Rendezvous Planning in Mobility-assisted Wireless Sensor Networks

Guoliang Xing; Tian Wang; Zhihui Xie, Weijia Jia
Department of Computer Science
City University of Hong Kong
{glixing, tianwang, zihuixie, itjia}@cityu.edu.hk

Abstract—Recent research shows that significant energy saving can be achieved in wireless sensor networks by using mobile elements (MEs) which are capable of carrying data mechanically. However, the low movement speed of MEs hinders their use in a class of data-intensive applications that need to collect high-bandwidth sensor data under temporal constraints. To address this issue, we propose a rendezvous-based approach in which a subset of nodes in the network serve as the rendezvous points (RPs) that buffer data sent (possibly through multiple hops) from sources. MEs then collect the cached data from RPs within the required deadlines. The use of RPs enables MEs to collect a large volume of data at a time without traveling long distances, which can achieve a desirable balance between network energy saving and data collection delay. We develop two rendezvous planning algorithms, RP-CP and RP-UG. RP-CP finds the optimal RPs when MEs move along the data routing tree while RP-UG greedily chooses the RPs with maximum energy saving to traversal distance ratios. We design the Rendezvous-based Data Collection protocol that can facilitate reliable data transfers from RPs to MEs when there are significant unexpected delays in the movement of MEs and network communication. Our approach is validated through extensive simulations under realistic settings of Mica2 motes.

I. INTRODUCTION

Recent years have seen the deployments of wireless sensor networks (WSNs) in data-intensive applications including emergency response, crisis management [1], structural health monitoring (SHM) [27], etc. WSNs in these applications often produce high-bandwidth sensor data that need to be collected under stringent delay constraints. For instance, SHM sensors must sample at higher than 100 Hz and stream the accumulated data to the base station every a few minutes when the health of a structure needs to be inspected. On the other hand, sensors in such applications must operate on limited power supplies like batteries for extended lifetime up to years. Therefore, a fundamental challenge for these WSNs is to support high-bandwidth data collection with minimum network energy consumption.

Several recent work have exploited the use of mobility-assisted WSNs in data collecting applications [6]. In this approach, a small number of mobile devices referred to as mobile elements (MEs) roam about sensing fields and collect data from sensors via short-range wireless communication. As MEs can replenish their energy supplies because of the mobility, significant network energy saving can be achieved by reducing or completely avoiding costly multi-hop wireless transmissions. Moreover, MEs are capable of transferring a large volume of data at a time, which leads to possible higher average bandwidth than multi-hop wireless transmissions [12].

However, the primary disadvantage of this approach is the increased latency. For instance, the typical speed of several practical ME systems (e.g., NIMs [19] and Packbot [22]) is about 0.1 – 1 m/s. As a result, it takes hours for a ME to tour a large sensing field, which cannot meet the delay requirements of many data-intensive applications. Table I summarizes the characteristics of practical MEs and multi-hop WSNs as two different means of data collection.

![Table I: Comparison of MEs and multi-hop WSNs in data collection.](image)

Table I illustrates a WSN that utilizes such a mechanism. By configuring the number and locations of RPs, the rendezvous-based approach can meet various data-intensive applications’...
delay requirements that range from a few minutes to hours.

This paper makes the following contributions. 1) We formulate the minimum-energy rendezvous planning (MERP) problem which aims to find a set of RPs that can be visited by MEs within a required delay while the network energy consumed in transmitting data from sources to RPs is minimized. 2) We develop two rendezvous planning algorithms, RP-CP and RP-UG. RP-CP finds the optimal RPs when MEs move along the data routing tree. RP-UG is a utility-based greedy heuristic that can find RPs with good ratios of network energy saving to ME traversal costs. 3) We design the Rendezvous-based Data Collection (RDC) protocol that facilitates reliable data transfers at RPs by efficiently coordinating MEs’ movement and data transmission/caching in the network. 4) Our simulations based on realistic settings of Mica2 motes show that our approach significantly outperforms several other schemes in high-bandwidth data collection under temporal constraints. Moreover, RDC maintains robust performance in presence of unpredictable delays in MEs’ movement and network communication.

The rest of the paper is organized as follows. Section II reviews related work and compares different ME-based data collection approaches. Section III introduces our assumptions and formulates the problem studied. Two rendezvous planning algorithms are presented in Section IV and V, respectively. The extensions of the algorithms are discussed in Section VI. Section VII describes the Rendezvous-based Data Collection (RDC) protocol. Section VIII presents the simulation results and Section IX concludes the paper.

II. RELATED WORK

Recent work has exploited controlled mobility to enhance the connectivity of sparse ad hoc networks [29], [14], [9], [23], and reduce the energy consumption of WSNs. We review three different approaches [6] of utilizing controlled mobility in data collecting WSNs.

Motivated by the observation that the nodes in the vicinity of base stations often deplete energy first as they forward more data, several work [16], [8], [25] propose to use mobile base stations to achieve balanced energy usage. It is showed in [16], [8] that the optimal path of MEs is the perimeter of the sensing field. However, the average network energy consumption in this approach is high as nodes must communicate with the mobile base stations through multi-hop routes. Moreover, as base stations often change their paths dynamically, additional overhead is incurred in maintaining efficient network topology.

In the second approach, MEs visit source nodes and gather data from them via one-hop communication. Shah et al. [20] model the performance of MEs based on the random mobility model. Several heuristics are proposed in [10], [22] to schedule the movement of MEs such that the source nodes can be visited before buffer overflow. While this approach minimizes the network energy consumption by avoiding multi-hop wireless transmissions, it incurs high latency when collecting data from large sensing fields due to the slow speed of MEs. Our approach addresses this issue by finding a set of rendezvous points (RPs) that buffer data from sources and transfer them to MEs when they arrive. Accordingly, new techniques different from the existing ME scheduling schemes [10], [22] are required to minimize the energy consumption on the network paths from sources to RPs while ensuring that the RPs can be visited by MEs within temporal constraints.

