor nodes deployed in a sensing field. Each node has the capability of sensing, collecting, processing, storing environment information and communicating with neighboring sensor nodes. Applications of WSNs include object tracking [9] [17], surveillance [13], and terrestrial ecology observing systems [1].

This paper considers the possibility that a WSN may become spatially separated into multiple subnetworks. We discuss how to utilize a mobile mule to visit these subnetworks to collect sensing data in an efficient way. Such separation may be due to several reasons. First, the sensing field might be huge, such as farms or mountain areas. Due to physical constraints (such as rivers), sometimes a WSN may naturally be separated. So having isolated subnetworks is sometimes inevitable and more cost-effective. Second, the WSN may be initially connected, but is partitioned due to emergencies, such as fires. Third, when random deployment is adopted, it may lead to a disconnected WSN. It is noted that the spatially separated deployment occurs frequently in some practical applications such as the terrestrial ecology observing systems [1]. However, coordination between spatially separated subnetworks is necessary. For example, to monitor long-term terrestrial ecology of animals and plants, cables, satellites, or mobile mules may be used to connect these subnetworks. Fig. 1 shows a WSN composed of several spatially separated subnetworks, where a helicopter is used as the mobile mule to connect these isolated subnetworks. Most related works have focused in connected WSNs. When there exist communication holes in a WSN, mobile sensors are utilized to cover these holes to provide better coverage or connectivity [15] [21]. It is still assumed that the WSN remains connected. How to bypass communication holes is discussed in [3] [4] [8]. Using mobile nodes to enhance connectivity between access points and sensors in a sparse WSN is addressed in [6] [12]. Random mobility is assumed for these mobile nodes. To mitigate data loss in a disconnected WSN due to memory overflow, [10] discusses how to redistribute data to improve storage utilization.

Data gathering is a fundamental issue in WSNs under our environment, the data gathering latency consists of the latency for the mule to travel between subnetworks and the latency for uploading data from each subnetwork. This work will focus on the former issue (the latter is relatively much smaller and has been intensively researched [11] [14] [16]). We consider the data gathering scenario in Fig. 1, where a mobile mule will start from any one node v_s in the subnetwork in which sink node is located, traverses any node of each isolated subnetwork to gather sensing data, and returns to the any node v_e in the sink subnetwork to deliver the gathered data to sink. It is noted that the data gathering will be conducted round by round, and for each round, the mobile mule will start from the node where it returned last round. The traveling path of the mobile mule may reflect the data gathering latency and the energy consumption of the mule. In this paper, we show that minimizing the per-round total traversal time of the mule can be formulated as a *MPDG (minimum-path data-gathering)* problem, which is a generalization of the Euclidean Traveling Salesman Problem (ETSP) [7]. So the MPDG problem is NP-complete. MPDG differs from ETSP in that it is sufficient to pick *any* landing node in each subnetwork and the traversal path is not necessarily a cycle (i.e., the staring and ending nodes of the traversal path are not necessarily the same). Fig. 1 illustrates two traversal paths for the helicopter (marked by solid and dashed arrows). It has been shown that when nodes are in a two-dimensional Euclidean space and the costs of edges can be reflected by Euclidean distances, the convex hull of nodes in ETSP has some properties closely related to the optimal solution, thus providing a good start for traversal path construction and path improvement [5]. We will propose some heuristics to solve the MPDG problem based on convex hulls. In addition, since MPDG allows us to traverse any node in each subnetwork, this also provides us opportunities to conduct further path improvement heuristics from the geometry properties of convex hulls.

Using mobile mules/ferries with intentional mobility in a highly disconnected wireless ad hoc network is discussed in [18] [19] [20]. In [20], there is a ferry moving along a publicly known route. With knowledge of the ferry route, nodes can proactively compute their transmission/reception with the ferry. An optimization problem is to find a ferry route such that the average message delay is minimized and the requirement of the transmission time for each node could be met. Two different ferrying schemes are developed in [18]. One allows nodes to periodically move closer to the ferry route. The other allows nodes to request the ferry, via high-power radios, to approach them when they have intention to transmit/receive. The optimization goal is to minimize the message drops. Multiple ferries are considered in [19]. A packet may be relayed by multiple ferries before reaching its destination. However, these works all assume that each node must be visited by a ferry, while our work assumes that it is sufficient to visit one representative node in each subnetwork.

