Optimized Interconnection of Disjoint Wireless Sensor Network Segments Using K Mobile Data Collectors

Fatih Senel and Mohamed Younis
Dept. of Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore, MD 21250
{fsenel1,younis}@cs.umbc.edu

Abstract—Due to harsh environmental conditions a Wireless Sensor Network (WSN) may suffer from large scale damage where many nodes fail simultaneously and thus the network gets partitioned into several disjoint network segments. Restoring inter-segment connectivity is essential to avoid negative effects on the application. Employing mobile data collectors (MDCs), which —by repositioning— facilitate the establishment of communication links between segments, may provide flexible solution to this problem. However the problem of finding shortest tours for MDCs is NP-Hard. In this paper we study the problem under constrained number of MDCs which makes the problem more challenging. We present a polynomial time heuristic for Interconnecting Disjoint Segments with k MDCs (IDM-kMDC). IDM-kMDC opts to minimize the tour lengths and balance the load on the k available MDCs. We model each segment by a representative. The IDM-kMDC heuristic finds k-subsets of representatives, computes an optimized tour for each subset and assigns one MDC for each tour. The performance of the algorithm is validated through simulation.

Keywords: Interconnecting disjoint network Segments, Recovery from multiple nodes failure, Mobile data collectors

I. INTRODUCTION

In recent years, a wide range of applications of Wireless Sensor Networks (WSNs) has attracted interest from the research and engineering communities. Most notable among these applications are those serving in inhospitable environment such as border protection, combat field surveillance, and search and rescue. In these applications a set of sensor nodes are deployed to an area of interest to collaboratively monitor certain events in the vicinity and transmit the readings to a base station (BS) or a command node. By deploying these sensors in harsh environments it would be possible to reduce the cost of the application and avoid risk of human life. Upon their deployment sensors are expected to form a connected network to coordinate their actions in the execution of a task and forward the collected data to the BS.

In harsh environments, a WSN may suffer from large scale failures where many nodes die simultaneously. For example some sensors may be wiped out due to flood or destroyed by enemy explosive in a combat field. In such a case, the WSN gets partitioned into disjoint segments and its services become very limited. To restore connectivity, proactive strategies, such as deploying redundant nodes at network setup [1], will not be effective as the damage may affect redundant nodes as well. In addition, reactive strategies such as repositioning some nodes [2][3] may not be feasible as the scope of the damage is so wide.

One of the viable solutions for restoring network connectivity is to deploy additional stationary relay nodes (RNs) to form multi-hop path between network segments. A RN is a more powerful device than a sensor in terms energy reserve, processing and communication capabilities. Relay node placement problem has been well studied in the literature. The problem is shown to be equivalent to finding Steiner Minimum Tree with Minimum Steiner Points and Bounded Edge Length (SMT-MSPBEL) which is an NP-Hard problem. Depending on the scale of the damage, however, approximate solutions for SMT-MSPBEL [4][5] may require many RNs for restoring the connectivity which may not be feasible to implement due to limitation in the available resources.

Instead of using stationary RNs, we propose to employ mobile data collectors (MDCs) as mobile relay nodes which provide more flexible solution with a reduced node count. A MDC facilitates the establishment of intermittent communication links between network segments. A MDC has to move inside the transmission range of a sensor, in order to receive the data. After receiving the data, it starts moving to its next stop to collect from another sensor. Every MDC has a touring path such that the network segments along the path are visited regularly and thus the collected data is carried from one segment to the other towards the destination.

This paper investigates the problem of finding shortest tours for constrained number of MDCs and presents a polynomial time heuristic for Interconnecting Disjoint Segments with k MDCs (IDM-kMDC). We model each network segment by a representative that will be visited by at least one MDC. The objective of IDM-kMDC is to minimize the total tour lengths and balance the travel load among the MDCs. IDM-kMDC is a greedy approach which runs in rounds. The algorithm first computes Minimum Spanning Tree (mst) edges and assigns one MDC for each mst edge. In each round the algorithm picks two collocated MDCs and merges their tours into one. Therefore the number of MDCs is reduced by one in each round. The algorithm iterates until the number of available MDCs is met.