The third is a hybrid approach that jointly considers multi-hop network transmissions and the movement of MEs in data collection. The rendezvous approach studied in this paper falls into this category. In [11], [2], source nodes send data to the nodes close to the ME paths which are picked up as the MEs pass by. Wang et al. [24] show that constraining MEs in the vicinity of the base station can maximize the network lifetime. These work are not concerned with collecting data with bounded delay requirements. In [3], urgent messages are sent to the source nodes that are visited by MEs more frequently in order to achieve early delivery. As MEs pick up most data (except the urgent messages) from data sources, such a scheme results in high latency in large networks. Moreover, different from our objective of minimizing network energy consumption in collecting high-bandwidth data within deadlines, the urgent messages are assumed to be infrequent in [3] and hence have limited impact on network energy consumption.

Our problem formulation is related to the Traveling Salesman Problem (TSP) [4] and the Vehicle Routing and Scheduling Problem (VRSP) [21]. However, new techniques are needed as the tour of MEs and network routes of data should be jointly considered in order to determine the optimal locations of rendezvous points while only the tour of visiting a fixed set of sites need to be found in TSP and VRSP.

III. PROBLEM FORMULATION

In our problem, a set of source nodes periodically generate data samples that must be delivered to the base station (BS) before the deadline. One or more MEs can move in the sensing field and communicate with nodes. Our goal is to find a set of nodes referred to as rendezvous points (RPs) from which MEs can pick up the data originated from sources and transport to the BS before the deadline. The energy consumed by the network to transmit the data from sources to the RPs should be minimized. We refer to this problem as the Minimum Energy Rendezvous Planning (MERP).

A. A Numerical Example

The effectiveness of the rendezvous-based approach in data collection depends on several factors such as the speed of MEs, the speed that data travels in the network and the tightness of data deadlines. Intuitively, a ME can directly fetch the data from source nodes and carry them back to the BS if the deadline is loose or the speed of ME is high enough. In contrast, if the deadline is very tight, the data should be transmitted through the network only because of the low speed of MEs. We now use a simple example to show that the rendezvous based approach is suitable for data collection under a broad range of temporal constraints.

Suppose there exists only one source in the network which is 600 m away from the BS. We first estimate the time it takes to deliver a 10 Kb data chunk if it is solely relayed through a multi-hop network or is solely transported by a ME, respectively. We set the radio parameters according to
the CC1000 radio on MICA2 motes [7]. To account for routing/MAC overhead and contention delay, we assume the effective bandwidth is 20 Kbps. The effective radio range is 20 m. The time it takes the data chunk to travel from the source to the BS along a straight line can be estimated as \[
\text{time} = \frac{600m \cdot 10Kbps}{20Kbps} = 15s.
\]
On the other hand, as the speed of practical MEs is much lower (about 0.1 \sim 1 m/s [22], [19]), it would take a ME 20 \sim 200 minutes to travel to the source and return to the BS.

Suppose a node on the path between the source and BS serves as the rendezvous point (RP) that receives data from the source and transfers to the ME when it arrives. Then a range of delay requirements, from 15s to 200 minutes, can be met by varying the location of the RP. We note that this range is much broader in large-scale networks. For instance, our simulations show that it takes a ME about 4 hours to visit 200 source nodes randomly distributed in a 500m \times 500m region when it moves at 0.5 m/s.

Another important observation from this example is that data travels at least an order of magnitude faster than MEs, and hence always arrive at RPs first. Accordingly, the data delivery deadline can be mapped to the maximum length of the tour taken by the ME to visit all RPs and return to the BS.

B. Assumptions

We make the following assumptions in this paper.

1) Nodes and MEs are aware of their own physical locations.
2) The maximum speed of MEs is \(v_m\), which is much slower than the speed that data travels in the network.
3) Each source generates a chunk of data synchronously at a point \(d_i\) and the data chunks must be delivered to the BS within \(D\).
4) The energy of MEs is replenishable (e.g., by recharging batteries at the BS).

Nodes and MEs may obtain the location information from the GPSs on them or a location service in the network. The delivery deadline is often determined by the data freshness required by users. For instance, a user may issue a sliding-window query [17] to an earthquake-monitoring WSN: “report the seismic data of region X every 10 minutes and the data is sampled every 10 seconds”. In this example, the deadline is 10 minutes.

C. The Min-energy Rendezvous Planning Problem

We now formally formulate our problem when there is only one ME available and all data chunks have the same deadline \(D\). We discuss how our solutions can be extended to more general cases in Section VI.

Min-energy Rendezvous Planning (MERP): Given a geometric tree \(T(V, E)\) rooted at \(B\) and a set of source nodes \(S = \{s_i\} \subseteq V\), find a tour \(U\) that originates from \(B\) and intersects the path from each source node \(s_i\) to \(B\) at point \(R_i\), such that the total length of \(U\) is no greater than \(L = v_m \cdot D\), and

\[
U = \arg \min_{s_i \in S} \sum_{s_i \in S} d_T(s_i, R_i) \quad (1)
\]

where \(d_T(s_i, R_i)\) is the length of the path from \(s_i\) to \(R_i\) on tree \(T\).

A tour originated from point \(B\) is a continuous path that starts from and ends at \(B\). The length of the ME’s tour is bounded by the maximum distance it can travel within data deadline \(D\). A ME always starts its tour from the BS, which allows it to pick up more data within the same traversal distance as the nodes close to the BS forward more data. Moreover, this strategy reduces the energy consumption of these bottleneck nodes and hence improves the network lifetime [24].

As the path from source \(s_i\) to the BS on tree \(T\) is unique, the RP for \(s_i\) and the ME is the intersection point between the path and the ME tour. If there exist multiple intersection points, the RP is the point closest to \(s_i\). We ignore the delay it takes to transfer the data from RPs to the ME. However, the above formulation can be easily extended to account for this delay if it is significant compared to the deadline.

The above formulation assumes that the storage capacity of a node is large enough to buffer the total volume of data generated by its descendants on the routing tree within time \(D\). Several recent sensor network platforms [18] can integrate 10 \sim 100 Mb NAND flash memory with ultra-low power consumption.

Finding the optimal ME tour requires global network information such as the locations of all nodes and the topology of the routing tree, which is often expensive to obtain in large networks. Therefore, we assume that tree \(T\) in the problem definition is a geometric approximation to the actual routing tree \(R\). Specifically, \(T\) is composed of the source nodes and the junction nodes of \(R\), and the paths between them on \(R\) are replaced by line segments on \(T\). A node is a junction node of \(R\) if it is the intersection of at least two paths from sources to the root.

