Time-Sensitive Ant Colony Optimization to Schedule A Mobile Sink for Data Collection in Wireless Sensor Networks

MURAT KARAKAYA[1]*

Department of Computer Engineering, Atilim University, TURKEY

Received May 2013; In final form 1 June 2013

In Wireless Sensor Networks, sensor nodes are deployed to monitor and record the changes in their surroundings. The collected data in the sensor memories is transferred to a remote central via static or mobile sinks. Because sensors have scarce memory capacity various challenges occur in gathering the data from the environment and transferring them to the remote control. For instance, a sensor's memory might get completely full with the sensed data if the sensor can not transfer them on time. Then, a memory overflow happens which causes all the collected data to be erased to free the memory for future readings. Therefore, when a mobile sink (MS) is employed to collect data from the sensors, the MS has to visit each sensor before any memory overflow takes place. In this paper, we study the design of a mobile sink scheduling algorithm based on the Ant Colony Optimization (ACO) meta-heuristic to address this specific issue. The proposed scheduling algorithm, called Mobile Element Scheduling with Time Sensitive ACO (MES/TSACO), aims to prepare a schedule for a mobile sink to visit sensors such that the number of memory overflow incidents is reduced and the amount of collected data is increased. To test and compare the effectiveness of the MES/TSACO approach, the Minimum Weighted Sum First (MWSF) heuristic is implemented as an alternative solution. The results obtained from the extensive simulation tests

* email: kmkarakaya@atilim.edu.tr

show that the MES/TSACO generates schedules with considerably reduced number of overflow incidents and increased amount of collected data compared to the MWSF heuristic.

Key words: Wireless Sensor Networks, Mobile Sink, Ant Colony Optimization, Scheduling.

1 INTRODUCTION

The main components of a Wireless Sensor Network (WSN) are sensors, base stations (sinks), and a remote central. Sensors monitor a field and record environmental changes. In turn, sinks collect data from sensors to upload to a remote central. Generally, sensors are designed for a single use until they deplete their batteries. Therefore, sensors need to be cheap in order to be deployed in large numbers in a given area. As a result, the main parts of a sensor - computation, sensing, and communication units - have very limited capabilities. These limitations create various challenges in WSN applications. For example, due to limited data memory capacity, sensors cannot store satisfactory large amounts of data for a very long period and, once the memory gets full, they need to free it up either by transferring the collected data to the sink or by deleting them entirely. As losing recorded data is not a desired outcome, gathering data from sensors before their memory overflows is an important requirement for a WSN application.

The data stored in the sensor memory can be collected by a mobile element (ME), or it can be forwarded to a static sink (SS) via a multi-hop wireless connection. ME can be either a mobile sink (MS), which has a direct connection to a remote control, or a mobile collector (MC), which collects the sensory data to carry to a static sink. Using either ME or SS in WSN has its own advantages and disadvantages. For instance, in WSN, where sensors route the sensory data to SS, in multi-hop communication among nodes there could be several issues to be handled by a routing algorithm such as creating and maintaining routing paths to achieve a lower energy consumption, managing data packets traveling on different paths to SS, guaranteeing timely delivery of data, minimizing management overhead in communications, etc. [16]. In general, exploiting sink mobility can extend the network lifetime by decreasing sensor energy spending in communications as shown in [5], [11], [12], [18], among others. However, since ME moves relatively more slowly than the speed of radio signals, data collection is more time-consuming compared to occasions when SS is used. Thus, the ME approach is more suitable for delay-tolerant applications [8].

When an ME is used to collect sensory data, one important issue is to create a schedule for visiting the sensors, such that the generated schedule can satisfy several application-specific performance metrics. For instance, the generated schedule should prevent or minimize the number of memory overflows and maximize the amount of the collected data. This process is referred to as "Mobile Element Scheduling" (MES) [19]. Unfortunately, creating such a schedule has been shown to be NP-complete and several heuristics proposed to prepare an optimal schedule in [10], [19], and [20].

Having been inspired by the behavior of real ants searching around for food, Colorni et al. introduced the Ant Colony Optimization (ACO) to find an optimum solution for the Traveling Salesman Problem (TSP) in [2], [3], and [7]. While ants wander for food, they leave specific pheromone on their path according to the quality of the located food. Thus, other ants are able to follow the same path to the food source by detecting the pheromone while posing their own pheromone on the way as well. As a result, the more commonly used paths tend to bear a higher level of pheromone. The authors exploited this mechanism by simulating artificial ants traversing the problem space, finding some solutions, and marking the usefulness of the path according to the quality of the solution. Thanks to the positive feedback mechanism, the ACO can converge to an optimal solution. In addition, it has been widely used to solve a number of combinatorial optimization problems while being compared with other heuristics [4], [6], [13] among the others.