The rest of this paper is organized as follows. Section 2 describes ours system model. We present our heuristics in section 3. Simulation results are in section 4. Section 5 concludes this paper.

2. System Model

A *spatially separated WSN (SS-WSN)* is modeled as an undirected graph $G = (V, E)$, where V is the set of sensor nodes and E is the set of communication links. The location of each sensor node v_i is denoted by (x_i, y_i) and assumed to be known in advance. In practice, we may conduct a subnetworks discovery procedure on the whole sensing field to obtain locations of sensor nodes. The Euclidean distance between any two nodes v_i and v_j is denoted by $d(v_i, v_j)$. We assume that the SS-WSN is partitioned into multiple node-disjoint subnetworks, $G_1, G_2, ..., G_n$. Each subnetwork is connected. A special node called *sink* is responsible for collecting sensing data. There is a mobile mule responsible for communications among these subnetworks.

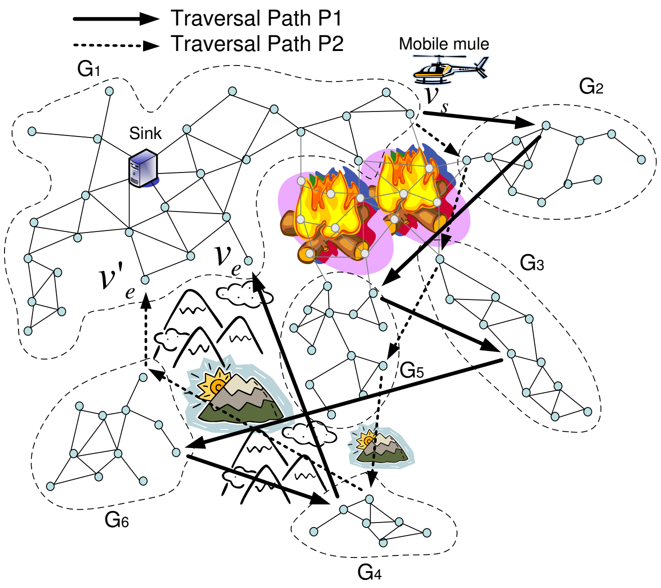


Figure 1. An example of the MPDG problem in a spatially separated WSN.

2.1. Formation and Operation of SS-WSN

There are three fundamental operations that a SS-WSN has to address. This paper will focus on the data gathering issue, and the other two can be found in our related works.

- Subnetworks discovery: If the locations of sensor nodes and subnetworks are not known in advance, the mobile mule has to traverse the whole sensing field along an exploring path to discover nodes' locations. This procedure has to be executed in the initial deployment and whenever the sink detects that some part of the network has been accidentally destroyed. When exploring the sensing field and whenever a sensor node is encountered, the mule can instruct the sensor node to communicate with the other nodes inside the same subnetwork to discover theses nodes. So a subnetwork is discovered once any one node in this subnetwork discovered by the mule. This opens up an opportunity for the mule to continuously optimize its initial exploring path until all subnetworks are discovered. The related optimization techniques can be addressed in our forthcoming papers.
- Data gathering by the mobile mule: From time to time, mule needs to conduct data gathering in the SS-WSN by traversing each subnetwork. The subnetwork where the sink is located is called the *starting subnetwork* G_s. The mobile mule is assumed to have unlimited storage and is initially located in a *starting node* v_s in G_s. Its responsibility is to leave from v_s, visit any node (called *landing port*) in each subnetwork other than G_s, and return to any node (called *ending node* v_e) in G_s. It