Fig. 2. The actual routing tree, the geometric tree and the ME tour are represented by gray, solid and dotted line segments, respectively. Node \(a\) is a junction node that lies on the paths from both \(s_1\) and \(s_2\) to BS. The RPs are represented by white nodes. Gray nodes represent the actual nodes on the routing tree from which the ME receives data.

As each edge on \(T\) represents a multi-hop network path on \(R\), we approximate the total power consumption of the nodes on the path by the Euclidean distance of the edge on \(T\). This assumption allows MEs to estimate the network power consumption without knowing the global network topology.

We note that such an energy model is also adopted by several
existing power-efficient data dissemination protocols [13]. We also assume any point on an edge of $T$ can serve as a RP. We discuss how the RPs on $T$ can be replaced by real nodes on $R$ in Section VII-A. In the rest of the paper, both $T$ and $R$ are referred to as the routing tree unless otherwise indicated.

Fig. 2 shows an example of the actual routing tree, the corresponding geometric tree and the ME tour. The use of geometric tree allows us to plan the ME path with the minimum information about the network. Our solutions can yield better performance if the topology of the actual routing tree is available. We have the following theorem regarding the complexity of the MERP problem.

**Theorem 4:** The MERP problem is NP-hard.

**Proof:** We now show that the decision version of the MERP problem is NP-hard by a reduction from the Geometric Traveling Salesman Problem (GTSP). A special-case decision version of the MERP problem is to ask if there exists a set of RPs such that the network energy consumption is zero. In order to incur zero network energy consumption, all the sources must be RPs as well. In other words, the ME must visit all the RPs on a tour no longer than $L$. This is exactly the decision version of the GTSP problem in which a salesman needs to visit a set of sites on a tour no longer than $L$.

---

### IV. OPTIMAL RENDEZVOUS PLANNING WITH CONSTRAINED ME PATH

In this section, we solve a special case of the MERP problem where the ME is only allowed to move along the routing tree. Although moving MEs along fixed paths reduces the opportunities of saving network energy, it significantly simplifies the motion control of MEs and is adopted by several mobile sensor systems in practice. For example, the meteorological sensors of the NIMS system deployed at the James San Jacinto Mountain Reserve can only move along fixed cables [5], [19]. Moreover, planning the ME path along the routing tree improves the system reliability as MEs are able to transfer the carried data back to the network whenever they experience mechanic problems. Finally, by developing the optimal solution to this simpler case, we gain important insights into tackling the general MERP problem discussed in Section V.

We first introduce the following definitions. Let $T(V,E)$ be a geometric tree with root $B$.

- For any vertex $u$ on $T$, vertex $v$ is a descendent of $u$ if $u$ lies on the unique path from $B$ to $u$ on $T$. $v$ is a child of $u$ if $v$ is adjacent to $u$ and is a descendent of $u$.
- A subtree of $T$, $T'(V',E')$, is a connected graph of $T$. An edge in $E'$ may be part of an edge of $T$. For instance, in Fig. 3, the bold edges constitute a subtree, and edge $(R_1, R_3)$ is part of edge $(R_1, s_2)$. For the convenience of discussion, we also refer to such a partial edge as an edge of $T$ if no confusion is caused.
- The preorder traversal (also referred to as the depth-first traversal) of $T$ is the recursive process of visiting all the nodes on $T$, starting from the root, and then traversing in the preorder each of the subtrees of the root. For the bold subtree in Fig. 3, the sequence of nodes visited in the preorder traversal is $< B, A, R_1, R_3, R_2 >$.

- The preorder tour of $T$ is a walk along $T$ that starts from the root and visits each node of $T$ in the preorder and returns to the root. For the bold subtree in Fig. 3, the preorder tour visits nodes in the order of $< B, A, R_1, R_3, R_1, A, R_2, A, B >$.

An interesting property of the preorder tour of tree $T$ is that each edge is visited exactly twice. Therefore, the total length of the tour is twice of the total length of edges on $T$.

We now describe an optimal algorithm called the Rendezvous Planning with Constrained Path (RP-CP). RP-CP is based on the key observation that, when the movement of ME is on tree $T$, the total length of the edges covered by any ME tour is at most $L/2$. This is because any tree does not contain cycles by definition and hence the ME must pass any edge on its path at least twice. RP-CP finds the most “important” edges whose total length is $L/2$. The importance of an edge is measured by the number of source-to-root paths it lies on. An interesting property is that the edges chosen in this fashion form a connected subtree of tree $T$ and hence the ME can traverse them by a preorder tour of length $L$.

The pseudo code of RP-CP is shown in Fig. 4. RP-CP first creates a list $W_T$ that includes all the edges sorted by the non-ascending order of $c(\cdot)$ value (step 2). For edge $(u, v)$, function $c(v)$ is the number of sources in the subtree rooted at $v$. RP-CP then finds a sublist of edges $W \subseteq W_T$ starting from the first edge such that the total length is no greater than $L/2$ (step 3). Part of the next unchosen edge is included in $W$ if necessary in order to ensure the total length of the (partial) edges in $W$ is exactly $L/2$ (step 4). As shown by Lemma 1, all the (partial) edges chosen form a connected subtree of $T$. Therefore, the ME can traverse this subtree by a preorder tour with length of $L$. The RPs include all the intersection points between the subtree and $T$. The time complexity of RP-CP is $O(|E| \log |E|)$.

---

**Fig. 3.** An example of RP-CP’s execution. Each node is labeled by its name and the $c(\cdot)$ value. The sources are represented by solid nodes. The length of all edges is one unit and $L$ is 7 units. The edges chosen by RP-CP are highlighted.
Lemma 1: The edges in $W$ form a connected subtree of $T$ that is rooted at $B$.

Proof: To prove all the edges (including the partial edge added at step 4 if it exists) in $W$ are connected, it suffices to show that, for any edge $(x, y) \in W$, the set of edges traversed when walking from the root $B$ to $y$ are also included in $W$. Suppose $(x', y')$ is an edge in this set. As $y'$ is visited before $y$ when walking from $B$ to $y$, $y'$ is a descendant of $y$. Therefore, $c(y') \leq c(y)$. As the edges in $W$ are added in the non-ascending order of $c(\cdot)$, $(x', y') \in W$.