In this work, we apply the ACO meta-heuristic to create schedules for the MES problem with better performance results. In our implementation, artificial ants simulate MS to construct a visiting schedule for sensors. Created by an ant, each schedule is evaluated according to its success, which is defined by the number of sensor memory overflows and the amount of the collected data. Then, according to the success of the produced schedule, ants lay down pheromones on the edges between the visited sensors. After a designated number of rounds, it is expected that the ants be able to find an optimum schedule.

The paper is organized as follows: the related work is presented in Section 2. Section 3 provides the details of the WSN model and the proposed MES algorithm. The simulation model, along with the related tests, are discussed in Section 4. Finally, in Section 5, the conclusion and future work are presented.

2 RELATED WORK

Somasundara et al. introduced the Mobile Element Scheduling (MES) problem in [19]. After proving that the MES problem is NP-complete, the authors proposed two heuristics with their variations: Earliest Deadline First (EDF) and Minimum Weighted Sum First (MWSF). In EDF, MS visits the node with the closest deadline first. As the authors express the EDF heuristic has an explicit pitfall since it takes into account only the deadlines but not the time to get to the sensor node. Therefore, the EDF heuristic could potentially produce some inefficient schedules. To improve this heuristic, the MWSF heuristic is designed such that the user can assign two weights, one to the sensor deadline and the other to the time to get to sensor node, in order to calculate a score for each node. The weight (α) is set a number between 0 and 1 for the deadline, and 1-α for the travel time. According to the experiment results, the best performance outcomes are obtained when α is around 0.1. The authors observed that the MWSF solution constructs schedules, which perform better than the other proposed heuristics. One drawback of the MWSF heuristic is that MS has to move back and forth frequently between farthest nodes. In a follow-up work, the authors extended their work using multiple MSs [20].

In [10], Gu et al. approached the MES problem with a two-phase solution to reduce the back-and-forth movement behavior. In the first phase, all sensors are initially partitioned into groups, called bins, with respect to their similarity in deadlines. Later, the sensors in each bin are further divided into sub-bins according to their geographical locations. In the second phase, after preparing a minimum cost schedule for each sub-bin using a Traveling Salesman Problem heuristic, all these schedules are concatenated into an overall schedule.

There are some other works to solve different versions of the MES problem, e.g. [1], [13] and [14]. In [1], Almi'ani et al. worked on a version of the MES problem in which the sensory data needs to be delivered to a static sink by a mobile collector before a given time expires. There are multiple mobile collectors scheduled to visit each sensor once during a tour. Each tour of the mobile collectors commences and ends at the same static sink. The authors aimed to minimize the total length of the traveling time of all the tours providing that only one mobile collector visit any node, and that all delivery deadlines be satisfied.

In [13] and [14], sensors are assumed to have unlimited data memory size and, thus, memory overflows are disregarded. Furthermore, it is assumed that these sensors can communicate with the MS by sending data via multiple hops along the shortest path. The sensors which are a single-hop away from the MS can function as relay nodes for distant sensors, which can send their data to these relay sensors by using intermediate sensors in the multi-hop communication. In this application scenario, to maximize the amount of the collected data with minimum power the authors try to optimize the number of relay sensors and their members. Since more members imply more message traffic to a relay sensor which consumes more power from the relay sensor, the authors formulated the problem as selecting relay sensors with an optimal number of members. An important factor to decide on the relay sensor is the residual energy of candidate sensors. As a solution for this version of the MES problem, Li and Xiao, proposed to use the Optimal Traversal Path Taboo Search Algorithm (OTP-TS) [14], whereas Kumar and Thomas designed a solution based on the ACO [13]. In both works, the main performance metrics to improve are the energy efficiency for data gathering and the total amount of the data collected, whereas, in this work, we aim to prevent memory overflows along with the increased amount of the collected data. As a result, the present work differs from Li and Xiao's and Kumar and Thomas's in terms of goals and assumptions.