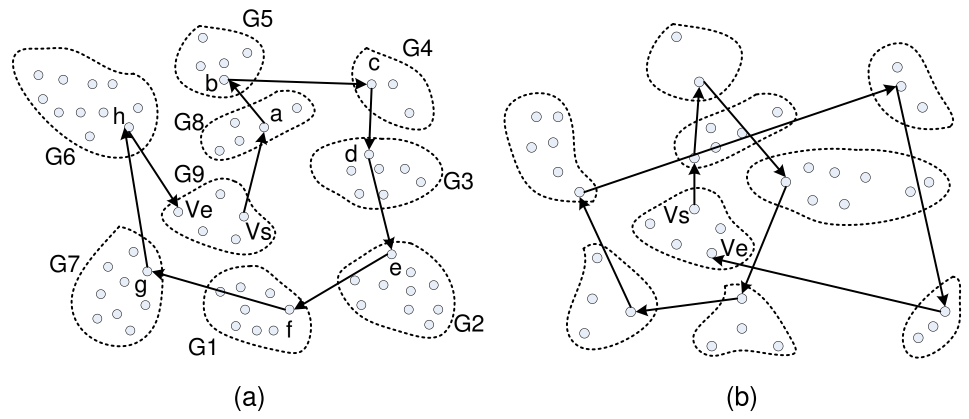


Figure 2. Examples of the greedy algorithm.

is noted that v_s and v_e are not necessarily the same. When the mule is within the communication range of a landing port, the latter can collect all sensing data in its subnetwork and forward to the mule. When returning to v_e, the mobile mule will relay all sensing data to the sink via the starting subnetwork. Our goal is to minimize the total traversal distance of the mule. This process can be repeated iteratively round by round and the ending node will become the starting node in the next round. Fig. 1 shows an examples with $G_s = G_1$.

- Data collection by landing ports: Assuming that the storage of sensor nodes is limited and the mule may not visit subnetworks frequently, we can regard the memory spaces of sensor nodes in each subnetwork as a distributed storage system. How to keep the more important data nearby the landing port of a subnetwork is a challenging issue. A novel distributed heapsort-like storage management scheme is proposed in our forthcoming papers.

2.2. Minimum-Path Data-Gathering (MPDG) Problem in a SS-WSN

Definition 1. *Given a graph $G = (V, E)$, v_s, and G_s, the Minimum-Path Data-Gathering (MPDG) problem is to find a gathering path P, starting from v_s, connecting one landing port in each subnetwork, and returning to an ending node in G_s such that the path length $|P|$ is minimized.*

3. Heuristics for the MPDG Problem

Below, we propose two heuristic algorithms for the MPDG problem. The first one tries to select the next landing port in a greedy way. The second one is derived based on a convex hull concept proposed in [5].

3.1. Greedy Algorithm

Initially, the mule is located at v_s. In this greedy algorithm, the mule will choose the next node, say, x to be visited such that x is located in an unvisited subnetwork and x is closest the current location of the mule. This process is repeated until all subnetworks are visited. Finally, the ending node will be the one in G_s which is closest to the last visited node. This forms the path P.

Fig. 2 shows two examples. It is to be noted that this greedy algorithm may find a path with intersections, as shown in Fig. 2(b). This should be avoided, as to be shown later.

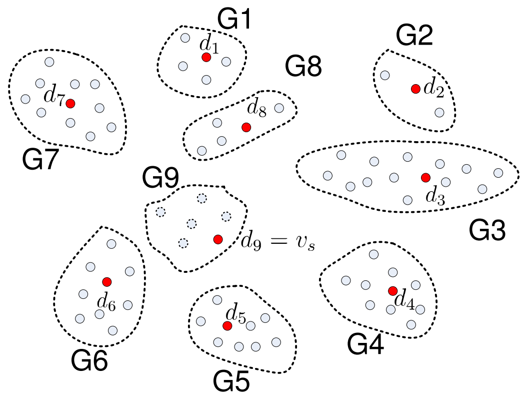
3.2. Convex Hull-Based Algorithm

This algorithm is designed based on selecting a delegation node in each subnetwork and constructing a convex hull from these delegations for continuous polishment. To start with, we first extend a property in [7].

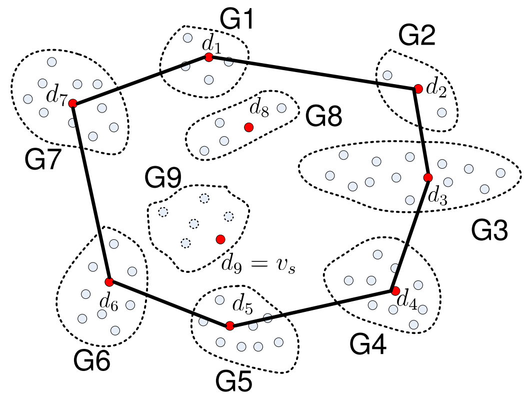
Theorem 1. *An optimal gathering path for the MPDG problem has no intersection with itself.*