We have the following theorem regarding the optimality of RP-CP.

Theorem 2: The RP-CP algorithm is optimal for the MERP problem when the tour of ME is constrained on the routing tree.

Proof: Suppose $T^*$ is the subtree of routing tree $T$ that is covered by an arbitrary ME tour. We define function $S(T^*)$ as follows:

$$S(T^*) = \sum_{(u,v) \in E(T^*)} c(v) \cdot |uv| \quad (2)$$

where $c(v)$ is the number of sources in the descendants of $v$. The total distance that all data chunks travel on $T$ before being picked up by the ME tour can be expressed as $S(T) - S(T^*)$. According to Lemma 1, the edges in $W = \{e_1, \ldots, e_m\}$ computed by RP-CP constitute a subtree of $T$. Let $T^*$ represent this tree. Therefore, to prove the optimality of RP-CP, it suffices to show that $S(T^*) \geq S(T^*)$.

Let list $W^* = \{e_1', \ldots, e_n'\}$ contain all edges in $E(T^*)$ sorted in the non-ascending order of $c(\cdot)$. Suppose the edges in $W^*$ and $W$ are concatenated into two parallel line segments $Y^*$ and $Y$, respectively. Align $Y^*$ and $Y$ according to the location of the first node as shown in Fig. 5. For each node $u$ on $Y$, add a virtual node $u'$ at the corresponding location on $Y^*$ if a node does not exist at that location. Define the $c(\cdot)$ value of a virtual node to be the $c(\cdot)$ value of the real node next to it from the left.

For any edge $e_k = (u, v) \in W$, suppose $u'$ and $v'$ are two (possibly virtual) nodes that are at the corresponding locations on $Y^*$. Without loss of generality, suppose there exist multiple nodes between $u'$ and $v'$ on $Y^*$. Let $W^*_k$ represent the set of the edges between these nodes. According to the RP-CP algorithm (Fig. 4), the edges are added to $W$ in the non-ascending order of $c(\cdot)$. Hence, $\forall (x, y) \in W^*_k, c(v) \geq c(y)$. We have:

$$c(v) \cdot |uv| \geq \sum_{(x,y) \in W^*_k} c(y) \cdot |xy| \quad (3)$$

Summing the above inequality over all the edges in $W$, we have:

$$S(T^*) = \sum_{(u,v) \in W} c(v) \cdot |uv| \geq \sum_{1 \leq i \leq m (x,y) \in W^*_k} c(y) \cdot |xy| \quad (4)$$

As the tour of ME is constrained on tree $T$, each edge of $T^*$ must be passed by the ME at least twice. That is, the total length of the edges on $T^*$ is at most $L/2$. Moreover, the length of $Y$ is equal to $L/2$ and hence the length of $Y^*$ is also $L/2$ due to the addition of virtual nodes. Therefore, we have

$$\sum_{1 \leq i \leq m (x,y) \in W^*_k} c(y) \cdot |xy| \geq \sum_{(u,v) \in W^*} c(v) \cdot |uv| = S(T^*) \quad (5)$$

From (4) and (5), we have $S(T^*) \geq S(T^*)$.

V. RENDEZVOUS PLANNING WITH UNCONSTRAINED ME PATH

We now turn our attention to the rendezvous planning when the ME’s movement is not constrained to be on the routing tree. We present a greedy algorithms referred to as RP-UG (utility-based greedy heuristic) in this section.

RP-UG operates in iterations. In each iteration, the current ME tour is expanded by including a new RP with the greatest utility until the maximum length is reached. The utility of a RP is defined as the ratio of the network energy saved by including it on the ME tour to the length increase of the tour. The intuition behind the iterative structure of RP-UG is that the utility of a RP is affected by the length of the ME tour, which can be illustrated by Fig. 7. Suppose each edge has a length of one unit. If the ME tour is only long enough to cover $B$ and another node, either $n_2$, $n_3$ or $n_4$, $n_2$ should be chosen because it saves the most distance (2 units for each source) that data travels along the tree. However, if the ME tour can cover $B$, $n_3$ and $n_4$, the utility of $n_2$ becomes zero because all data are picked up by the ME at either $n_3$ or $n_4$ before they reach $n_2$. The iterative structure allows RP-UG to dynamically update the utility of nodes when the ME tour is expanded and hence better RP choices can be made.
Fig. 6 shows the pseudo code of RP-UG. Initially, the RP list only includes the base station B. RP-UG then adds a number of virtual nodes to the tree such that each edge longer than a constant $L_0$ is split into multiple line segments of length $L_0$. $L_0$ is a parameter set by the application according to the desirable trade-off between solution quality and computational complexity. A smaller $L_0$ provides RP-UG more choices of candidate RPs while resulting in more iterations.

RP-UG uses procedure $TSP(I)$ to compute the minimum length of a tour that visits all points in set $I$. $TSP(I)$ can be implemented by a Geometric Traveling Salesman Problem solver. At the beginning of each iteration, RP-UG first finds all RP candidates (step 2). A node is a RP candidate if it can be visited together with all existing RPs by a tour no longer than $L$. RP-UG then computes the utility of candidate $x$ by:

$$u(x) = \frac{\sum_{s_i \in S} d_T(s_i, Q) - \sum_{s_i \in S} d_T(s_i, Q \cup \{x\})}{TSP(Q \cup \{x\}) - TSP(Q)}$$

where $d_T(s_i, I)$ is the minimum distance along tree $T$ from $s_i$ to any node in $I$. $u(x)$ is equal to the ratio of the reduction of total distance that data chunks travel along the tree to the tour length increase. RP-UG then adds the RP candidate with the greatest utility to the RP list (step 4). Then all the RPs whose utilities become zero are removed from the RP list (step 5), which is necessary since the addition of a new RP may invalidate some existing ones as discussed earlier. Finally, if all source nodes are included in the RP list, RP-UG terminates because the ME can visit all of them within the deadline and the total network energy is zero. Otherwise, a new iteration is started to find more RPs.