3 MOBILE ELEMENT SCHEDULING WITH TIME-SENSITIVE ANT COLONY OPTIMIZATION

The *Mobile Element Scheduling with Time Sensitive ACO* (MES/TSACO) has a two-fold objective: collecting data from sensors before the allocated deadlines, and collecting the maximum amount of sensor data. Below, the WSN model and the MES problem are first presented, and then the implementation details of the proposed solution are provided.

3.1 Problem Definition

It is assumed that a WSN has been deployed for monitoring some environmental changes, for instance heat, light, or mobility. WSN has three important components: sensor nodes (SN), a mobile sink (MS), and a remote central (RC). We assume that the SN and MS locations, sensor sampling rates, sensor memory capacities and their current states are known to, or can be computed by, the RC. Thus, RC is able to implement a centralized solution.

Each SN monitors the environment with a fixed sampling rate, stores these readings in a limited memory, and transfers the recorded data to the MS whenever the MS contacts with it. If any SN cannot upload the data to the MS and the memory gets completely full, SN purges all the data and frees the memory, which is called *a memory overflow*. MS moves freely with a fixed velocity on the monitored field to collect data from SN via direct (one-hop) communication according to *a schedule* determined by RC. While transferring the collected data to the RC, MS navigates to the next sensor according to the schedule. The constructed schedule for visiting sensors is expected to lead to a minimal number of memory overflows and to a maximal amount of collected data for a given period of time, namely *tour time*.

Provided below are the details of the MES/TSACO implementation

3.2 The Time-Sensitive ACO

A combinatorial optimization problem can be static or dynamic with respect to the given characteristics of the problem. In static problems, the underlying system properties stay the same throughout the problem-solution process. A typical example of these kinds of problems is the Traveling Salesman Problem (TSP). In the TSP, there are a number of towns (vertices) connected to each other by some arcs (edges). Each edge is associated with a cost (distance). In essence, the solution to a TSP is to construct a minimum distance circuit passing through each vertex once and only once. Therefore, the cost of a solution depends on the distances among the towns. These problem characteristics -town locations and distances- do not change during the development of a solution. On the other hand, in dynamic problems, problem characteristics can be changed over time as a solution is being generated.

The MES problem is an example of dynamic combinatorial optimization problems. The aim is to visit all, or the maximum number, of nodes before dynamically changing deadlines expire. The generated schedule is defined in terms of sequences of sensors as in Fig. 1. Since the duration of a MS tour time is limited, a solution will constitute finite lengths of states. Then, a set of candidate solutions can be generated by the permutation of sensors. However, a set of feasible solutions are unlimited due to the MES problem definition: each ant does not have to visit all the sensors; any ant can visit the same sensor more than once; and all the nodes are reachable from any other.

In the TSP, the costs between cities are defined as the distances, which are fixed. However, the costs between sensor nodes in the MES are based on the possible number of overflows, which are dependent on the MS visit time. For example when ant k arrives at sensor A at the time t_0, the least cost can be between A and B as in the schedule T^k. However, after some time when ant k arrives again at sensor A at the time t_1, now the costs between nodes can change due to the elapsed time. Furthermore, sensor memories can store different amounts of data according to their different sample rates and

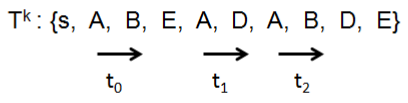


FIGURE 1
A sample schedule showing the dynamic nature of the MES.

capacities, which define various memory overflow periods. Thus, ant k might select sensor D as it offers the least cost at time t_1.

To adapt the dynamic nature of the MES problem, the ACO meta-heuristic implementation of the pheromone storing and the heuristic calculation is modified as explained below. In a static problem, the ACO meta-heuristic stores pheromones to give weight to directed edges from a current node to the others in order to select the next possible move. The weight is calculated according to the quality of the previous solution found by an ant. As the problem is static and the nodes appear on the list only once, their order is sufficient to relate the success of the previous selections with the future selections. As discussed above, the MES problem has a different setting. Therefore, the author introduced the time factor for storing pheromone values and selecting the next node by labeling pheromones and edges with a time tag as in Fig. 2. Thus, the pheromone mechanism becomes Time-Sensitive and can record once an ant decides to move from node A to node B.

Likewise, the heuristic calculation method is also modified. In the general ACO implementation, calculation of heuristic values between nodes is static, that is, calculation is done at the beginning of the problem and stays the same during the solution process. However, in our problem, the heuristic value calculation is dynamic such that any sensor's time remained to overflow is taken into consideration while setting heuristic values dynamically. Therefore, when an ant refers to some heuristic value between two nodes it has to be calculated at each time with the current information. The details about the time-sensitive pheromone and heuristic mechanisms are provided in Section 3.3.