This paper is organized as follows. The next section discusses related work and highlights the distinct features of IDM-kMDC. Section III describes the assumed system model and formally defines the problem. In section IV, we describe the IDM-kMDC algorithm in detail. Section V presents the simulation results and finally Section VI concludes the paper.

497
II. RELATED WORK

Employing mobile elements has been pursued to enhance the network performance. Most of the published work has focused on minimizing the energy consumption in forwarding the data. For example, in [6] and [7], the authors have proposed energy efficient data collection protocols in single hop WSNs by employing mobile sinks moving on a fixed path. Meanwhile, in [8][9][10] the same problem has been tackled for multi-hop WSNs. The common objective of these efforts is to define an energy-efficient protocol to disseminate the data of each sensor towards the mobile sink. Unlike these publications, we focus on finding the tours for MDCs in order to establish intermittent communication links between isolated segments.

The authors of [11][12] have studied a similar problem to what we tackle in this paper. Their main aim is to find tours for mobile elements in order to improve the network lifetime. In [11], Ekici at al., have presented an algorithm that strives to minimize the data loss rates due to buffer overflow for a specific data generation frequency. Our approach is different in the sense that we do not consider data generation rates as a design issue. In [12], Alsalih et al., presented a MDC placement approach to provide energy efficiency in highly energy constrained sensor networks. Sensors are not assumed to spatially distant from each other and their transmission intersects. Using the intersection points, the authors formulate the problem as Mixed Integer Linear Program (MILP) to find an optimal solution. However, especially when the scope of the damage is so wide, the nodes that MDC is going to visit may be too far from each that their transmission disks do not intersect. Therefore, a MILP-based solution may not be feasible.

III. SYSTEM MODEL AND FUNDAMENTAL DEFINITIONS

A. System Model

In the context of this paper, a WSN is composed of a set of sensors scattered in an area of interest. A sensor is a miniaturized battery-operated device with limited processing and communication capabilities. Sensors collaboratively probe their vicinity, collect data from the environment and forward the collected data to a base station or a sink node through a multi-hop path. Therefore, inter-sensor connectivity is crucial for the application level interaction and routing. We assume that sensors are stationary and they communicate over a shared unidirectional wireless link. A wireless link can be established between every pair of nodes if they are within their radio range of each other. We assume that all sensors have same transmission range which is equal to “R”.

A mobile data collector (MDC) is a more powerful device in terms of energy reserve, better processing communication capabilities. Even though MDCs are equipped with sensing circuitry, their main objective is to collect the data from segments and establish intermittent communication links by carrying the data from one segment to another. Intuitively MDCs are more expensive than sensors and thus their availability may be limited.

B. Preliminaries and Problem Statement

In harsh environments sensors are susceptible to large scale failures which may split the network into multiple isolated islands of segments. The focus in this paper is on reconnecting these segments using a fixed number of MDCs. Before formally stating the problem we make the following definitions:

Definition 1: Let $s_i$ be the point where $i^{th}$ segment representative is located. The point $p_i$ is called a collection point of $s_i$ if $d(s_i, p_i) \leq R$ where $d(s_i, p_i)$ is the Euclidean distance between $s_i$ and $p_i$, and $R$ is the communication range of a sensor.