Proof: Referring to Fig. 3, an intersection is defined as two line segments that cross each other in a 2D plane. We assume that path $P = v_s \rightarrow ... \rightarrow v_i \rightarrow v_{i+1} ... \rightarrow v_k \rightarrow v_{k+1} ... v_e$ is an optimal solution such that $\overline{v_i v_{i+1}}$ and $\overline{v_k v_{k+1}}$ intersect each other. However, we can find another gathering path $P' = v_s \rightarrow ... \rightarrow v_i \rightarrow v_k \rightarrow ... \rightarrow v_{i+1} \rightarrow v_{k+1} ... \rightarrow v_e$ such that $|P'| < |P|$ by the triangle inequality. It is a contradiction. So this theorem is proved. □

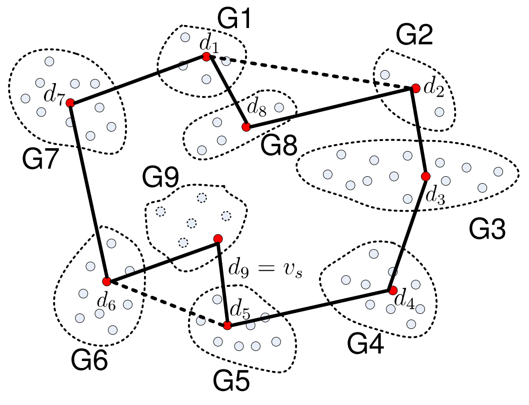
In [5], [7], it proves that the order in which the nodes on the boundary of the convex hull of ETSP's input nodes must appear in the optimal solution of the ETSP. In addition, the convex hull never intersects with itself. These observations motivate us to conduct path construction and path improvement based on convex hulls. Below, we modify the algorithm in [5] into one fitting our need.



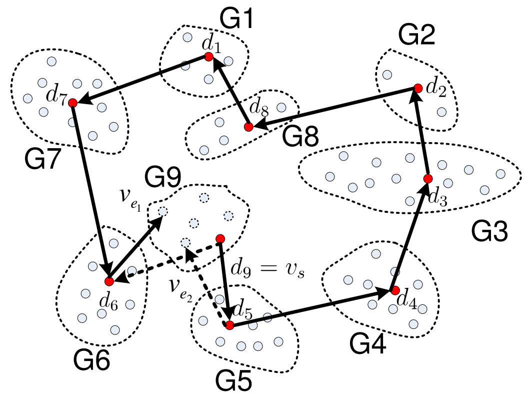
(a) delegation node selection.



(b) convex hull construction.



(c) insertion of delegation nodes.



(d) ending node selection.

Figure 4. An example of the convex hull-based algorithm.

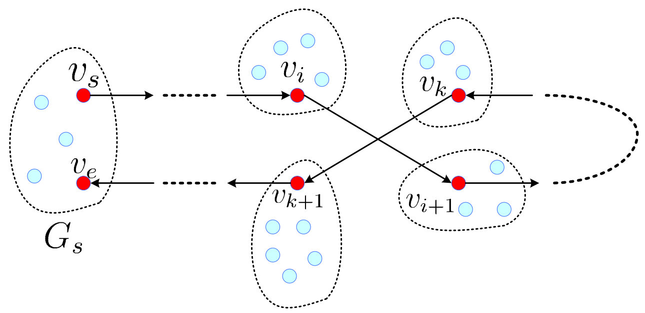


Figure 3. The proof of Theorem 1.

