Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 50 (2006) 3449–3465
Computer Networks
www.elsevier.com/locate/comnet

Data harvesting with mobile elements in wireless sensor networks ☆

Yaoyao Gu, Doruk Bozdağ, Robert W. Brewer, Eylem Ekici *

Department of Electrical and Computer Engineering, Ohio State University, 2015 Neil Avenue, 205 Dreese Lab., Columbus, OH 43210, United States

Received 6 September 2005; received in revised form 23 January 2006; accepted 24 January 2006
Available online 7 March 2006

Responsible Editor: I.F. Akyildiz

Abstract

In recent studies, using mobile elements (MEs) as mechanical carriers to relay data has been shown to be an effective way of prolonging sensor network life time and relaying information in partitioned networks. As the data generation rates of sensors may vary, some sensors need to be visited more frequently than others. In this paper, a partitioning-based algorithm is presented that schedules the movements of MEs in a sensor network such that there is no data loss due to buffer overflow. Simulation results show that the proposed Partitioning-Based Scheduling (PBS) algorithm performs well in terms of reducing the minimum required ME speed to prevent data loss, providing high predictability in inter-visit durations, and minimizing the data loss rate for the cases when the ME is constrained to move slower than the minimum required ME speed.

Keywords: Sensor networks; Sensor–actor; Mobility; Scheduling; Path planning; Data collection

1. Introduction

The use of wireless sensor networks (WSNs) have been proposed for critical applications such as battlefield surveillance, habitat monitoring [1–4], traffic monitoring [5], and nuclear, chemical and biological attack detection [6]. The collected data at the sensors are usually transmitted to the sinks via power efficient multi-hop routing protocols [7–10]. The traditional approach for data relaying in wireless sensor networks involves multi-hop communication from data sources to sinks. A major problem with multi-hop routing is observed when networks become partitioned. Furthermore, relaying data over a large number of hops also reduces the life time of sensor nodes. As the number of sinks in wireless sensor networks is relatively small, the

☆ A preliminary version of this paper has appeared in the Proceedings of Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks.

* Corresponding author. Tel.: +1 614 292 0495; fax: +1 614 292 7596.

E-mail addresses: guy@ece.osu.edu (Y. Gu), bozdagd@ece.osu.edu (D. Bozdağ), brewerr@ece.osu.edu (R.W. Brewer), ekici@ece.osu.edu (E. Ekici).

1389-1286/$ - see front matter
doi:10.1016/j.comnet.2006.01.008

nodes close to the sinks may run out of energy before the others since all traffic is funneled through these nodes. Although battery replenishment and power harvesting techniques are under development, reducing energy consumption of individual sensor nodes still plays the most important role in maximizing the network lifetime.

Recently, mobile elements have been proposed as mechanical carriers of data in wireless sensor networks to address the connectivity and lifetime problems in WSNs [11,12]. In some application scenarios, such as urban traffic monitoring using stationary sensors, mobile elements such as vehicles already exist in the environment. In other scenarios such as surveillance of large and inhospitable areas with sparse sensor networks, mobile elements like UAVs can be incorporated into the design of the WSN. In both cases, the efficiency of data transfer from sensors to MEs depends heavily on the path followed by MEs. The ZebraNet project [2] and Manatee project [3,4] are among the first to explore the idea of using mobility in sensor networks. In [13], a three-tier MULE (Mobile Ubiquitous LAN Extensions) architecture has been proposed, where the mobile elements are vehicles (cars, buses) outfitted with transceivers and move randomly to collect data from the sensor nodes as they approach them. In other scenarios, mobile devices can be incorporated into the design of the WSN architecture. Message Ferrying is introduced in [14–16], where a set of special mobile elements (called message ferries) provide communication service for nodes in sparse ad hoc networks. Since each node communicates only with the message ferries, any long distance communication is avoided, resulting in increased node lifetime. In Wireless Sensor and Actor Networks (WSANs) [6], a set of mobile nodes, called *Actors*, are employed to perform appropriate actions based on the data collected by sensor nodes.

Using mobile elements, the battery lifetime of individual sensors can be increased by shifting the energy consumption burden for communication to MEs. MEs can easily be recharged periodically at a base station or switched with spare MEs as opposed to sensor nodes which do not possess energy replenishment capabilities. Low density networks also benefit from utilization of MEs. Furthermore, in disconnected networks, the use of MEs may be the only solution.

A mobile element capable of short-range wireless communication can collect data from the nearby sensor nodes as it approaches during its motion. We refer to data collection by mobile elements in sensor networks as "data harvesting". Controlling the mobile element motion leads to the *Mobile Element Scheduling* (*MES*) problem [17], which is defined as the problem of scheduling the visits of a mobile element to sensor nodes so that there is no data loss due to sensor node buffer overflow. It is assumed that all sensors have limited capacity buffers, when a mobile element visits a node, it transfers the data from the sensor to its own memory and the sensor's memory is freed.

The amount and frequency of data generation in sensor nodes varies based on the event occurrence frequency, which is generally a function of the sensor location. In the example given in [17], sensor networks can be used to sense air pollution levels in large urban areas. Since the variation of pollution levels are expected to be higher in industrial areas than in residential areas, sensors in industrial areas generate data at higher rates than sensors in residential areas. As a result, buffer overflow time of sensors at different locations may not be the same. To avoid buffer overflow, a node with a higher data generation rate may be visited multiple times before another one with a lower data generation rate is visited. As soon as a node is visited and data is transferred, its next visiting deadline is updated. Therefore, the deadlines are *dynamically* updated as the mobile element performs its job as a data collector.

In this paper, we investigate the data harvesting problem and propose a *Partitioning-Based Scheduling* (*PBS*) algorithm which tackles the related MES problem by dividing it into two sub-problems: Partitioning and Scheduling. First, all nodes are partitioned into several groups with respect to their data generation rates and locations. Then, within a single group, the scheduling algorithm generates node visiting schedules for the ME by minimizing the overhead of moving back and forth across faraway nodes. Finally, the scheduling solutions of the groups are concatenated forming the entire ME path so that all nodes can be visited at adequate frequencies to prevent any buffer overflow. In this paper, the investigated performance metrics are the data loss rate for a given ME speed and the minimum speed to completely avoid buffer overflow. We confirm through simulations that the proposed PBS algorithm performs well in terms of both metrics as well as providing high predictability in nodes inter-visit schedule.

The remainder of the paper is organized as follows: the related work is presented in Section 2, followed by a detailed description of the MES problem and the PBS algorithm in Section 3. Simulation results are presented and discussed in Section 4. Finally, the paper is concluded in Section 5.

2. Related work

The use of mobile elements to harvest data has recently been considered in the literature. Data MULES [13] focuses on utilization of mobile elements (called MULEs) in sparse sensor networks. The MULEs move randomly and collect data opportunistically from sensor nodes. The movement of data gathering elements are not controlled in this framework. In the message ferrying (MF) [14,15] approach, message ferries are used to route data from one node to another in a sparse ad hoc network. Based on a given traffic matrix, the goal of message ferrying approach is to find the optimal route of a ferry so that the average delay from source to destination is minimized while meeting the bandwidth requirement of flows.

Related to the MES problem are the Orienteering Problem (OP) [18], the Prize Collecting Traveling Salesman Problem (PC-TSP) [19], as well as the original TSP. These problems deal with routing a vehicle to visit each city at most once. However in our problem, a node may need to be visited more than once before all other nodes are visited because of the difference in buffer overflow deadlines. In OP and Prize Collecting TSP, each city has an associated non-negative *prize* and the vehicle aims to collect the maximum total prize. Although the mobile element in the MES problem also collects data that can be considered as prize, the value of the prize is dynamic and depends on the time of the visit.

The Vehicle Routing Problem (VRP) [20] is defined as finding a route for a vehicle that minimizes the total travel cost to deliver cargo between a depot and customers. Unlike TSP, VRP considers more than one vehicle and nodes can be visited more than once. Among many variants of VRP, VRP with deadline [21,22] and Periodic VRP (PVRP) [20] are relevant to the MES problem. The goal of *VRP with Deadline* is to schedule a vehicle to visit as many nodes as possible by their deadlines. Different from our problem, each node in Deadline VRP is visited at most once. Furthermore, the deadline of the visit to a node in Deadline VRP is fixed, whereas the deadline of a node changes dynamically in our case.