The complexity of RP-UG is $O(|V|^2 \cdot C(TSP))$ where $C(TSP)$ is the complexity of the TSP procedure. The TSP algorithm in [4] has an approximation ratio of $(1 + 1/c)$ and a complexity of $O(|V|(|\log |V||)^{O(c)})$ for any fixed $c > 1$. Notice that RP-UG invokes the TSP procedure after adding a new RP to the current RP list whose TSP tour has been computed in the previous iteration (step 2). Accordingly, an optimization can reduce the complexity by a factor of $|V|$ by utilizing the partial TSP tours computed previously. Therefore, the overall complexity of RP-UG is $O(|V|^2 \cdot (|\log |V||)^{O(c)})$. We note that RP-UG is only run by the BS or MEs, which have more computational power than network nodes.

**Input:** routing tree $T(V, E)$, source set $S = \{s_i\}$, $L, L_0$

**Output:** RP list $Q$

1. $Q = \{B\}$. Add virtual nodes on the edges of $T$ such that each new edge is no greater than $L_0$.
2. $W = \{v|v \in V \setminus Q, TSP(Q \cup \{v\}) \leq L\}$. If $W = \emptyset$, exit.
3. Find $x \in W$ with maximum $u(x)$ defined by (6). If multiple nodes have the same $u(\cdot)$ value, choose the one with maximum $TSP(Q \cup \{x\}) - TSP(Q)$.
4. $Q = Q \cup \{x\}$.
5. $\forall z \in Q$, if $u(z) = 0, Q = Q \setminus \{z\}$.
6. If $S \subseteq Q$, exit. else goto 2.

Fig. 6. RP-UG – a utility-based greedy rendezvous planning algorithm.

We now discuss an example of RP-UG’s execution. Fig. 7 shows the RP lists and the ME tours at the end of three iterations. $n_2$ is included on the tour in the first iteration. In iteration 2, $n_3$ has the greatest utility among all nodes. Although $n_4$ and $n_3$ cause the same length increase of the tour, $n_4$ has a smaller utility as it only saves 1 unit of distance on edge $(n_4, n_2)$ that is traversed by the data from $s_3$ while $n_3$ saves 2 on edge $(n_3, n_2)$ (1 for the data from each of $s_1$ and $s_2$). $n_4$ is added on the tour in iteration 3. $n_3$ is removed from the tour (at step 5 of RP-UG) because including $n_3$ on the tour renders it useless to be zero. As a result, the tour length is shortened and hence more RPs can be included in the following iterations.

As illustrated by Fig. 7, a shortcoming of RP-UG is that it may include on the ME tour the RPs whose utility become zero later, resulting waste of computational time. However, the chance such a case occurs decreases when the ME tour becomes longer. This is because, as the subtree covered by the ME tour grows wider, the candidate RPs with similar utilities appear on more branches, which renders the possibility that the descendants of a particular existing RP are included on the tour (hence makes it useless) lower. Based on this observation, we optimize RP-UG by starting with the RP list found by RP-CP. As discussed in Section IV, the ME tour in RP-CP is along the routing tee. Therefore, the ME tour that covers the same RPs is shorter when this constraint is relaxed. For example, to cover the RPs found by RP-CP, $R_1$, $R_2$ and $R_3$ in Fig. 3, the length of ME tour is much shorter than 7 units which is needed if the ME moves along the tree. Therefore, more RPs will likely be added to the initial RP list, resulting better performance.

**VI. EXTENSIONS**

In this section, we discuss several extensions to our rendezvous planning algorithms. Specifically, our algorithms can achieve more energy saving by utilizing multiple MEs if they are available. Moreover, they can also be extended to collect data with different deadlines.

A. **Rendezvous Planning of Multiple MEs**

The basic idea of utilizing multiple MEs is to assign a subset of source nodes to each ME. As a result, each ME can plan its tour independently, which avoids the overhead of inter-ME coordination. Specifically, suppose there exist $N$ MEs available, we first decompose the routing tree into $N$ edge-disjoint subtrees. The decomposition should ensure that the workloads of MEs are balanced. A simple tree decomposition heuristic works as follows. For a given $N$ and tree $T$, we fist
compute a value \( w(u) \) for each node \( u \), which is equal to the sum of the distances from the sources in the descendants of \( u \) to \( u \). \( w(u) \) quantifies the workload of the ME that collects data from the subtree rooted at \( u \). We traverse tree \( T \) in the preorder, and trim the subtree rooted at \( u \) if \( w(u) < W/N \) where \( W \) is the total distances from all sources to the root. The same process is repeated on the rest of the tree until all edges of \( T \) are traversed or the number of subtrees obtained is equal to \( N \). The key of this heuristic is that the rest of tree remains connected every time a subtree is removed.

Note that the root of a subtree after the decomposition may not be the base station. In such a case, the ME transfers the carried data to the root node, which then sends to the BS along the routing tree. Fig. 8 illustrates the case of three MEs. The routing tree is decomposed into three subtrees, each of them contains two source nodes. As the BS is not on subtree 2, ME2 sends its data to the root node \( A \) which then forwards to the BS along subtree 1. Note that \( A \) may also be visited by ME1 as it is contained by subtree 1. However, ME1 cannot pick up from \( A \) the data transferred from ME2, because these data have experienced a delay on ME2 and hence must be transmitted through the network in order to meet the deadline.

![Subtree Diagram](image)

**B. Rendezvous Planning under Multiple Deadlines**

We now discuss how to extend RP-UG to collect data with different deadlines. In such a case, the movement of ME may become aperiodic because different RPs need to be visited at different frequencies. If the set of RPs is given, finding a feasible ME tour is analogous to the task scheduling problem [15] in which tasks arrive at a processor and must be finished before their deadlines. Specifically, the traversal time between two nodes can be viewed as the context switching time between two tasks. All tasks have zero execution time as we neglect the delay a ME stays at a node. Several scheduling algorithms such as Rate Monotonic Scheduling (RMS) and Earliest Deadline First (EDF) [15] can be applied to solve this problem.