Definition 2: Let $S = \{s_1, s_2, ..., s_m\}$ be the set of $m$ points where segment representatives are located and also let $P = \{p_1, p_2, ..., p_m\}$ be the set of corresponding collection points where $p_i$ is a collection point of $s_i \forall i \in \{1, 2, ..., m\}$. A tour, which is denoted as $T_s$, is the shortest possible cycle that visits every point in $P$ once and returns to the starting point. The length of $T_s$ is denoted as $|T_s|$

The problem that we study in this paper can be formally defined as follows: “Given the set of $n$ representatives $S = \{s_1, s_2, ..., s_n\}$ and $k$ MDCs, determine the set of tours $\{T_{S_1}, T_{S_2}, ..., T_{S_k}\}$ such that $S_i \subseteq S, 1 \leq i \leq k$ and $S = \bigcup_{i=1}^{k} S_i$ where $\sum_{i=1}^{k} |T_{S_i}|$ is minimized.” We can split the problem into two sub-problems:

1) For a given $S_1$, how to calculate the shortest tour $T_{S_1}$

2) How to partition the set $S$ into $k$ different subsets of $\{S_1, S_2, ..., S_k\}$ where $S = \bigcup_{i=1}^{k} S_i$ and $\sum_{i=1}^{k} |T_{S_i}|$ is minimized.
The first sub-problem is equivalent to Euclidean Travelling Salesman Problem which is a well-known NP-Hard problem if the optimal set of collection points is known. Finding a set of collection points of \( S_t \) makes the problem more challenging. In addition, the second sub-problem is also NP-Hard even if we have an optimal solution for the first sub-problem. The proof is by reduction to the set covering problem. The next section describes how IDM-kMDC tackles these sub-problems.

IV. ESTABLISHING COMMUNICATION LINKS

A. Optimized Tour Computation

This section describes a heuristic for the first sub-problem in detail. The key part of the algorithm is finding the set of collection points for the set \( S = \{s_1, s_2, \ldots, s_m\} \). In other words, the MDC is going to move to \( p_i \) in order to collect the data from \( s_i \). However optimal collection point \( p_i \) may be located anywhere inside the transmission disk of \( s_i \). To find the set of collection points the algorithm considers two cases:

- **Case 1 (m=2):** This case is illustrated in Figure 1 (a). The collection points for \( p_1 \) and \( p_2 \) will be the intersection points of the corresponding transmission disks of \( s_1 \) and \( s_2 \) with the line connecting \( s_1 \) and \( s_2 \). This yields the optimal solution for two nodes, since a straight line is the shortest path connecting two points. The MDC tours in between \( p_1 \) and \( p_2 \). The length of the tour is twice the distance between \( p_1 \) and \( p_2 \).

- **Case 2 (m>2):** This case is more complicated than case 1. First the algorithm finds the largest subset of representatives which form a convex polygon and calculates its center of mass. The collection point \( p_i \) for the representative \( s_i \) is the intersection of the transmission disk of \( s_i \) with the lines connecting \( s_i \) and the center of mass. Figure 1 (b) illustrates the case with \( m=4 \). Collection points for non-convex representatives are computed at the end by finding the intersection of the transmission disk and the perpendicular line from the representative to the closest tour edge. The algorithm updates the list of tour edges with the newly calculated collection point. Figure 2 shows an example for this case. Let \( \{p_1, p_2, p_3, p_4\} \) be the collection points for the convex vertices of the representative set. Also let \( s_5 \) be a non-convex point where the transmission disk of \( s_5 \) does not intersect with the tour. In such a case, the algorithm draws a perpendicular from \( s_5 \) to the closest tour edge, which is the line passing from \( p_2 \) and \( p_3 \), in this example as shown in Figure 2 (a) (i.e. the two lines are perpendicular \( (p_2, p_3) \perp (s_5, O) \)). The collection point for the non-convex point \( s_5 \) is the intersection of the transmission disk of \( s_5 \) with the line \( (s_5, O) \). The algorithm modifies the tour according as shown in Figure 2(b).

B. IDM-kMDC Heuristic

This section describes the IDM-kMDC heuristic in detail. IDM-kMDC is a greedy algorithm which runs in rounds. In each round the algorithm identifies two MDCs and merges the two tours into one. Thus, each iteration reduces the number of MDCs by one, and the algorithm loops until the number of available MDCs is matched. Merging tours is performed by finding a new touring path for the union of two representative sets. For example let \( T_{s_i} \) and \( T_{s_j} \) be the two tours with the representative sets \( S_i = \{s_1, s_2\} \) and \( S_j = \{s_3, s_4, s_5\} \) as shown in Figure 3(a). The merged tour \( T' \) spans all the representatives in the union set of \( S_i \) and \( S_j \), and denoted as \( T' \leftarrow \text{Merge}(T_{s_i}, T_{s_j}) \) where \( S' = S_i \cup S_j \) as shown in Figure 3(b).