Periodic VRP is the problem of designing routes for delivery vehicles for a given T-day period where not all customers require delivery on every day in the period. Customers are associated with a set of feasible schedules that are some combinations of days they can be visited. In PVRP, the feasible solution set consists of a finite number of possibilities. However, in MES, the feasible solution set consists of an infinite number of possibilities such that the time difference between any two consecutive visits scheduled to the same node is smaller than the associated buffer overflow time. Moreover, in the MES problem, the vehicle does not need to go back to a certain node at the end of every cycle whereas the vehicles in PVRP go back to the depot every day. Although the MES problem can be discretized in the time domain, the resulting size of the feasible solution set does not scale well with the range of data generation rates.

The MES problem in wireless sensor networks is proved to be NP-complete and three heuristic algorithms are presented in [17]. The first one is the Earliest Deadline First (EDF) algorithm, where the node with the closest deadline is visited first. To improve EDF, the second algorithm, EDF with k-lookahead, is proposed. Instead of visiting a node whose deadline is the earliest, this algorithm considers the $k!$ permutations of the k nodes with smallest deadlines, and chooses the next node which leads to the earliest finish time. Consequently, the EDF with k-lookahead algorithm performs better than pure EDF. The third algorithm is the Minimum Weight Sum First (MWSF) algorithm, which accounts for the weights of deadlines as well as distances between nodes in determining the visiting schedule. The MWSF algorithm performs the best among the three proposed algorithms.

Even though the MWSF solution considers both deadlines as well as distances, "back-and-forth" movement between far away nodes occurs frequently. In our proposed PBS algorithm, we consider the deadline and distances of *all nodes* simultaneously and utilize a two-layer scheduling approach to reduce the back-and-forth movement behavior. This is achieved by partitioning the set of all nodes according to deadlines as well as their geographic locations. The resulting schedules and paths are usually shorter, which reduces the minimum required speed of the ME to prevent buffer overflow.

Algorithm 1. [PBS($\{w_{ij}\}, \{o_i\}$)]

1: Partition the nodes $\{n_i\}$ into M bins according to the buffer overflow times $\{o_i\}$.
2: Geographically partition each bin B_j into 2^{j-1} sub-bins using partitioning algorithm.
3: Calculate a TSP path for each sub-bin.
4: Concatenate TSP paths appropriately to build the overall schedule.

3. Partitioning-Based Scheduling algorithm

Our proposed Partitioning-Based Scheduling (PBS) algorithm is designed to solve the MES problem, and aims to schedule the visits of the mobile element to each sensor to avoid data loss due to sensor buffer overflow. With the PBS algorithm, we first partition all nodes into several groups, called *bins*, such that nodes in the same bin have similar deadlines. Then each bin is further divided into sub-bins with respect to geographical locations of the nodes. To decide the ME path within a single sub-bin, we solve the Traveling Salesman Problem, which computes a minimum cost tour that visits each node exactly once. Finally, the schedules for individual groups are concatenated to form the entire schedule. While calculating the schedules, the data transfer time between the sensor nodes and the ME is also considered. We first outline our notation and problem formulation and then present a detailed description of our solution to the MES problem in the remainder of this section.

3.1. Problem formulation and notation

In this paper, wireless sensor networks composed of homogeneous sensor nodes are considered. The nodes are equipped with wireless communication interfaces with limited ranges. Sensor nodes capture the events in their surroundings and record them to their buffers. The following assumptions are also made regarding the sensor nodes and the mobile element.

- The physical sizes of sensor nodes and the mobile element are negligible.
- The mobile element can move in any direction without any latency of making any turns.
- Data transfer time between sensor nodes and the mobile element is not negligible compared to the delay due to ME movement. Data transmission rate is denoted by r_{tr}.
- All sensors have the same finite buffer size, and at time $t = 0$, all sensor node buffers start in an empty state. The mobile element has sufficiently large data storage and it does not suffer from the buffer overflow problem.

We denote the number of nodes in the network by N and the set of nodes by $\{n_i\}$, where $i = 1, \ldots, N$. Let w_{ij} denote the distance between nodes n_i and n_j. The buffer overflow time and data generation rate of each node are denoted by o_i and r_g^i, respectively. For a buffer of size b, $o_i = \frac{b}{r_g^i}$, for $i = 1, \ldots, N$. We assume that the data generation rate is directly related to event occurrence rate.

The MES problem ($\{w_{ij}\}, \{o_i\}$) is to find a sequence of visits to nodes $\{n_i\}$, for $i, j = 1, \ldots, N$, and calculate the minimum speed $v_{\min}$ of the ME so that no node buffer overflow occurs. We attack the problem by first ignoring the transmission time between the sensor nodes and the mobile element, then including it appropriately into our calculations.

3.2. The proposed PBS algorithm

Let B_j, $j = 1, \ldots, M$, denote bin j, where M is the total number of bins. In PBS, nodes are first partitioned into bins in such a way that overflow times of the nodes in bin B_i is smaller than those in B_j, for $j > i > 0$. Moreover, the range of overflow times for nodes in B_{i+1} is twice that of B_i. This allows the nodes in B_i to be visited twice more frequently than the nodes in B_{i+1} during generation of the visit schedules. Then, each bin further is partitioned into sub-bins so that nodes in the same sub-bin are geographically close to each other. This two-level partitioning results in groups of nodes with similar deadlines and locations. Therefore, in each sub-bin, node visit schedule of the ME can be computed using a TSP solution. Finally, schedules for individual sub-bins are concatenated to form the entire schedule that guarantees all deadline constraints are satisfied. In the following, we describe the details of the PBS algorithm outlined in Algorithm 1.

3.2.1. Bin partitioning according to overflow times

Minimum and maximum overflow times in the network are defined as $o_{\min} = \min_i\{o_i\}$ and $o_{\max} = \max_i\{o_i\}$, respectively, for $i = 1, 2, \ldots, N$. Then,

nodes are assigned to bins according to the following equation:

$$n_i \in B_j, \quad \begin{cases} \text{if } 2^{j-1}o_{\min} \leqslant o_i < 2^j o_{\min}, \ j = 1, \ldots, M-1; \\ \text{if } 2^{j-1}o_{\min} \leqslant o_i \leqslant o_{\max}, \ j = M. \end{cases} \tag{1}$$

Fig. 1 shows an example of partitioning nodes into three bins. The overflow times of nodes in B_1 range from $o_{\min}$ to $2o_{\min}$, the overflow times of nodes in B_2 range from $2o_{\min}$ to $4o_{\min}$, and the overflow times of nodes in B_3 range from $4o_{\min}$ to $o_{\max}$. As far as PBS algorithm is concerned, there is no distinction between nodes in the same bin in terms of their overflow times. Therefore, all nodes in a bin B_j are considered as if they are assigned an overflow time of $2^{j-1}o_{\min}$. Every bin is then visited at different frequencies: all nodes in B_1 are visited every cycle, nodes in B_2 are visited every other cycle, and nodes in B_3 are visited every four cycles, where we define a *cycle* as a closed path among a set of nodes, such that no node is included more than once in the same cycle. We also define a *supercycle* as a closed path composed of concatenated cycles such that every node is included at least once in a supercycle. In our algorithm, the duration it takes for the ME to complete a supercycle is defined as the period of the ME schedule. The notions of cycle and supercycle in this context will be explained further in the following sections.