Fig. 9 shows a scheduler implemented based on the EDF algorithm. For a given set of RPs \( Q \), it first constructs a deadline set \( D \). Each element of \( D \) contains a tuple composed of a RP and a deadline. The deadlines associated with a RP include all the deadlines of its source descendants. In addition, for each source-deadline pair \((s_i, d_i)\), a new pair \((B, d_i)\) is also added to \( D \), which represents the requirement that the ME must visit \( s_i \) and return back to the BS within the deadline. In each iteration, the node with the earliest deadline is visited and its ID and the visiting time is recorded. The algorithm terminates if a scheduling period is finished or there exists a node that cannot be visited within the deadline. The scheduling period can be computed according to the set of deadlines of all sources.

**Input:** \( Q \), Distance matrix \( dist[|Q|][|Q|] \), Source set \( S \), Deadline set \( D \)

**Output:** \( \Omega \) - RP sequence visited by ME

1. Deadline list \( \Phi = \emptyset \). For each \( R_i \in Q, D = D \cup \{(R_i, d_i)\} \cup \{(B, d_i)\} \), if \( s_i \) is a descendant of \( R_i \).
2. Current time \( t = 0 \), current node \( n = B \).
3. If \( t > T \), return Success; Else \( \forall (r, d) \in D, r = s_i \text{ or } B, d = d_i + t \).
4. Choose \((r, d) \in D, s.t., d = \max d_i \). If two elements in \( D \) have the same maximum deadline, choose the one that does not include the base station \( B \). Otherwise, break the tie arbitrarily.
5. If \( d < t + dist[n][r]/v \), return Failure. Else, \( \Omega = \emptyset \cup \{(r, t)\} \), goto 2.

**Fig. 9.** An earliest deadline first (EDF) based scheduler that finds the visiting sequence of a given set of RPs.

The extended RP-UG operates in iterations. Given the current set of RPs and their deadlines, RP-UG uses the scheduler as a procedure to identify all RP candidates and adds the one with the maximum utility to the RP set. A node is a RP candidate if a feasible ME tour still exists after adding it to the current RP set. The utility of a RP candidate is defined as the ratio of the energy saving to the increase of the scheduling period computed. Such a process repeats until no nodes can be added to the current RP set. We note that this algorithm has a similar structure to original RP-UG shown in Fig. 6. The key difference is that the ME tour is computed using the scheduler instead of a TSP solver.

**VII. RENDEZVOUS-BASED DATA COLLECTION PROTOCOL**

We develop the Rendezvous-based Data Collection (RDC) protocol that utilizes MEs to collect high-bandwidth sensor data under temporal constraints. A key challenge in the design of RDC is that both MEs’ movement and network communication may experience unexpected delays that can cause MEs and data to miss each other at rendezvous points or even result in data deadline misses. To address this issue, RDC employs two mechanisms called safe waiting and online path adaptation, which ensure data to meet their deadlines while maximizing the opportunity of data transfer from the network to MEs. We now describe the key components of RDC.

**A. Initialization**

Initially, the BS constructs an approximate routing tree using the locations of source and junction nodes on the routing tree as described in Section III-C. The BS can have source nodes send their locations along with data packets. Junction nodes piggyback their locations in the data packets when forwarding them. The locations of source nodes may also be contained in the user queries received by the BS. In the case of multiple MEs the BS decomposes the tree as described in Section VI and then assigns each ME a separate subtree. The following discussion focuses on the operation of RDC on one ME.
After initialization, the ME computes RP locations by running RP-CP or RP-UG (depending on the application requirement). Note that the found RPs are physical locations at which there may exist real nodes. RDC addresses this issue as follows. For each computed RP location, the ME sends a message to the end node (which is either a junction or source node) of the edge that hosts the RP. As the message is being transmitted along the actual routing tree, it finds the real node closest to the RP location on the path. For example, to replace RP \( R_1 \) with a real node in Fig. 2, a message is sent to \( s_1 \), which will reach node \( b \) on the actual routing tree. After the ME identifies all RP nodes, it replaces a computed RP location with the location of corresponding RP node if the distance between them exceeds the effective communication range. The ME skips a RP if the new location causes the the ME tour to be longer than \( L \).

\section*{B. Handling Unexpected Delays}

The ME’s movement may experience delays due to mechanic problems or the obstacles on its motion path. Moreover, the network may also suffer from communication delays due to congestion or node/link failures. Consequently, these unexpected delays may cause the ME and data to miss each other at RPs or even result in deadline misses. To cope with this issue, RDC employs two mechanisms called safe waiting and online path adaptation described in the following.

The basic idea of safe waiting is to have the party that has arrived at a RP first wait for the maximum amount of time as long as the deadline can still be met. Specifically, each RP sets a timeout as the ratio of its distance to the BS to the maximum ME speed. If the ME arrives before the timeout, the RP sends the buffered data and cancels the timer. Otherwise, it sends all buffered data to the BS along the routing tree after the timeout. The intuition is that the ME will not be able to return to the BS before the deadline if it arrives at the RP after the timeout.

We now discuss the safe waiting mechanism used by the ME to handle delayed data. Suppose total \( m \) RPs are on the ME’s tour, denoted as \( R_i \) (\( 1 \leq i \leq m \)) in the order of being visited by the ME. Both \( R_1 \) and \( R_m \) represent the BS. Suppose the ME just arrives at \( R_i \). It first solicits data from \( R_i \). If no data has arrived or the volume of data arrived is below a threshold, the ME sets a timeout of \( T_i \) seconds and notifies \( R_i \). If a data chunk arrives before the timeout, \( R_i \) then sends it to the ME. If all the expected data chunks are received or the timeout fires, the ME skips a RP if the new location causes the the ME tour to be longer than \( L \).

\begin{equation}
T_i = \frac{D_i - \sum_{j=1}^{m-i} d(R_i,R_{i+1})}{m - i}
\end{equation}

where \( D_i \) is the time left before the deadline. \( d(R_i,R_{i+1}) \) represents the Euclidean distance between \( R_i \) and \( R_{i+1} \) if RP-UG is used, or the on-tree distance if RP-CP is used. \( \overline{D_{\text{ME}}} \) is the average speed of ME, which is estimated by the ME based on its movement history. In (7), the total idle time before the deadline (after extracting the time spent on moving) is evenly divided among the remaining RPs, which allows the ME to wait for the delayed data in the rest of its tour. Such a strategy is effective for dealing with the network congestion that causes the late arrivals of data at consecutive RPs.