Although merging tours reduces the number MDCs, it increases the total travel distance. In other words, it can be said that the length of the merged tour is larger than the sum of the two tours which are merged. The reason is intuitive; basically, two MDCs will have to share the cost of collecting data from a set of representatives, while one MDC needs to span all nodes by itself. Therefore we define the cost of merging the tours \( T_{s_i} \) and \( T_{s_j} \) as the increase in total travel distance. The merging cost is denoted as:

\[
C(T_{s_i}, T_{s_j}) = |T_{s_i}| - (|T_{s_i}| + |T_{s_j}|) \quad \text{where} \quad S' = S_i \cup S_j
\]

In addition, merging cost also indicates the proximity of the tours. It is expected that merging cost of two far apart tours will be the combined tour significantly, as compared to merging two close tours. Due to this fact it is desired to merge closer tours to minimize the total tour lengths. Therefore IDM-kMDC uses merging cost as the greedy decision criteria while reducing number of tours iteratively.

The algorithm first calculates \( \text{mst} \) of the representatives. For each \( \text{mst} \) edge a MDC is assigned to tour between the vertices of the \( \text{mst} \) edge. Let \( \mathcal{H} \) be the set of all tours. Since there are \( n-1 \) \( \text{mst} \) edges for \( n \) representatives, \( \mathcal{H} \) initially contains \( n-1 \) tours. In each iteration, IDM-kMDC selects the two tours \( T_{s_{i_1}}, T_{s_{j_1}} \in \mathcal{H} \) having the least merging cost \( C(T_{s_{i_1}}, T_{s_{j_1}}) \) among all possible two-combinations in \( \mathcal{H} \). Then the algorithm removes the tours \( T_{s_{i_1}}, T_{s_{j_1}} \) from set \( \mathcal{H} \) and adds \( T' \) to \( \mathcal{H} \) where \( S' = S_i \cup S_j \), to the set \( \mathcal{H} \). IDM-kMDC loops until the size of the set \( \mathcal{H} \) is equal to \( k \) where \( k \) is the number of available MDCs.

C. Illustrative Example

Figure 4 illustrates how the proposed IDM-kMDC heuristic works. Given that there are 7 representatives and 3 MDCs, the algorithm needs to find 3 tours to collect data from all representatives. Initially IDM-kMDC forms the \( \text{mst} \) of the representatives and assigns one MDC for each \( \text{mst} \) edge as shown in Figure 4 (a). Let \( \mathcal{H} = \{T_{s_{i_1}}, T_{s_{i_2}}, T_{s_{i_3}}, T_{s_{j_1}}, T_{s_{j_2}}, T_{s_{j_3}}\} \) be the initial tour configuration where \( S_1 = \{s_1, s_2\}, S_2 = \{s_3, s_4\}, S_3 = \{s_5, s_6\}, S_4 = \{s_7, s_8\}, S_5 = \{s_9, s_{10}\}, S_6 = \{s_{11}, s_{12}\} \) and \( S_7 = \{s_{13}, s_{14}\} \). The first iteration the algorithm lists all possible combinations of two
The IDM-kMDC algorithm is iterates tours. The complexity of each iteration is: iteration IDM-kMDC finds a pair of tours having the least cost of a given set of representatives can be bounded by the convex hull through simulation. This section discusses the simulation updates the set of tours as the size of the set. Let $T_{S_1}$ and $T_{S_2}$ be the tours which are selected in the second iteration. Figure 4(c) illustrates how the algorithm merges $T_{S_1}$ and $T_{S_2}$ into $T_{S_5}$. After merging the tours the algorithm updates the set of tours as $\mathcal{H} = \{T_{S_1}, T_{S_2}, T_{S_3}, T_{S_4}, T_{S_5}\}$, where $S_{S_5} = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$. It is also worth mentioning that IDM-kMDC does not consider $s_5$ in calculating the tour length as it is a non-convex point in the set $S_{S_5}$. In addition, since the tour for the convex points intersects with the transmission disk of $s_5$, no further modification is needed. Any point along the tour path and inside the transmission disk of $s_5$ can be selected as a collection point for $s_5$. In the last iteration the tours $T_{S_1}$ and $T_{S_2}$ are selected as they have the least merging cost. The algorithm merges the tours as illustrated in the Figure 4(d) and updates the set of tours as $\mathcal{H} = \{T_{S_{12}}, T_{S_3}, T_{S_4}, T_{S_5}\}$, where $S_{S_{12}} = \{s_1, s_2, s_3\}$. Since the number of available MDCs is equal to the size of the set $\mathcal{H}$, IDM-kMDC stops.