1) For each subnetwork $G_i \neq G_s$, compute a delegation node d_i, where d_i is the node closest to the center-of-gravity of G_i, i.e., $(\frac{\sum_{(x_i,y_i)\in G_i} x_i}{|G_i|}, \frac{\sum_{(x_i,y_i)\in G_i} y_i}{|G_i|})$. These delegation nodes will be the landing ports.
2) Let $D = \{v_s\} \cup \{d_i | G_i \neq G_s\}$. Following the convex hull property of ETSP, we construct a convex hull of D (such algorithms can be found in [2]). Let the convex hull be P.
3) We iteratively add nodes in $D - P$ into the traversal path. Specifically, in each iteration, for each node $d_i \in D - P$, we try to insert d_i into each link (x, y) of P. The insertion cost of putting d_i between x and y is $cost(x, d_i, y) = d(x, d_i) + d(d_i, y) - d(x, y)$. Then d_i will be inserted between link (x, y) such that $cost(x, d_i, y)$ is minimum. Let P be the new path after the insertion. We will repeat this process until all delegation nodes are included.
4) We will select v_e as follows. Let the current P after step 3 be $P = v_s \rightarrow d_{p_1} \rightarrow ... \rightarrow d_{p_{n-1}}$. We compute two paths $P_1 = v_s \rightarrow d_{p_1} \rightarrow ... \rightarrow d_{p_{n-1}} \rightarrow v_{e_1}$ and $P_2 = v_s \rightarrow d_{p_{n-1}} \rightarrow ... \rightarrow d_{p_1} \rightarrow v_{e_2}$, where v_{e_1} and v_{e_2} are nodes in G_s which are closest to $d_{p_{n-1}}$ and d_{p_1}, respectively. The final path P will be the one of P_1 and P_2 with a shorter total length.

Fig. 4 shows an example. Fig. 4 (a) identifies the delegation nodes after step 1. Fig. 4 (b) shows the convex hull after step 2. Fig. 4 (c) shows that the insertion of three delegation nodes (the insertion order is d_9 and d_8) after step 3. Fig. 4 (d) shows P_1 and P_2. The final result is P_1.

3.3. Discussion: Complexity Analysis and Some Local Optimizations

Let n be the number of subnetworks and $N = |V|$ be the number of nodes. Normally, $N \gg n$. In the greedy algorithm, each iteration takes $O(N)$ time to find the next landing port, and there are n iterations. So its time complexity is $O(Nn)$.

For the convex hull-based algorithm, step 1 takes $O(N)$ time to compute the delegation nodes. Finding a convex hull in step 2 takes $O(nm)$ time by Jarvis's march [2], where $m \leq n$ is the number of nodes of the convex hull. In step 3, at most $O(n)$ unvisited delegation nodes will be checked in each iteration and there are $O(n)$ iterations. Also, all links in P will be checked, so the cost is $O(n^3)$. Step 4 costs $O(N)$. So the total complexity is $O(N + n^3)$.

4. Simulation Results

A simulator has been developed by JAVA. To simulate a SS-WSN, we randomly generate N sensor nodes in a $S \times S$ m^2 field. The field is divided into 100 grids, each of size $S/10$ m^2. Each sensor node has a transmission range of 20 m. Each grid has a failure probability of P_f. If a grid is determined to fail, all sensor nodes inside the grid fail. This would partition the network into multiple subnetworks. G_s and v_s are randomly selected. Our simulation results are all from the average of 1000 runs.

We compare the performance of the proposed heuristics. The main performance metric is the traversal length of the mule. We vary factors such as N, network size S, fail probability P_f, and the number of subnetworks.

First, we investigate the effect of the number of nodes. Fig. 5(a) shows that the convex hull-based algorithm perform the best. As can be seen, when the number of nodes N is relatively small, the traversal length will be relatively longer because there are a lot of subnetworks. As N increases, the path length decreases gradually. Second, we vary the network size S. Fig. 5(b) shows that the traversal length increases proportionally with S. There are two factors contributing to this: (i) there are more subnetworks, and (ii) the distances between subnetworks are relatively longer. Third, we vary the fail probability P_f. Fig. 5(c) shows that the convex hull-based algorithm perform much better than greedy heuristic when the number of subnetworks is relatively larger ($P_f = 0.1 \sim 0.7$). When there are less subnetworks ($P_f = 0.8 \sim 0.9$), the gaps between these schemes shrink. So we conclude that the convex hull-based algorithm are more useful when there are more subnetworks, in which case a badly designed scheme may incur serpentine-like traversal paths. Finally, we compare our heuristics against an exhaustive search algorithm by varying the number of subnetworks. Unfortunately, only a small number of subnetworks (around 10) can be computationally handled by an exhaustive scheme (note that a subnetwork may contain a lot of sensor nodes, which also have to be searched.) Fig. 5(d) shows that the convex hull-based heuristic perform very closely to the optimal scheme.