While safe waiting can cope with short delays in ME’s movement and network communication, special care must be taken when the ME cannot finish the rest of its tour before the deadline due to significant delays. As a result, \( T_i \) computed by (7) is negative. To deal with this issue, the ME periodically checks the idle time before the deadline and adjusts its path dynamically. Suppose the ME is on the way to RP node \( R_i \) and the current slack time is \( S \). The ME computes a maximum subset of remaining RP nodes, \( RN_i \), which can be visited before the deadline as follows:

\begin{equation}
RN_i = \{ R_j \mid d(C,R_j) + \sum_{i \leq j \leq m-1} d(R_j,R_{j+1}) \leq \frac{r_m \cdot D_i}{m} \}
\end{equation}

where \( C \) represents the current location of the ME. If a RP is skipped by the ME, the data buffered by the RP node will be sent to the BS along the routing tree after the safe waiting timer fires, as discussed earlier. An empty \( RN_i \) computed by (8) indicates that the ME even cannot return to the BS from the current location before the deadline. In such a case, it transfers the carried data to a nearby node, which will then send them to the BS along the routing tree. As the data travels much faster in the network, they still likely meet the deadline.

\section*{VIII. PERFORMANCE EVALUATION}

This section presents the evaluation of RP-CP and RP-UG, which are implemented in the RDC protocol. The simulations are conducted in a network simulator written in C++. To simulate the the highly unreliable links [28] of WSNs, we implemented a link layer model from USC [30]. Experimental data shows that the USC model can capture the highly probabilistic link characterization of Mica2 motes [30]. To improve the link reliability, we implemented an ARQ (Automatic Repeat Request) scheme that retransmits a packet if an acknowledgment is not received after a timeout of 20ms. The maximum number of retransmissions before dropping a packet is 5. The radio parameters are set according to the data sheet of the CC1000 radio on Mica2 motes [7]. Radio bandwidth is 40 Kbps and transmission power is 4 dbm with the current consumption of 11.6 mA. The size of each packet is 30 bytes.

\begin{algorithm}
\caption{Algorithm for safe waiting}
\begin{algorithmic}
\State \textbf{Input:} ME location \( R \), \text{Time left before deadline} \( T \), ME speed \( \overline{D_{\text{ME}}} \).
\State \textbf{Output:} New location \( R' \).
\State ME sends a message to the BS.
\State if \( T > 0 \) then
\State \( T' = T - \frac{\overline{D_{\text{ME}}}}{m - i} \cdot d(R,R') \)
\State \textbf{else}
\State \( T' = 0 \)
\State \textbf{end if}
\State ME moves to \( R' \) and waits for \( T' \) seconds.
\State return \( R' \).
\end{algorithmic}
\end{algorithm}

\section*{A. Simulation Methodology}

In all simulations, nodes are randomly distributed in a 300m x 300m region. The BS is located at the top left corner of the region. The number of nodes is 400 unless otherwise indicated. 100 source nodes are randomly chosen. The results are the average of 5 different topologies. During the initialization, a shortest-path routing tree is created to connect all the nodes to the BS. The cost metric of a link is the expected number of transmissions [26]. A source generates and stores a data sample of 4 bytes every second. It sends all the accumulated data (40 Kb) to the BS every 20 minutes (i.e., the deadline is 20 minutes). Each simulation lasts for 100 periods. A ME requests data from a RP if they are within 10 m range. Empirical studies [28] showed that such a short range is needed in order to achieve reliable communication between motes.

Besides RP-UG and RP-CP, we implemented four different algorithms, RP-UGO, RP-SRC, PRE and SECTOR, in RDC.
for performance comparison. RP-UGO is the optimized version of RP-UG that starts with the RPs computed by RP-CP, as described in Section V. RP-SRC is similar to RP-UGO except that only source nodes are eligible to be RPs. Similar to RP-CP, PRE also constrains the ME path to be on the routing tree. The difference lies in the subtree they traverse within the deadline. The subtree of PRE contains a subset of the routing tree edges in preorder. In SECTOR, MEs move on a path in the shape of two concentric sectors (with the BS as the center). The central angle of the sectors is 45 degree. The sources send their data through the routing tree to the nodes close to the ME path.

We note that the aforementioned baseline algorithms represent several existing ME based data collection schemes. RP-SRC is similar to the non-rendezvous based algorithms proposed in [3], [22] in which data are picked up by MEs from sources within buffer overflow deadlines. As our approach targets at collecting data from large sensing fields under stringent deadlines (a few minutes), MEs cannot visit all source nodes within the deadline in most of our settings. For a fair comparison with non-rendezvous based schemes, RP-SRC allows a subset of source nodes that can be visited by MEs within the deadline to serve as RPs, while other sources send their data to the closest RP. A similar scheme is also used in [3] to deliver infrequent urgent messages. SECTOR is similar to several existing algorithms with regular-shape ME paths such as the one proposed in [2]. RP-CP and RP-UG are compared against these baseline algorithms under a range of simulation settings with different network densities, ME speeds, number of MEs/deadlines, and variance of ME speeds.

B. Network Energy Consumption

We first evaluate the network energy consumption when the speed of ME varies from 0.1 to 2 m/s. Only one ME is used in this simulation. As the baseline, we also plot the total network energy consumption without using the ME, denoted by NET. Fig. 10 shows that all algorithms yield lower energy consumption when the ME moves faster. This is because the ME is able to collect data on a longer tour within the deadline. RP-UGO and RP-UG yield the best performance among all algorithms. In particular, RP-UGO outperforms RP-SRC by 24% ~ 73%, which validates the effectiveness of the rendezvous-based approach. Moreover, when the ME moves along the routing tree, RP-CP saves about 6 ~ 46% more network energy compared to PRE. SECTOR and PRE yield the worst performance because they do not consider the locations of sources when planning the ME path. The results of this simulation show that our algorithms can effectively take advantage of speed increase of the ME, which is particularly important when they are implemented on different ME platforms. In the following simulations, the ME speed is set to be 1 m/s.