**Theorem:** The run time complexity of the IDM-kMDC algorithm is $O(n^4 \log n)$ where $n$ is the number of segments.

**Proof:** The run time complexity of finding the optimized tour for a given set of representatives can be bounded by the convex hull algorithm. The convex hull algorithm can be implemented in $O(n \log n)$ time using Graham scan’s algorithm. In each iteration IDM-kMDC finds a pair of tours having the least merging cost, by searching all possible combinations of two tours. The complexity of each iteration is: $O((n-1) \cdot n \cdot \log n)$, since there are $(n-1)$ mst edges which is equal to the initial number of tours.

Since the initial cardinality of the set $\mathcal{H}$ is $(n-1)$ and the final cardinality of the set $\mathcal{H}$ is $k$, the “while” loop in line 7 iterates $(n - k - 1)$ times. Therefore the complexity of the IDM-kMDC algorithm is

$$O((n - k - 1) \cdot n^2 \cdot n \log n) = O(n^4 \log n)$$

Since $k < n$, $O((n - k - 1) \cdot n^2 \cdot n \log n) = O(n^4 \log n)$.$\blacksquare$

V. PERFORMANCE EVALUATION

The performance of the IDM-kMDC algorithm is validated through simulation. This section discusses the simulation environment, performance metrics and simulation results.

A. **Experiment Setup and Performance Metrics**

In the simulation, segment representatives are randomly deployed to a 1500m x 1500m area. The number of representatives varies from 3 to 15. The transmission range ($R$) of the representatives and the MDCs is fixed to 100m. The number of available MDCs is set to the number of stationary RNs which is required to fill the gaps along the mst edges, multiplied by constant number. The number of stationary RNs required $m$ can be computed from the following formula:

$$m = \sum_{(u,v) \in E} \left( \left\lfloor \frac{\|uv\|}{R} \right\rfloor - 1 \right)$$

where $E$ is the mst edges. In our simulation we set the number of available MDCs using the formula: $k = m \cdot \varphi$ where $k$ is the number of available MDCs and $0 < \varphi < 1$. In order to study the performance of the algorithm under varying $k$, we repeated the experiment for different $\varphi$ values (0.2, 0.3, and 0.4), instead of changing $k$ directly. The reason is that, depending on the topology the representatives may be located too far or too close to each other, which directly affects the total tour lengths. Due to this fact, employing the same number of MDCs for two different topologies may yield unfair comparison. In order to handle such cases we have varied $\varphi$ rather than $k$.

In the simulation experiments the following performance metrics are considered:

- **Total Tour Lengths (TTL):** This is the sum of all tour lengths. Obviously minimizing TTL is desired to improve the lifetime of MDCs.
- **Maximum Tour Length (MTL):** This metric report the longest tour that a MDC will have to make. If MTL is too large, it might take too long for a MDC to complete one tour and collect all the data, which increases the data collection latency and also boosts the travel overhead that the corresponding MDC incurs. It may also cause bandwidth and memory problems for a MDC if the volume data is large. Therefore it is desired minimize MTL.
- **Standard Deviation of Tour Lengths:** This metric captures fairness. If all tours are close to each other in length, it means that the load is well shared among the MDCs. Thus low standard deviation is desired.