3.3 The MES/TSACO implementation details

In this implementation of the ACO meta-heuristic, an individual artificial ant simulates an MS, and its schedule is constructed by incrementally selecting the next sensor to visit until the designated tour period expires. At the beginning, each ant is located at the start point s in the monitored field, and the cost of the tour is zero. The ant selects the next sensor to visit from the list of candidate sensors according to the current cost. It does not have to return to the start and throughout its tour, the total overflow number and the amount of data collected are computed as objective function values. Whenever a tour is completed, a pheromone value is computed according to the cost of the tour. Next, these values between the visited sensors are updated with the computed value labeled with the visit time. Then, ants start their second tour from the same initial location, and this cycle continues until all the artificial ants complete their predetermined number of rounds. Finally, ACO outputs the best schedule found so far.

From the above explanation, it is clear that the MES/TSACO requires the information about the current status of sensor memories in WSN to implement the proposed centralized solution. Thus, one drawback of the centralized solution would be the requirement of communication among sensors and the RC. However, the amount of communications can be reduced to a minimum level if RC can keep track of sensors memory usage according to the sampling rate. Furthermore, MS can help to sync RC information about sensors by collecting current sensor memory capacities and uploading them to RC as MS goes by the sensors.

Below provided is a formal characterization of the implementation of the ACO meta-heuristic to find a minimum-cost feasible solution for the MES problem.

Problem Representation

Assume that a wireless sensor network be defined by a set of sensor nodes, a set of edges (L) between all pair nodes (i, j) with distance information (d_{ij}), and the objective function (f) be inversely based on according to the cost (c) between sensor nodes which is dynamically associated with the overflow numbers as described in Section 3.3. Then, the Mobile Element Scheduling problem (MES) is the problem of finding a minimum-cost schedule to visits sensor nodes in the network for a designated tour time. Formally, the MES problem is a maximization problem (S, f, Ω), where S is the set of candidate solutions, f is the objective function, and Ω is a set of constraints. The objective function f assigns a cost value $c(s, t)$ to each candidate solution $s \in S$ considering the constraints $\Omega(t)$. The parameter t indicates that the objective function, cost value, and the constraints are dynamic and time-dependent. The ACO meta-heuristic is applied to find a globally optimal feasible solution $s*$ to this maximization problem.

Construction Graph

The construction graph is the graph G_C = (C,L), where C corresponds to the set of sensor nodes, and L is the set of connections which fully connect to G_C.

Constraints

The only constraint is that ants use connections $l_{i,j} \in L$ to visit the sensor nodes within the specified tour time.

Pheromone trails and heuristic information

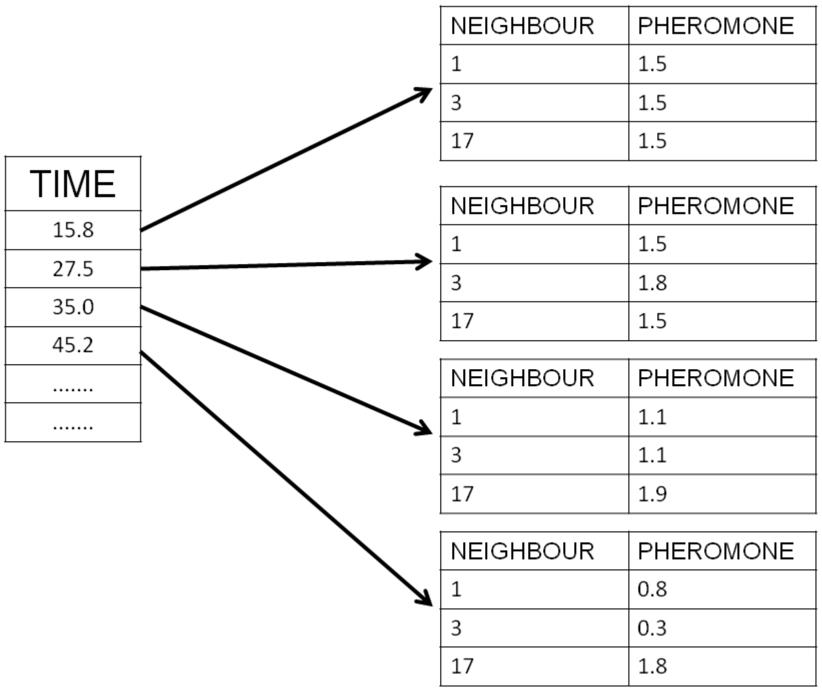
The pheromone values between two sensors serves as a long-term memory for the previous solutions of the ants. Thus, the entire search attempts by all the ants so far contribute to the pheromone values. The MES problem involves finding a set of minimum cost path problems for a limited tour time. Since each connection $l_{i,j} \in L$ can have many different pheromone trails associated with it according to an ant's visit time, each connection $l_{i,j}$ is associated with only one pheromone trail value $\tau_{i,j,t}$ for each ant visit on time t for sensor node i. As seen in Fig. 2, *a Time-Sensitive pheromone data structure* is used to store the pheromone value of each sensor node for the outgoing edges based on any ant's visit time to the sensor.