5. Conclusions

In this paper, we consider a new SS-WSN topology which contains multiple separated subnetworks. We discuss the issue in a SS-WSN with the support of a mobile mule. We formulate the MPDG problem and propose heuristics based on geometrical properties. Our simulation results show that these algorithms perform very close to optimal ones in most practical cases.

Acknowledgment

Y.-C. Tseng's research is co-sponsored by MoE ATU Plan, by NSC grants 95-2221-E-009-058-MY3, 96-2218-E-009-004, 96-2219-E-007-008, 97-3114-E-009-001, 97-2221-E-009-142-MY3, and 97-2218-E-009-026, by MOEA under grant 94-EC-17-A-04-S1-044, by ITRI, Taiwan, and by III, Taiwan.

References

[1] Terrestrial ecology observing systems. http://research.cens.ucla.edu/.

[2] T. H. Cormen. Introduction to Algorithms. The MIT Press, 2001.

[3] Q. Fang, J. Gao, and L. J. Guibas. Locating and bypassing routing holes in sensor networks. In *INFOCOM*, pages 2458 – 2468, 2004.

[4] Q. Fang, J. Gao, L. J. Guibas, V. de Silva, and L. Zhang. GLIDER: gradient landmark-based distributed routing for sensor networks. In *INFOCOM*, pages 339 – 350, 2005.

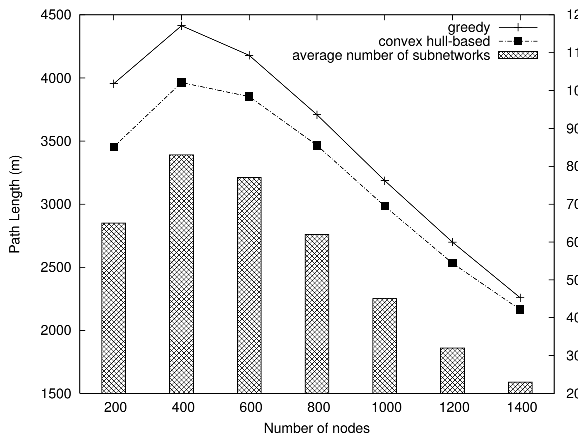
[5] B. Golden, L. Bodin, T. Doyle, and W. S. Jr. Approximate Traveling Salesman Algorithms. *Operations research*, 28(3):694–711, 1980.

[6] S. Jain, R. C. Shah, W. Brunette, G. Borriello, and S. Roy. Exploiting mobility for energy efficient data collection in wireless sensor networks. *Mobile Networks and Applications*, 11(3):327 – 339, 2006.

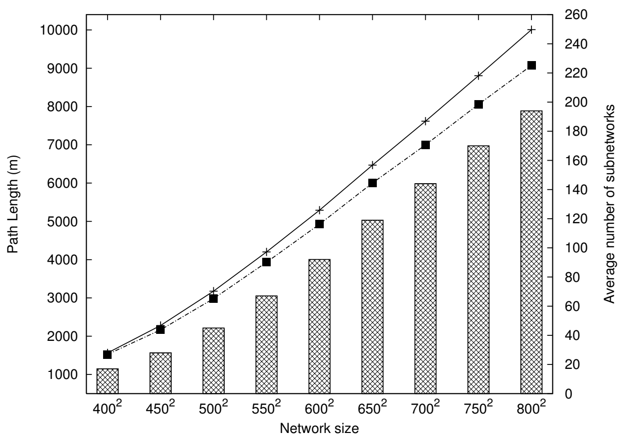
[7] R. C. Larson and A. R. Odoni. Urban Operations Research. Prentice-Hall, 1981.

[8] M. Li and Y. Liu. Rendered path: Range-free localization in anisotropic sensor networks with holes. In *MOBICOM*, 2007.