Algorithm 2. [2D-tree(S, *cutonx*, depth, d)]

```
1:  if d=depth then
2:     return
3:  endif
4:  if cutonx then
5:     S1 = {ni | xi ≤ (1/|S|) Σ_{nj∈S} xj}
6:  else
7:     S1 = {ni | yi ≤ (1/|S|) Σ_{nj∈S} yj}
8:  endif
9:  S2 = S − S1
10: cutonx ← !cutonx
11: KD-tree(S1, cutonx, depth, d + 1)
12: KD-tree(S2, cutonx, depth, d + 1)
```

3.2.2. Sub-bin partitioning according to locations

Each bin obtained in Step 1 is then partitioned into sub-bins according to the node locations such that the nodes in the same sub-bin are geographically close to each other. The number of sub-bins

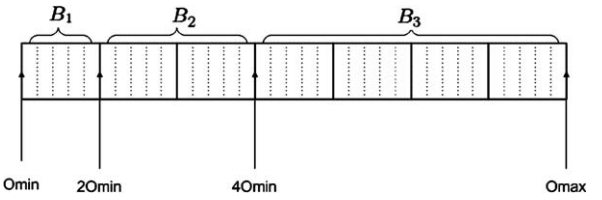


Fig. 1. The diagram of step 1: partition according to overflow times.

of a bin B_j is calculated based on the index j. As an example, the nodes in B_2 need to be visited only half as frequently as the nodes in B_1. Hence, B_2 is partitioned into two sub-bins: B_2^1 and B_2^2, where B_2^1 is visited every even cycle, and B_2^2 is visited every odd cycle. Following the same rule, B_3 is partitioned into four sub-bins: B_3^1, B_3^2, B_3^3 and B_3^4, and in general, bin B_i is partitioned into 2^{i-1} sub-bins: $B_i^1, \ldots, B_i^{2^{i-1}}$. We utilized the KD-tree algorithm [23] to realize this partitioning, since it is a simple algorithm with satisfactory performance. KD-tree is a k-dimensional binary search tree for information retrieval by associative searches. In our case, we use the 2D-tree where the two dimensions are the length and width of the sensor deployed field.

The pseudocode of the 2D-tree partitioning algorithm is shown in Algorithm 2. The input S is the set of nodes in the bin to be partitioned. *cutonx* is a boolean flag showing the cut criteria: x-coordinate (true), denoted by x_i, or y-coordinate (false), denoted by y_i for node n_i. d is the current cut level and *depth* is the desired depth of the KD-tree, which is determined by the requested number of sub-bins. In each procedure call, the nodes are partitioned into two, based on the cut criteria decided by *cutonx*. Then, *cutonx* flag is negated and the procedure is called recursively on resulting partitions until the desired 2D-tree depth is reached.

An example of 2D-tree partitioning is shown in Fig. 2 for 16 randomly deployed nodes in the given region. As shown in the figure, the nodes are first geometrically divided into two balanced parts by the cut A with respect to their x-coordinates. The nodes having x-coordinates smaller than the average of the x-coordinates is assigned to one part and rest to the other.

As the cut A partitions the region vertically, cuts B and C horizontally partition the resulting two parts considering the y-coordinates of the nodes. This process is repeated alternately until the desired number of partitions is obtained. The number of partitions, which is also the number of sub-bins in

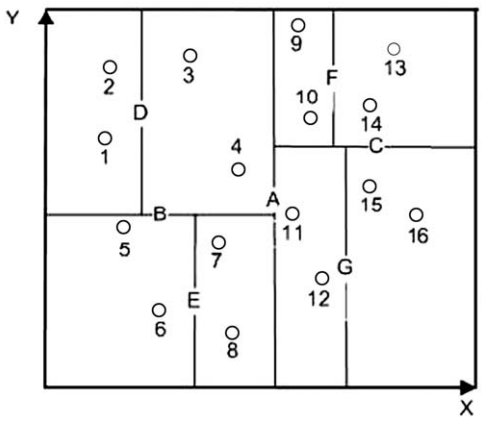


Fig. 2. Nodes in a two dimension space cut by KD-tree.

our problem, decides the depth of the 2D-tree. In our PBS solution, bin B_j is partitioned into 2^{j-1} sub-bins. Therefore, the depth of 2D-tree for bin B_j is

$$depth_j = \log_2 2^{j-1} = j - 1. \quad (2)$$

3.2.3. Forming a TSP solution on each sub-bin

The two-level partitioning results in groups of nodes with similar deadlines and locations. Therefore, the ME scheduling problem is reduced to the Traveling Salesman Problem (TSP) for each sub-bin. In the literature, several algorithms to calculate TSP paths are proposed such as the nearest neighbor, LKH [24], and Prim's algorithm [25]. In our solution, we adopt Prim's algorithm to calculate the TSP path. While any TSP algorithm can be incorporated into PBS, we adopted Prim's algorithm for its satisfactory TSP path length performance at the reasonable time complexity of $O(N^2)$ [26]. In PBS, the path of the ME is slightly different from the computed TSP path such that the ME does not return back to the first visited node after visiting the last node in the sub-bin. Instead, the ME visits the first node of the next sub-bin. Then, it follows the computed ME path in that sub-bin and proceeds to the following sub-bin. As a special case, in B_1, the node with the minimum overflow time is taken as the start node.

3.2.4. Forming the supercycle

After the ME paths within the sub-bins are calculated, the visit order of the sub-bins should be decided to form the complete ME path. At the end of partitioning, there are 2^{i-1} sub-bins of bin B_i, each composed of nodes with deadlines at least twice the deadlines of the nodes in sub-bins of B_{i-1}, $i = 1, \ldots, M$. Therefore, in a reasonable ME schedule, sub-bins in the same bin should be visited with the same frequency and a sub-bin in B_{i-1} should be visited twice more frequently than a sub-bin of bin B_i. This heuristic choice results in a sub-bin of B_i to be visited 2^{M-i} times for each visit to a sub-bin of bin B_M. Recall that in a supercycle each sub-bin, hence each node, is visited at least once. In other words, each sub-bin in the least frequently visited bin B_M is visited at least once. Without loss of generality, let the ME visit each sub-bin in B_M exactly once in a supercycle. Then, a sub-bin of bin B_i should be visited exactly 2^{M-i} times in a supercycle according to our heuristic choice.

Let $I_{i,j}$ be defined as the maximum duration between two consecutive visits to a node in sub-bin B_i^j. Then, the sufficient condition to avoid buffer overflow for all nodes of B_i^j can be stated as

$$I_{i,j} \leqslant o_{\min} \times 2^{i-1}. \quad (3)$$

Let $L_{i,j}$ denote the longest ME path between two consecutive visits to a node of sub-bin B_i^j, i.e., $L_{i,j} = I_{i,j} \times v$, where v is the speed of the ME. Hence, to avoid buffer overflow in B_i^j, $v \geqslant \frac{L_{i,j}}{o_{\min} \times 2^{i-1}}$ should be satisfied. This can be achieved by either increasing the ME speed or decreasing $L_{i,j}$. Our objective of minimizing the ME speed for a lossless schedule can be achieved by minimizing $L_{i,j}$ for each sub-bin and setting v to the largest required value to satisfy Inequality (3) for every sub-bin.

Since all sub-bins are formed according to geographical proximity as well as overflow deadlines, the TSP tours of sub-bins B_i^j of bin B_i of similar lengths. In order to have a predictable visiting schedule of sub-bins, we form cycles such that only one sub-bin from each bin is contained in a cycle. Furthermore, all cycles preserve the order of bins B_i from which sub-bins are selected. In particular, sub-bins are visited starting from B_1 in increasing bin number order and a sub-bin in B_i is always visited after the same sub-bin in B_{i-1}. This ensures that in every cycle that a sub-bin B_i^j is visited, the nodes in B_i^j are visited at exactly the same time as they were visited in the previous cycle relative to the start times of the cycles. The rationale behind this choice will be discussed in more detail in Section 3.4.

There are two times more sub-bins in bin B_i than B_{i-1} and each sub-bin in B_i is always visited after the same sub-bin in B_{i-1}. As a result, exactly two sub-bins in B_i follow each sub-bin in B_{i-1} for $i = 2, \ldots, M$. In PBS, the two sub-bins to follow a particular sub-bin is decided by considering the geo-

graphic locations of the sub-bins. We use the center of gravities of sub-bins to measure the distances between sub-bins. For each sub-bin of B_{i-1}, we greedily schedule the closest two sub-bins from B_i to follow it. Note that more complicated concatenation algorithms can be used to further minimize the distances between consecutively visited sub-bins.

In Fig. 3, a visiting sequence example of sub-bins in a supercycle is given for $M = 3$. The supercycle in this case consists of four cycles and the visit schedules of the sub-bins are as follows: B_1 is visited every cycle, B_2^1 is visited in cycles 1 and 3, and B_2^2 is visited in cycles 2 and 4 and the sub-bins of the last bin, B_3, are visited in cycles 1, 2, 3, and 4, one at a time.