B. **Baseline for comparison**

We compare IDM-kMDC with a variant that we refer to as the exponential set partitioning (ESP) and which pursues exhaustive search. Instead of iteratively merging the tours, ESP exploits all possible combinations in order to find $k$ subsets of

Figure 4: An illustration of how IDM-kMDC works. A MDC tours in between black dots (a) Initial configuration of tours. Tours are along the mst edges (b) First iteration: the tours for $\{s_1, s_2, s_3\}$ and $\{s_4, s_5\}$ are merged (c) second iteration: the tours for $\{s_1, s_2, s_3\}$ and $\{s_4, s_5\}$ are merged (d) final iteration: the tours for $\{s_1, s_2\}$ and $\{s_3, s_4\}$ are merged.
representatives such that the sum of tour lengths for each subset is minimized. The complexity of the ESP $O(S(n, k)n \log n)$ where $S(n, k)$ is the Stirling number of the second kind.

$$S(n, k) = \binom{n}{k} = \frac{1}{k!} \sum_{j=0}^{k} (-1)^{j} \binom{k}{j} (k - j)^n$$

C. Performance Results

This section provides the performance results. Each individual simulation experiment involves 50 different topologies and the average result is reported. We observed that with a 95% confidence interval, our results stayed within 6%-12% of the sample mean.

Figure 6 shows the observed total tour lengths (TTL). The results indicate that ESP performs better than IDM-kMDC. It is expected since ESP finds the optimal subsets of representatives for minimizing TTL. However, as the value of $\varphi$ increases the gap between IDM-kMDC and ESP diminishes. The figure also shows that the TTL grows as the number segments increases. The reason is that the number of collection points that MDCs have to visit is equal to number of segment representatives which increases total distance. However, both IDM-kMDC and ESP performs better for larger $\varphi$ values. Therefore we can conclude that TTL is inversely proportional with the $k$ (number of available MDCs).

Figure 7 indicates that IDM-kMDC always outperforms ESP in terms of the Maximum Tour Length (MTL). Since in every iteration IDM-kMDC merges the two tours having the least merging cost in order to reduce the tour count by $\varphi$. The performance gap between IDM-kMDC and ESP is significant especially when $\varphi = 0.20$. Even for small $k$ values, IDM-kMDC produces more practicable solutions where the length of the largest tour stays almost constant as the number of segments increases. It is also shown in Figure 7 that, both algorithms perform better for larger $\varphi$ values. Figure 8 confirms these conclusions. Basically, Figure 8 shows that IDM-kMDC provides more load balancing among the MDCs by reducing the standard deviation of tour length. In other words, the lengths of the tours in IDM-kMDC are closer to each other, which means that the movement cost is fairly distributed among the MDCs.

VI. CONCLUSION

Due to the harsh operation conditions, WSNs may suffer large scale damage and the network may be split into multiple disjoint segments. In this paper we have presented IDM-kMDC, a novel algorithm which finds optimized tours for $k$ mobile data collectors (MDCs) in order to establish intermittent communication links among segments. In IDM-kMDC, a network segment is modeled using a representative node. The algorithm initially finds a minimal spanning tree (mst) for the representatives and computes the tours for each mst edge. The idea of the algorithm is to iteratively merge the two tours that have the least merging cost in order to reduce the tour count by one per iteration until the number of tours is equal to $k$. We have validated the performance of the algorithm through simulation. The results confirm the effectiveness of the algorithm in terms of minimizing the total tour lengths as well as achieving a balanced load distribution among the employed MDCs. Our future work will focus on finding hybrid solutions to the same problem using both mobile and stationary nodes together.

Acknowledgements: This work was supported by the National Science Foundation (NSF) award # CNS 1018171.

REFERENCES