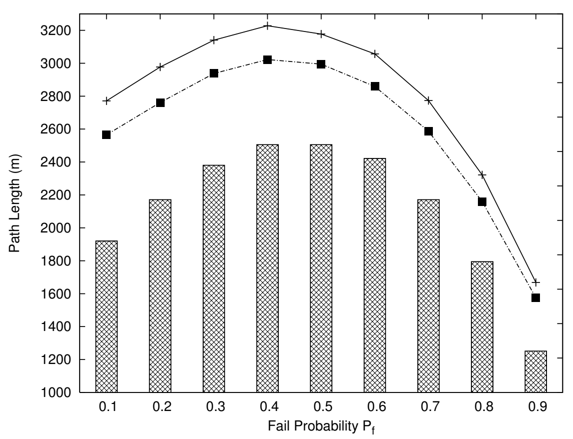
[9] C.-Y. Lin, W.-C. Peng, and Y.-C. Tseng. Efficient in-network moving object tracking in wireless sensor networks. *IEEE Transactions on Mobile Computing*, 5(8):1044 – 1056, 2006.



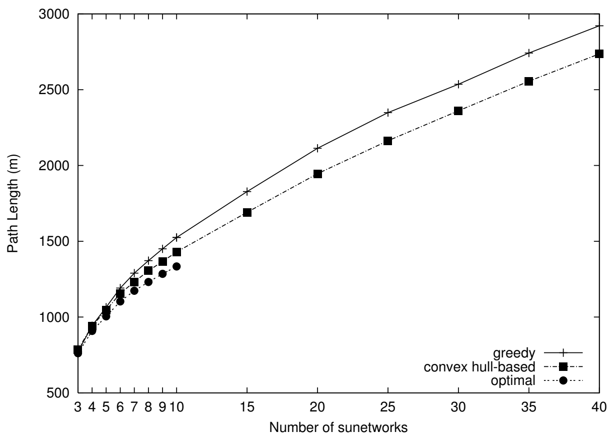
(a)$N = 200 \sim 1400$, $S = 500$, and $P_f = 0.5$



(b)$N = 1000$, $S = 400 \sim 800$, and $P_f = 0.5$



(c)$N = 1000$, $S = 500$, and $P_f = 0.1 \sim 0.9$



(d)$N = 1000$, $S = 500$, $P_f = 0.5$, $n = 10 \sim 40$

Figure 5. Simulation results when there is one mule.

[10] L. Luo, C. Huang, T. Abdelzaher, and J. Stankovic. EnviroStore: A cooperative storage system for disconnected operation in sensor networks. In *INFOCOM*, 2007.

[11] M.-S. Pan and Y.-C. Tseng. Quick convergecast in ZigBee beacon-enabled tree-based wireless sensor networks. *Computer Communications*, 31(5):999–1011, 2008.

[12] R. C. Shah, S. Roy, S. Jain, and W. Brunette. Data MULEs: modeling a three-tier architecture for sparse sensor networks. In *Proc. of IEEE Workshop on Sensor Network Protocols and Applications (SNPA)*, pages 30–41, 2003.

[13] Y.-C. Tseng, Y.-C. Wang, K.-Y. Cheng, and Y.-Y. Hsieh. iMouse: An Integrated Mobile Surveillance and Wireless Sensor System. *IEEE Computer*, 40(6):60–66, 2007.

[14] S. Upadhyayula, V. Annamalai, and S. Gupta. A low-latency and energy-efficient algorithm for convergecast in wireless sensor networks. In *GLOBECOM*, pages 3525 – 3530, 2003.

[15] G. Wang, G. Cao, T. L. Porta, and W. Zhang. Sensor relocation in mobile sensor networks. In *INFOCOM*, pages 2302 – 2312, 2005.

[16] Y. Yu, B. Krishnamachari, and V. Prasanna. Energy-latency tradeoffs for data gathering in wireless sensor networks. In *INFOCOM*, 2004.

[17] W. Zhang and G. Cao. DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks. *IEEE Transactions on Wireless Communications*, 3(5):1689–1701, 2004.

[18] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MOBIHOC*, 2004.

[19] W. Zhao, M. Ammar, and E. Zegura. Controlling the mobility of multiple data transport ferries in a delay-tolerant network. In *INFOCOM*, 2005.

[20] W. Zhao and M. H. Ammar. Message ferrying: Proactive routing in highly-partitioned wireless ad hoc networks. In *Proc. of the 9th IEEE Workshop on Future Trends in Distributed Computing Systems*, 2003.

[21] Y. Zou and K. Chakrabarty. Sensor deployment and target localization based on virtual forces. In *INFOCOM*, pages 1293 – 1303, 2003.