3.3. Data transfer time considerations

Let f_i denote the minimum visit frequency for node n_i to avoid sensor buffer overflow. Without considering the data transmission time and assuming a lossless schedule, the maximum duration between two consecutive visits to n_i is o_i. Thus, $f_i = \frac{1}{o_i}$. Considering the transmission bandwidth, f_i is determined by n_i's data generation rate r_g^i, transmission rate r_{tr} and buffer size b. Let t_{tr} denote the transmission time, which is the duration between the ME visits the sensor and leave the sensor. Then

$$t_{\text{tr}}^i = \frac{\beta b + t_{\text{tr}}^i r_g^i}{r_{\text{tr}}}, \tag{4}$$

$$t_{\text{tr}}^i = \frac{\beta b}{r_{\text{tr}} - r_g^i}, \tag{5}$$

$$= \frac{\beta b o_i}{o_i r_{\text{tr}} - b}, \tag{6}$$

where β ($\beta \leqslant 1$) is the percentage fill level of the buffer of the sensor i. The maximum duration between two consecutive visits is $o_i + t_{\text{tr}}$. Therefore, using $o_i^{\text{new}} = o_i + t_{\text{tr}}$ as the new overflow time in PBS step 1 (bin partitioning) will generate a schedule for MES with transmission time consideration. Note that β is node specific and is not known beforehand. The case of $\beta = 1$ refers to completely full buffers and the transmission time is the maximum. Therefore, a β value of 1 is chosen in our algorithm as we are interested in the lowest ME speed which can guarantee a lossless schedule.

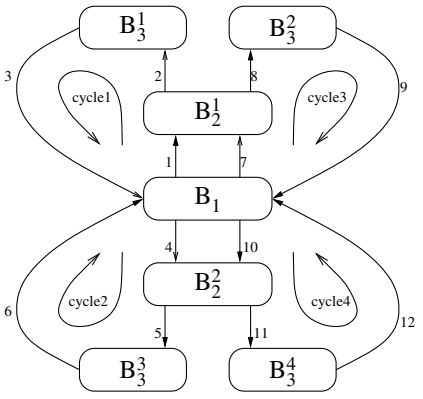


Fig. 3. An example of the visiting sequence of sub-bins in a *supercycle*.

3.4. Discussion of minimum required speed

To minimize the power consumption of the ME, any solution to the MES problem should minimize the ME speed. In this paper, we calculate a lower bound for the PBS solutions on the ME speed, denoted as $v_{\min}$, such that no buffer overflow occurs in the network nodes. Considering an arbitrary node n_i in bin B_j, let $L_{i,k}$ and $C_{i,k}$ denote the path that ME has traveled and the set of nodes ME has visited between the kth visit and $(k+1)$th visit to n_i, respectively. The transmission time at node n_i is denoted by tr_i. To avoid buffer overflow at node n_i, the following condition should be satisfied:

$$2^j o_{\min} \geqslant o_i \geqslant \frac{L_{i,k}}{v} + \sum_{n_s \in C_{i,k}} tr_s, \tag{7}$$

where $\frac{L_k}{v}$ is the time ME spends on traveling between nodes and $\sum_{n_s \in C_{i,k}} tr_s$ is the total data transmission time. To guarantee no data loss at n_i, the sum of these two terms should be less than or equal to the overflow time of n_i. According to Eq. (1), we have $o_i \leqslant 2^j o_{\min}$. Note that it is hard to find the data transmission time for each node, but we can easily find the upper bound, which is the transmission time for a full buffer by setting $\beta = 1$ in Eq. (6)

$$tr_i \leqslant \frac{b o_i}{o_i r_{\text{tr}} - b}. \tag{8}$$

With Eqs. (7) and (8), we have

$$v \geqslant v_{\min}^i = \frac{L_{i,k}}{2^j o_{\min} - \sum_{n_s \in C_{i,k}} \frac{b o_i}{o_i r_{\text{tr}} - b}}, \tag{9}$$

$$v_{\min} = \max_{\forall n_i} v_{\min}^i. \tag{10}$$

As mentioned in Section 3.2, the PBS algorithm generates a periodic schedule. Therefore, there is no need to keep track of all visits to the nodes. For a node in bin B_j, if there is no overflow during 2^{j-1} consecutive visits, which occur during a single supercycle, no overflow condition is guaranteed.

In general, $L_{i,k}$, $i = 1,2,\ldots,N$, $k = 1,2,\ldots,2^{j-1}$, cannot be approximated easily except for certain special deployments of sensor nodes. Once an ME schedule is generated by the PBS algorithm, L_k and $v^i_{\min}$, therefore $v_{\min}$, can be computed numerically. If the ME is constrained to move at a smaller speed than $v_{\min}$, there may be data losses due to buffer overflow. Besides trying to minimize $v_{\min}$ for a lossless schedule, the proposed algorithm is also designed to minimize the data loss rate in such cases. Although not discussed in this paper, further optimizations can be done for the latter objective by trading off performance of the former one. In the performance analysis section, we also present the performance of our PBS algorithm as a function of the ME speed.

3.5. Time complexity analysis

Let N and M denote the number of sensor nodes and number of bins, respectively. Then, the complexity of PBS algorithm steps (Algorithm 1) are as follows:

- *Step* 1: Partitioning with respect to overflow times can be achieved by comparing the overflow time of each node with the bin boundaries resulting in complexity of $O(NM)$.
- *Step* 2: In the geographical partitioning phase, the number of times a node in bin B_i is processed and assigned to a sub-partition is upper bounded by $i-1$. Since assignment is done by simply comparing the requested coordinate of the node with the average, it is O(1). In the worst case, all nodes except the one with the minimum overflow time are in bin B_M, therefore the complexity of the 2D-tree algorithm is $O(NM)$.
- *Step* 3: The complexity of calculating TSP path by Prim's algorithm for N nodes is $O(N^2)$. Assume that each sub-bin is assigned a unique identity from the set $U = \{1,2,\ldots,2^M-1\}$. Recall that 2^M-1 is the total number of sub-bins. Let k_i denote the number of nodes in the sub-bin with identity $i \in U$. Then, overall complexity of computing TSP for every sub-bin is $O(k_1^2 + k_2^2 + \cdots + k_{2^M-1}^2)$. Since

$$k_1^2 + k_2^2 + \cdots + k_{2^M-1}^2 \leqslant (k_1 + k_2 + \cdots + k_{2^M-1})^2 \quad (11)$$

$$= N^2 \quad (12)$$

  the time complexity of this step is $O(N^2)$.
- *Step* 4: In concatenation step, first the center of gravity for each sub-bin is calculated in $O(N)$ time. For each sub-bin B_i^j, two sub-bins in B_{i+1} with center of gravities closest to that of B_i^j are selected to be visited after B_i^j. If a selected sub-bin is already decided to be visited after another sub-bin in B_i, the next closest sub-bin to B_i^j is considered. Since there are 2^{i-1} sub-bins in B_i, it takes $2^{i-1} \times 2^i = 2^{2i-1}$ comparisons for each bin. Therefore, the time complexity of the concatenation step is $O(N + 4^M)$.

Consequently, the PBS algorithm has an overall time complexity of $O(N^2 + 4^M)$.

4. Performance evaluation

To evaluate the performance of our proposed PBS algorithm considering transmission time, we have run an extensive set of simulations. In this simulation study, the following scenarios are considered for performance evaluation.

- Simulation I: We observe the data loss rate as a function of the mobile element speed.
- Simulation II: We observe the effect of the node density on the minimum required ME speed to prevent data loss in the PBS algorithm.
- Simulation III: The effect of number of bins on the performance of PBS algorithm is evaluated for different network sizes and properties.
- Simulation IV: We observe the impact of sensor buffer size in sensor nodes on the performance of the PBS algorithm.
- Simulation V: The effect of wireless transmission rate between sensor nodes and ME on the data loss rates is analyzed.
- Simulation VI: Sensor visit predictability is investigated as a function of overflow time and node density through inspection of the standard deviation of inter-visit times.

All simulations except for Simulation III are also run for the MWSF [17] algorithm to compare the performance of the algorithms. The details of the simulation setup as well as the detailed discussion of the results are presented in the following sections.