We now evaluate the algorithms under different network densities. Fig. 11 shows that all algorithms perform better with a higher density since the quality of links among nodes becomes better when each node has more neighbors. However, the performance improvement of SECTOR is not as significant as other algorithms because it uses a fixed ME path independent of node density. In contrast, other algorithms find the ME path on a geometric tree that better resembles the actual routing tree when the network is denser.

Fig. 12 shows the performance of different algorithms when the number of MEs increases from one to six. Consistent with the results in Fig. 10 and 11, RP-UGO and RP-UG yield the best performance among all algorithms. Interestingly, their network energy consumption decreases linearly when the number of MEs is fewer than four and then decreases slower thereafter. This is because more than half of the source nodes are within one hop of at least one ME’s path when there are more than three MEs. In particular, most data are picked by RP-UGO directly from their sources in the case of six MEs, which leads to the similar behavior as the non-rendezvous based data collection approaches. These results show that the performance of our algorithms scales with the number of MEs.

Fig. 13 shows the energy consumption when data have different deadlines. Total 8 deadlines, from 5 to 40 minutes at an increment of 5 minutes, are used. In the first run, the deadline is set to 5 minutes. Then a new deadline (in the increasing order) is added in each following run. The number of sources with each deadline is the same. As the deadline is the end of period in our setting, less data are generated when the number of deadlines becomes larger, resulting in lower energy consumption under all algorithms as shown in Fig. 13. However, we can see that the curves of RP based algorithms drop much faster than SECTOR because these algorithms can find better RPs when the deadlines become looser.

C. Impact of Speed Variance

The ME’s speed may suffer from significant variance due to various reasons such as mechanic problems, complex terrains, or the obstacles on its motion path. In this section, we evaluate the robustness of the RDC protocol under different algorithms when the ME has a highly variable speed. All algorithms plan the ME tour based on a speed of 1 m/s. However, the actual ME speed is subject to dynamic changes. The new speed after each change is set to be the difference of 1 m/s and a value randomly drawn from $[-\alpha, \alpha]$. A speed change occurs at each RP location. $\alpha$ varies from 0.1 to 0.5 m/s.

Fig. 14 shows that all algorithms perform worse when the speed of ME has a higher variance. This is because the ME must skip some RPs (by the path adaptation mechanism of RDC described in Section VII) if it experiences a speed significantly lower than 1 m/s. However, we can see that the increase of network energy consumption under RP-UG is small relative to the significant variance of ME speed. For instance, only about 20% data are missed by the ME even when its speed has a variance of 50%.

We now evaluate the effectiveness of the safe waiting and online path adaptation mechanisms employed by RDC. RP-UG is used in this simulation. We compare RDC against the implementation (denoted as RDC-Static) without these two mechanisms. We plot the percentage of data chunks that miss their deadlines versus the speed variance of ME in Fig. 15. We note that this result includes the data dropped due to excess of the maximum number retransmissions. As a baseline, we also plot the percentage of packet losses denoted by NET. Fig.
hence fewer data packets are dropped on lossy links. The distance data travels in the network is shorter in RDC and results in fewer deadline misses than NET. This is because deadline misses do not grow significantly. Moreover, RDC shortens its tour when its speed becomes lower and hence the number of resulting late arrivals at the BS. In contrast, RDC shortens its path dynamically when it experiences delays.

15 shows that the number of deadline misses in RDC-Static grows quickly with the variance of ME speed since the ME does not adjust its path dynamically when it experiences delays resulting in late arrivals at the BS. In contrast, RDC shortens its tour when its speed becomes lower and hence the number of deadline misses does not grow significantly. Moreover, RDC results in fewer deadline misses than NET. This is because the distance data travels in the network is shorter in RDC and hence fewer data packets are dropped on lossy links.

**IX. Conclusion**

In this paper, we present the rendezvous-based approach for utilizing mobile elements to collect data under temporal constraints. We introduce the minimum-energy rendezvous planning (MERP) problem with the objective of finding rendezvous points (RPs) that can be visited by MEs within a deadline while minimizing the energy consumption on the network paths from sources to RPs. Two rendezvous planning algorithms, RP-CP and RP-UG, are developed. RP-CP is an optimal algorithm when MEs move along the data routing tree while RP-UG is a greedy heuristic that chooses RPs with maximum ratios of network energy saving to ME traversal costs. We design the Rendezvous-based Data Collection (RDC) protocol that facilitates data transfers at RPs through efficient coordination between MEs' movement and data transmission/caching in the network. Our simulations show that our approach can significantly reduce network energy consumption and scale well with network density, ME speed, and the number of MEs.

In the future, we will extend our approach to heterogeneous WSNs by taking advantage of the nodes with more computational power and storage capacity. In addition, we plan to integrate other power management schemes such as topology control in the rendezvous-based data collection.

15 shows that the number of deadline misses in RDC-Static grows quickly with the variance of ME speed since the ME does not adjust its path dynamically when it experiences delays resulting in late arrivals at the BS. In contrast, RDC shortens its tour when its speed becomes lower and hence the number of deadline misses does not grow significantly. Moreover, RDC results in fewer deadline misses than NET. This is because the distance data travels in the network is shorter in RDC and hence fewer data packets are dropped on lossy links.

**IX. Conclusion**

In this paper, we present the rendezvous-based approach for utilizing mobile elements to collect data under temporal constraints. We introduce the minimum-energy rendezvous planning (MERP) problem with the objective of finding rendezvous points (RPs) that can be visited by MEs within a deadline while minimizing the energy consumption on the network paths from sources to RPs. Two rendezvous planning algorithms, RP-CP and RP-UG, are developed. RP-CP is an optimal algorithm when MEs move along the data routing tree while RP-UG is a greedy heuristic that chooses RPs with maximum ratios of network energy saving to ME traversal costs. We design the Rendezvous-based Data Collection (RDC) protocol that facilitates data transfers at RPs through efficient coordination between MEs’ movement and data transmission/caching in the network. Our simulations show that our approach can significantly reduce network energy consumption and scale well with network density, ME speed, and the number of MEs.

In the future, we will extend our approach to heterogeneous WSNs by taking advantage of the nodes with more computational power and storage capacity. In addition, we plan to integrate other power management schemes such as topology control in the rendezvous-based data collection.

**References**