4.1. Metrics and methodology

We observe the following metrics to evaluate the performance of the PBS algorithm.

- *Data loss occurrence rate* is defined as the ratio of the number of sensors missing their deadlines to the total number of nodes visited. It presents the performance in terms of the number of sensors missing their deadlines rather than the amount of data lost.
- *Data loss rate* is defined as the ratio of the data lost due to buffer overflow to the total amount of data generated.
- *Minimum required speed* is defined as the minimum speed of the mobile element to prevent any buffer overflow.
- *Predictability* is defined as the standard deviation of the inter-visit duration which is the duration between two visits of the ME to the same node.

The simulator is developed in C++ language. A graph generator forms the random sensor network topology and determines the sensor data generation rates and overflow times according to the scenario at hand. Given the geographical topology of sensor nodes as well as the distribution of sensor overflow time, a schedule for the ME to visit sensor nodes is generated by the PBS algorithm. Then, we simulate the visits of the ME to each sensor node in a discrete manner by keeping track of the time. After each visit, the current time is incremented by the time that ME spends traveling from the previous node to the current node. If the time between two successive visits of the ME to the same node exceeds the node's deadline, then data loss occurs. In such a case, the loss is recorded in the simulation.

In our simulations, we use the following default settings unless specified otherwise. Each simulation is run on a network with 200 sensor nodes randomly placed following the uniform distribution on a 200 m × 200 m area. Each sensor node is equipped with same size buffers (10 Mb). Data transmission rates between sensor nodes and the ME is 500 kb/s. The data generation rate of each sensor differs due to different event occurrence rates at different regions. To simulate this, we assume that events are concentrated at certain locations, which we call *Eyes*. The nodes in the eye centers have the highest data generation rates, which drops radially outward. Four topologies,[1] A, B, C, and D, are considered in our simulations. Topologies A, B, and C have one, four, and nine eyes, respectively. Topology D corresponds to uniformly distributed data generation rates over the sensor network. It can also be considered as having infinite number of eyes. As shown in Fig. 4(a), in topology A, a sequence of concentric circles divides the given area into several ring shaped regions; Region 1 to Region n. The radius of each concentric circle is denoted by $R_1, R_2, R_3, \ldots, R_n$, where $R_1 = 20$ m. The value of each radius is calculated as

$$R_i = i \cdot R_1, \quad i = 1, \ldots, n.$$

The nodes in the innermost region are assigned the smallest overflow time, which is called the *base_time*, and overflow times for nodes in regions radially outward are calculated as: $overflowTime_i = base\_time + (i-1) \cdot step$, $i = 1, \ldots, n$, where $overflowTime_i$ is the overflow time assigned to nodes in Region i and *step* is the size of the increments. In our simulations, we take *base_time* as 500 s and 500 s for *step*. Similarly, we consider the grids with four eyes and nine eyes as shown in Fig. 4(b) and (c), respectively. We assume the data size is 16bits. Therefore, for a node with sampling frequency of 100 Hz, the data generation rate is 1.6 kb/s and the corresponding buffer overflow time is 6250 s. To ensure that all topologies have the same *overflowTime* distribution, overflow times are distributed as follows: In Topology B with four eyes, the smallest circle has a radius of 10 m, and radius increases by $\frac{1}{2}$step. Similarly, in Topology C, the smallest radius is $\frac{2}{3}$units and radius increases by $\frac{1}{3}$step. In Topology D, nodes are first generated in the same way as in topology A and then deployed randomly in a space.

Each run of simulations lasts 10 Ms. Each data point in the figures correspond to the average of 500 independent runs. Ninety nine percent confidence intervals are also shown for each data point. We have run the experiments for both PBS and MWSF algorithms on all four topologies to make the comparisons. The MWSF algorithm is run with weight $\alpha = 0.1$, where α is the weight of deadline, and $1 - \alpha$ is the weight of distance. According to the experimental results in [17], MWSF algorithm yields the best performance when the value of α is around 0.1. We considered PBS with the default bin number $M = 3$ unless specified otherwise.

4.2. Impact of the ME speed on data loss

Fig. 5(a) and (b) show data loss occurrence rates and data loss rates for simulations run with both

[1] 'Topologies' A, B, C, D refer to the distribution of data generation rates over the sensor field, where all nodes are placed randomly.

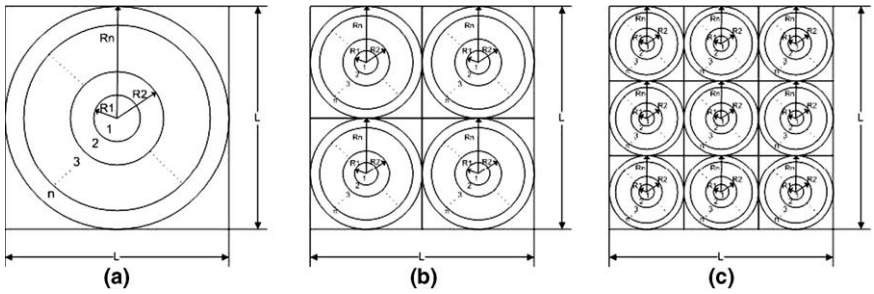


Fig. 4. Three types of topologies considered in simulation. Each topology A, B, and C have one, four, and nine eyes, respectively. (a) Topology of type A, (b) topology of type B and (c) topology of type C.

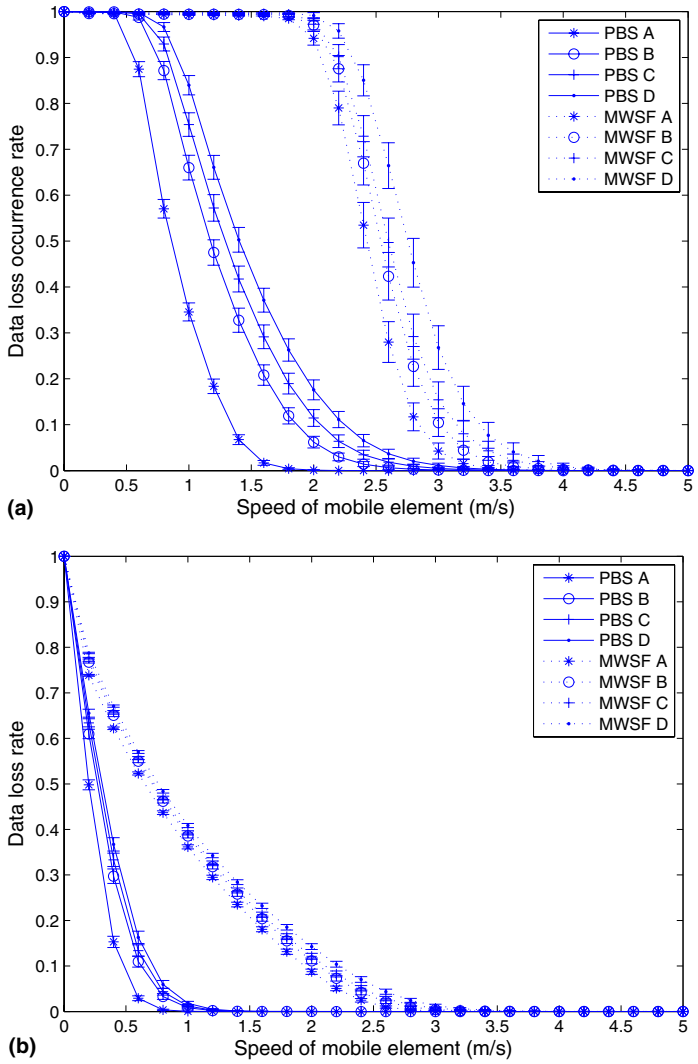


Fig. 5. Sensor loss rate under different mobile element speed for PBS and MWSF algorithms on topologies A, B, C and D. (a) Data loss occurrence rate and (b) data loss rate.

PBS and MWSF algorithms on topologies A, B, C, and D. We have collected experimental results for different speeds of the mobile element ranging from 0 m/s to 5 m/s. Loss rates range from 0 to 1, which correspond to the no loss and complete loss cases, respectively. In both figures, loss rates decrease with the increased speed. Furthermore, the performance on Topology A is better than Topologies B and C, which are more similar to each other. Topology D results in the worst performance. The ME covers larger distances when nodes with similar overflow times are not located closely together. The simulation results shown in Fig. 5(a) and (b) also match in the sense that the data loss occurrence rate and data loss rate drop to zero at the same ME speed. Compared to MWSF, the loss rate of PBS algorithm decreases at a higher rate as the speed increases. Therefore, we conclude that the PBS algorithm is more effective in terms of reducing the loss rate.

4.3. Impact of node density on minimum required ME speed

In this section, the impact of the node density is evaluated to observe the scalability of the PBS algorithm. In this simulation the number of nodes deployed on a 1 km^2 area is varied between 150 and 200. With the increasing node density, we measure the minimum required speed of the ME to avoid buffer overflow. Fig. 6 shows the results of running this simulation on four topologies. As expected, the minimum required speed increases with the increasing node density. When the node density increases, the path length in a supercycle increases, as well. This leads to a higher ME speed. The minimum required speed by the PBS algorithm is in general lower than the MWSF algorithm on the same topology. The minimum required speed for Topology A is lower than B, which is lower than C. The minimum required speed for Topology D is the highest. This behavior also matches previous results in Section 4. The path that ME covers in a supercycle is longer when the number of eyes is larger.

4.4. Impact of number of bins on data loss

In this section, we study how the number of bins M affects the performance of the PBS algorithm. $M = 1$ corresponds to the case where all the nodes are contained in one bin and are visited every cycle. No geographical partitioning is used in this case. When $M > 1$, nodes are first partitioned according to overflow times and then partitioned geographically. According to Figs. 7–10, $M = 1$ results in a lower loss rate when the ME speed is small, whereas as the ME speed increases using larger number of bins becomes more beneficial. When $M = 1$, the supercycle is equivalent to the TSP solution over all nodes. On the other hand, if $M > 1$, the supercycle consists of TSP solutions in each sub-bin, and the paths introduced due to concatenation step. Therefore, it is expected that with $M = 1$, the ME

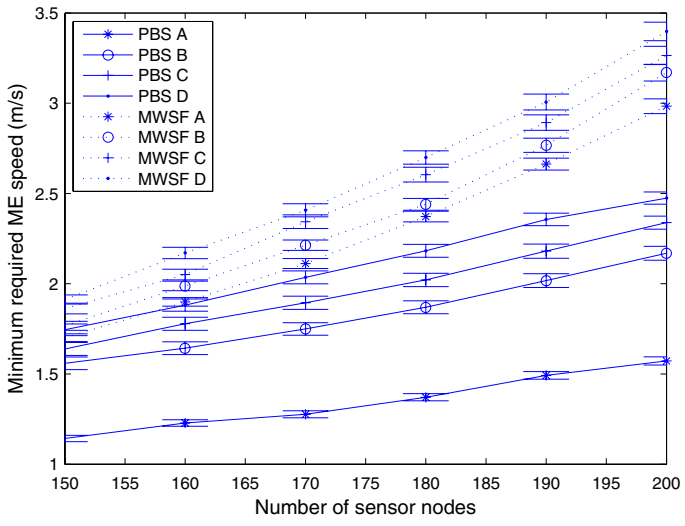


Fig. 6. The impact of node density on the minimum required ME speed of PBS and MWSF algorithm on topologies A, B, C and D.

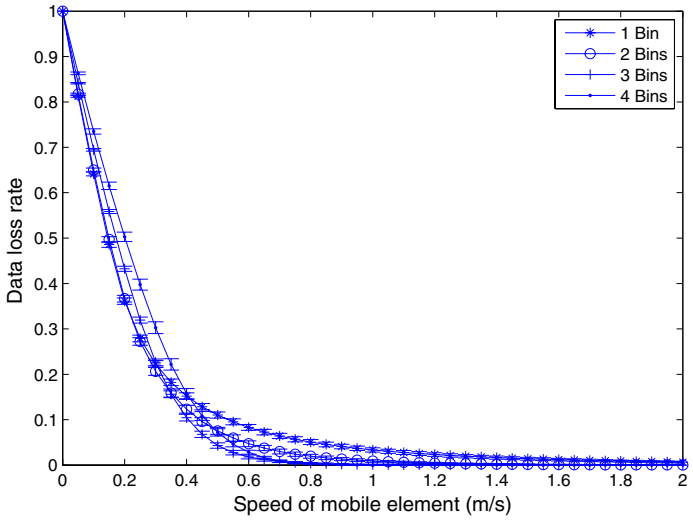


Fig. 7. Impact of number of bins in PBS Algorithm on data loss rate on topology A.

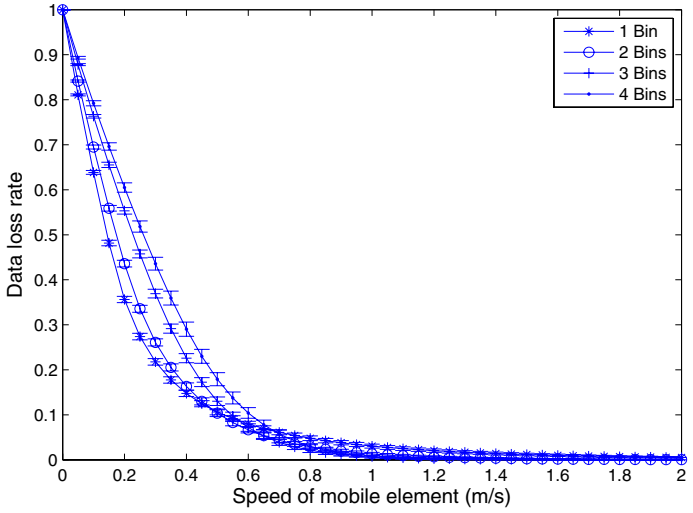


Fig. 8. Impact of number of bins in PBS Algorithm on data loss rate on topology B.

visits a larger number of nodes within a given time period compared to $M > 1$. If the ME speed is very small, almost every node buffer will be full at the time the ME visits them. Hence, by visiting a larger number of nodes in a given time period, data loss rate is smaller with $M = 1$. As the ME speed gets larger, visiting nodes with high overflow times as frequently as those with low overflow times will introduce redundancy for $M = 1$ case. As the number of bins gets larger, this redundancy is less pronounced since nodes are better distinguished with respect overflow times. As a result, with increasing ME speed, the concatenation overhead is overwhelmed by the benefits of reduced redundancy thanks to classification with respect to overflow times.

It can also be observed that a large number of bins provides much smaller minimum speed to avoid buffer overflow. As an example, in Fig. 7, when the data loss rate for $M = 4$ case drops to zero, the curve for $M = 1$ case still asymptotically approaches x-axis. Scheduling with small number of bins sacrifices performance on nodes with low overflow times severely. Therefore, although the performance of scheduling with one bin scheme has smaller data loss when mobile element moves at lower speeds, its advantage disappears for moderate and high speeds.

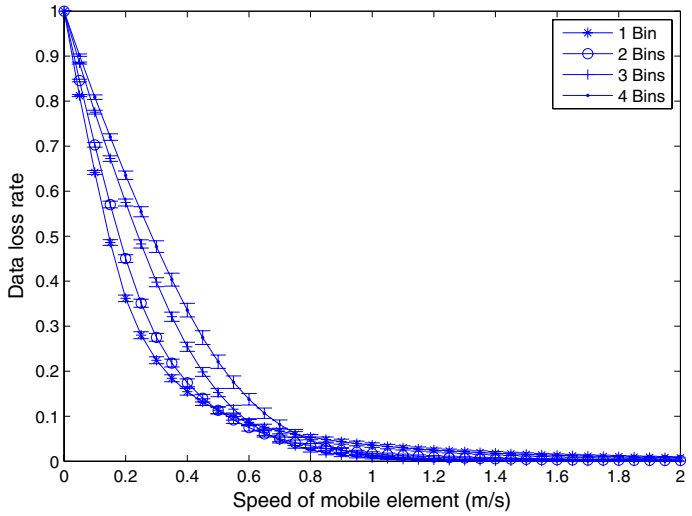


Fig. 9. Impact of number of bins in PBS Algorithm on data loss rate on topology C.

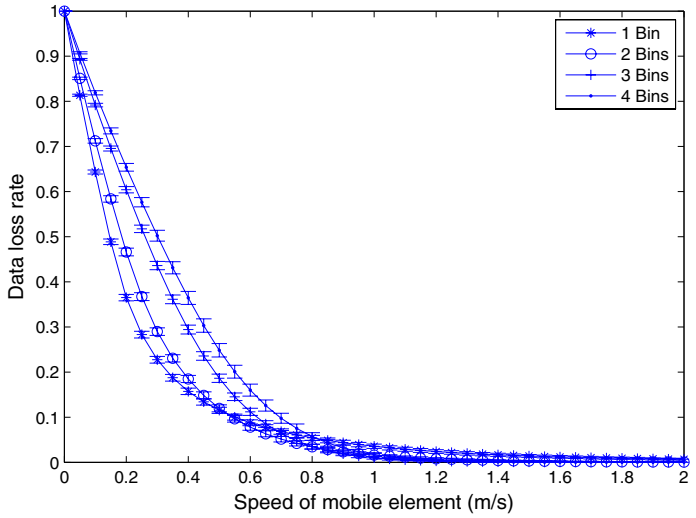


Fig. 10. Impact of number of bins in PBS Algorithm on data loss rate on topology D.

4.5. Impact of buffer size

Fig. 11 shows the impact of buffer size on data loss rate for both PBS and MWSF on different topologies. The buffer size ranges from 1 Mb to 80 Mb. The ME is running at a speed of 1 m/s. We observe in Fig. 11 that the data loss rates of both PBS and MWSF decrease as buffer size increases. With the larger buffer but the same data generation rate, all sensors have more capacity to carry data and have larger overflow time. When the ME is running at the same speed as before, it can collect more data. Thus the data loss rate decreases with increasing buffer size. We also notice that the performance of PBS is much better than MWSF except when the buffer size is extremely low. On Topology A, the data loss rate of PBS decreases to zero when buffer size is around 10 Mb. However, in MWSF still over 20% of the data is lost even when buffer size is over 50 Mb. For PBS with the same buffer size, data loss rate on Topology A is smaller than Topology B, which is smaller than Topology C. Topology D has the largest loss rate. This is due to the traveling overhead within each bin when there are more eyes in the deployment area. When the buffer size goes to infinity, there should not be any data loss in the ideal case. Therefore, all curves converge to zero.

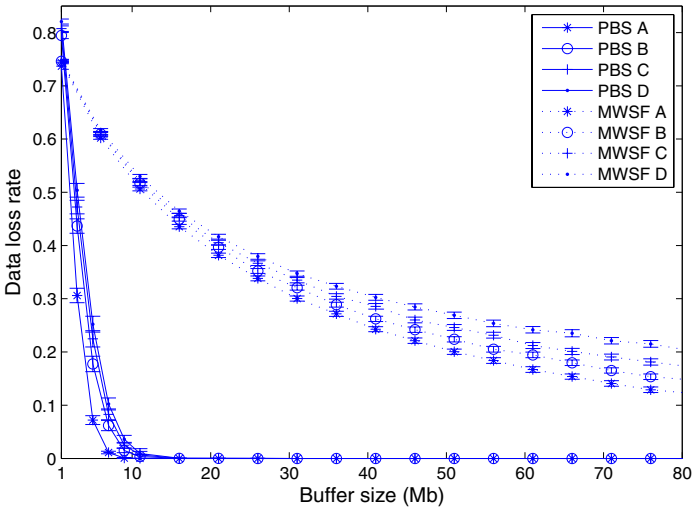


Fig. 11. Impact of buffer size on the performance of PBS algorithm.

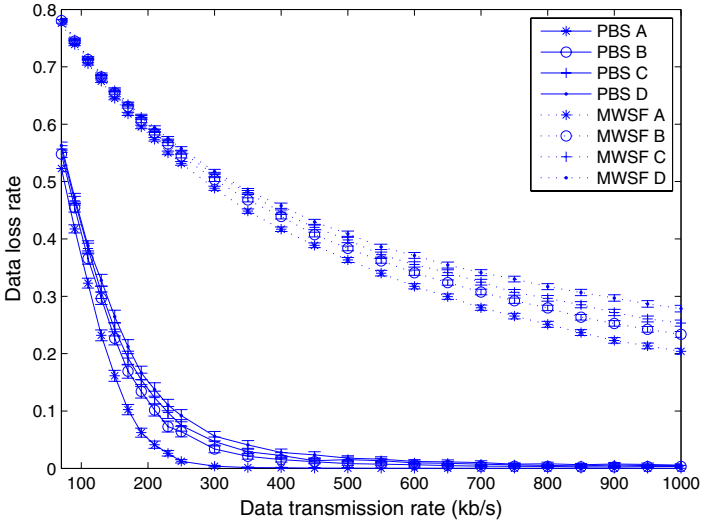


Fig. 12. Impact of data transmission rate on the performance of PBS algorithm.

4.6. Impact of transmission rate

The wireless transmitter on the sensor nodes also has impact on the trade-off between cost and performance. The transmission rate affects the power consumption at each sensor node, which is the current major bottleneck of WSNs. Fig. 12 shows the impact of transmission rate on the performance of PBS and MWSF on four topologies. Transmission rate ranges between 0.1 Mb/s and 1 Mb/s. The speed of the ME is chosen as 1 m/s. For both PBS and MWSF, the data loss rate is decreasing with the increasing transmission rate. When the transmission rates are the same, performance of PBS in Topology A is better than Topology B. Performance in Topology B is better than Topology C, which is better than Topology D. On same topology, data loss rate for PBS is less than MWSF. Larger transmission rates lead to smaller transmission times. Spending smaller portion of time on data transmission, the ME visits more nodes before their buffers overflow. Thus, the data loss rate decreases as transmission rate increases. When the data transmission rate goes to infinity, the ME actually visits one sensor node, then moves to the next node immediately without stopping. Hence, all curves in Fig. 12 are expected to converge to non-negative values.

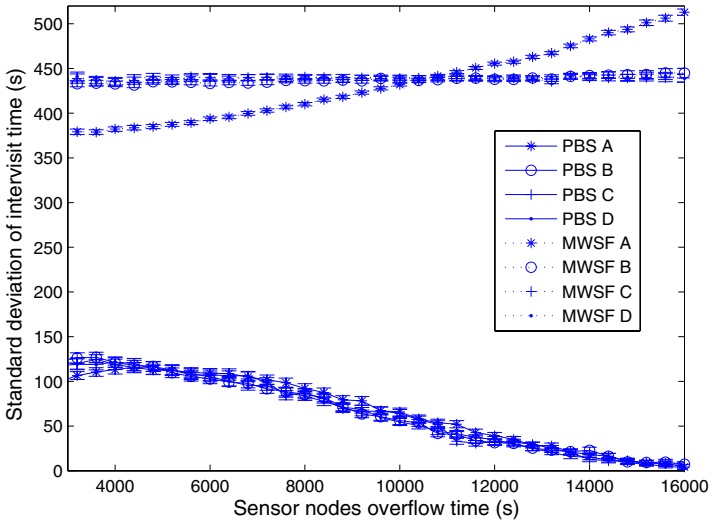


Fig. 13. Comparison of predictability between PBS and MWSF algorithms.

4.7. Sensor visit predictability

In Section 3, we claim that our PBS algorithm provides periodic supercycle scheduling. While this in correct, the inter-visit times for a given sensor in a supercycle are not necessarily equally spaced. In this section, we analyze the variation between inter-visit times of the ME to a given sensor node for different sensor overflow times. The standard deviation of inter-visit durations is used to measure the predictability of visits to sensor nodes, which shows the periodical feature of our scheduling scheme. We observe the predictability of sensor node visit times as a function of different data generation rates. We focus on the standard deviations of inter-visit durations and group the results according to overflow times. Fig. 13 shows the results of running PBS and MWSF algorithms on Topologies A, B, C and D. To guarantee the confidence interval, each result is the average value of 5000 independent runs. We choose 1 m/s as the speed of the mobile element, where buffer overflows still occur. As expected, Fig. 13 shows that the standard deviation of inter-visit times for PBS is much smaller than MWSF for all overflow time values. Compared with the MWSF algorithm, PBS provides higher predictability for the visits to sensor nodes due to higher periodicity.

Furthermore, note that the standard deviation of inter-visit times of PBS decreases as the overflow time of sensor nodes increases in general. Especially for nodes with high overflow times, the standard deviation is nearly zero. Therefore, the PBS algorithm favors nodes with high overflow times with high predictability. This is due to the fact that the nodes with high overflow times are visited less frequently and more periodically. Especially the nodes of the last bin are visited once every supercycle. The length of the supercycle is always the same. But due to different length of cycles, the transmission times for nodes are not always the same. Therefore, the period of supercycles are almost the same with some small variations. The standard deviation of inter-visit times for these nodes is close to zero.

Notice that for MWSF in Topology A, the standard deviation of inter-visit duration increases as overflow time increases, while Topology B, C and D do not reflect such behavior. Examining the MWSF scheme with topology A, since nodes in the central region have smaller overflow times, mobile element would visit this region more frequently. This leads the ME to visit nodes with small overflow times with more predictably. Because of the long distance between the edge areas and the center of the eye in topology A, the mobile element's visit is more unpredictable for the nodes in the edge area. However, in topology B with four eyes and topology C with nine eyes, nodes with high visiting frequencies is more spread over the entire region and the predictability is more balanced for all nodes.

5. Conclusion and future work

Using a controlled mobile element is a promising approach for data harvesting from sparsely

deployed sensor nodes. The sensor nodes may have different data generation rates, which leads to the Mobile Element Scheduling Problem. In this paper, we proposed a Partitioning-Based Scheduling (PBS) algorithm to address this problem. We compared our algorithm with Minimum Weighted Sum First algorithm and showed that our PBS algorithm provides higher performance in terms of decreasing loss rate, reducing the minimum required speed, and providing high predictability. Our future work includes investigation of methods to utilize more than one mobile element for data collection and to cater to the needs of urgent real-time communication events.

References

[1] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, J. Anderson, Wireless sensor networks for habitat monitoring, in: Proceedings of ACM Wireless Sensor Networks and Applications Workshop (WSNA 2002), 2002.

[2] P. Juang et al., Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet, in: International Conference on Architectural support for programming languages and operating systems, 2002.

[3] Manatee website, http://distlab.dk/manatee/, 2002.

[4] A. Beafour, M. Leopold, P. Bonnet, Smart tag based data dissemination, in: ACM Workshop on Wireless Sensor Networks and Applications, 2002.

[5] M. Vahdat, D. Becker, Epidemic routing for partially connected ad hoc networks, Technical Report, Duke University, 2000.

[6] I.F. Akyildiz, I.H. Kasimoglu, Wireless sensor and actor networks: research challenges, Ad Hoc Networks Journal 2 (October) (2004) 351–367.

[7] J.-H. Chang, L. Tassiulas, Energy conserving routing in wireless ad hoc networks, in: Proceedings of the 19th IEEE INFOCOM, 2000.

[8] A. Manjeshwar, D. Agrawal, Teen: a routing protocol for enhanced efficiency in wireless sensor networks, in: International Proceedings of 15th Parallel and Distributed Processing Symposium, 2001, pp. 2009–2015.

[9] K. Kar, M. Kodialam, T. Lakshman, L. Tassiulas, Routing for network capacity maximization in energy-constrained ad hoc networks, in: Proceedings of the 22th IEEE INFOCOM, 2003.

[10] A. Sankar, Z. Liu, Maximum lifetime routing in wireless ad hoc networks, in: Proceedings of the 23nd IEEE INFOCOM, 2004.

[11] Q. Li, D. Rus, Sending messages to mobile users in disconnected ad hoc wireless networks, in: MobiCom'00: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, ACM Press, 2000, pp. 44–55.

[12] M. Grossglauser, D.N.C. Tse, Mobility increases the capacity of ad hoc wireless networks, IEEE/ACM Transactions on Networking 10 (4) (2002) 477–486.

[13] R. Shah, S. Roy, S. Jain, W. Brunette, Data mules: modeling a three-tier architecture for sparse sensor networks, in: IEEE Workshop on Sensor Network Protocols and Applications (SNPA), 2003.

[14] W. Zhao, M.H. Ammar, Message ferrying: proactive routing in highly-partitioned wireless ad hoc networks, in: Proceedings of the Ninth IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS'03), 2003.

[15] W. Zhao, M. Ammar, E. Zegura, A message ferrying approach for data delivery in sparse mobile ad hoc networks, in: MobiHoc'04: Proceedings of the 5th ACM International Symposium on Mobile Ad hoc Networking and Computing, ACM Press, 2004, pp. 187–198.

[16] A. Kansal, A.A. Somasundara, D.D. Jea, M.B. Srivastava, D. Estrin, Intelligent fluid infrastructure for embedded networks, in: MobiSYS'04: Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services, ACM Press, 2004, pp. 111–124.

[17] A.A. Somasundara, A. Ramamoorthy, M.B. Srivastava, Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines, in: 25th IEEE International Real-Time Systems Symposium (RTSS'04), 2004, pp. 296–305.

[18] B.L. Golden, L. Levy, R. Vohra, The orienteering problem, Naval Research Logistics 34 (1987) 307–318.

[19] R. Bar-Yehuda, G. Even, S.M. Shahar, On approximating a geometric prize-collecting traveling salesman problem with time windows, Journal of Algorithm 55 (April) (2005) 76–92.

[20] P. Toth, E.C. Vigo, The vehicle routing problem, Society for Industrial and Applied Mathematics (SIAM), 2001.

[21] N. Bansal, A. Blum, S. Chawla, A. Meyerson, Approximation algorithms for deadline-tsp and vehicle routing with time-windows, in: STOC '04: Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing, ACM Press, 2004, pp. 166–174.

[22] M. Solomon, Algorithms for the vehicle routing and scheduling problem with time window constraints, Operations Research 35 (2) (1987) 254–265.

[23] J.L. Bentley, Multidimensional binary search trees used for associative searching, Communications of the ACM 18 (9) (1975) 509–517.

[24] J. Bentley, Fast algorithms for geometric traveling salesman problem, ORSA Journal on Computing 4 (1992) 387–411.

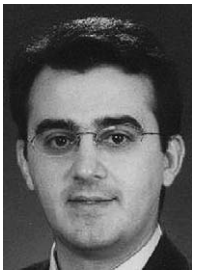
[25] T.H. Cormen, C.E. Leiserson, R.L. Rivest, et al., Introduction to Algorithms, The MIT Press, 2002.

[26] G. Gutin, A.P. Punnen, The Traveling Salesman Problem and Its Variations, Kluwer Academic Publishers, 2002.

Yaoyao Gu has received her B.S. degree in Electrical Engineering from Beijing University in China. Currently, she is working towards her M.S. degree in the Department of Electrical and Computer Engineering at the Ohio State University. Her research interests include Mobility-Assisted Forwarding in wireless sensor networks with a specialization in algorithmic solutions for mobile element scheduling for data harvesting.

Doruk Bozdağ has received his M.S. in Electrical and Computer Engineering from the Ohio State University in 2005 and B.S. in Electrical and Electronic Engineering and B.S. in Physics from Boğaziçi University, Turkey, in 2002. He is currently a graduate student in the Department of Electrical and Computer Engineering at the Ohio State University. His research interests include scheduling algorithms for multiprocessor systems, parallel graph algorithms and high-performance computing.

Robert W. Brewer is an undergraduate student pursuing his B.S. in Electrical and Computer Engineering at the Ohio State University, Columbus, OH. He is also conducting undergraduate research in wireless networks. His current research interests are network security and web based programming.



Eylem Ekici has received his B.S. and M.S. degrees in Computer Engineering from Boğaziçi University, Istanbul, Turkey, in 1997 and 1998, respectively. He received his Ph.D. degree in Electrical and Computer Engineering from Georgia Institute of Technology, Atlanta, GA, in 2002. Currently, he is an assistant professor in the Department of Electrical and Computer Engineering of the Ohio State University, Columbus, OH. His current research interests include wireless sensor networks, vehicular communication systems, next generation wireless systems, and space-based networks, with a focus on routing and medium access control protocols, resource management, and analysis of network architectures and protocols.