In the ACO approach, contrary to pheromone value, the heuristic value provides apriori information regarding the problem case to the ants and each edge is also assigned a heuristic value η_{ij}. Likewise pheromone values, we have also modified the heuristic formula by considering dynamic nature of the problem using the MWSF heuristic suggested in [19]. The heuristic value η_{ij} of an edge from sensor i to sensor j is set as:

$$\eta_{i,j} = \frac{1}{TtO_j * d_{ij}} \tag{1}$$

where TtO_j is the *remaining time to overflow* for the sensor j, and d_{ij} is the *distance* between sensor i to sensor j. As a result, the heuristic value is dynamic and changes in time as the sensor memory gets full with sensed data. This formulation assigns higher heuristic values to the sensors, which have less time to overflow and which are nearer to the current sensor i. ⋆

⋆ In fact, in simulation tests we have experimented with other heuristics such as prioritizing only the nearer sensors (nearest first) or only the less time to overflow sensors (deadline first). However, these heuristics performed far worse than the one in Formula 1.



TIME
15.8
27.5
35.0
45.2
....
....

NEIGHBOUR	PHEROMONE
1	1.5
3	1.5
17	1.5

NEIGHBOUR	PHEROMONE
1	1.5
3	1.8
17	1.5

NEIGHBOUR	PHEROMONE
1	1.1
3	1.1
17	1.9

NEIGHBOUR	PHEROMONE
1	0.8
3	0.3
17	1.8

FIGURE 2
Each sensor node stores pheromone values for the outgoing edges according to the ant visit time.

Solution Construction

A finite set of sensors are given, where NS is the number of all sensors. Each ant begins its tour from the same source node s. During a predefined tour duration d, it moves from one sensor node to the next until the tour duration completes.

When ant k is located at node i, it chooses the next node j using a probabilistic decision rule which is a function of the visit time t, the local pheromones, and the heuristic information as in Formula 2 [7].

$$p^k_{i,j,t} = \frac{[\tau_{i,j,t}]^{\alpha} [\eta_{i,j}]^{\beta}}{\sum_{l \in N^k_i} [\tau_{i,l,t}]^{\alpha} [\eta_{i,l}]^{\beta}}, \text{ if } j \in N^k_i \tag{2}$$

where α and β are two weight parameters, which affect the relative influence of the pheromone trail and the heuristic information. N^k_i is the set of candidate sensors to visit next once ant k arrives at the sensor i at the time t. N^k_i does not contain either the sensors which are not accessible from the sensor i before their overflow time, or the sensor i itself. The probability of choosing all those excluded sensors is assigned as 0.

For the pheromone values, if it is the first time an ant is visiting sensor i at time t, the pheromone values for all the edges sensor i to its neighbors in N^k_i will be initialized to τ_{init} as in Formula 3. As seen in Figure 2, the pheromone values of all the outgoing edges from the sensor i are set an initial value when an ant arrives at it when $t = 15.8$. Once an ant completes its tour, it updates the pheromone values of the visited sensor according to the visit time t. For example, in the same figure, the sensor i has different pheromone values for its neighbors when an ant visited it when $t = 27.5$.

$$\tau_{i,j,t} \leftarrow \tau_{init}, \forall (i,j) \in L \tag{3}$$

After deciding the probability of each neighbor, their probabilities are normalized such that the sum of all the probabilities is 1. Then, we generate a random number between 0 and 1 to select the next sensor.

According to Formula 2, the probability of choosing a particular edge (i,j) increases with the value of the associated pheromone trail and of the heuristic information value.

Each ant k maintains a memory M^k to hold the amount of the collected data and the number of overflows occurred. This memory is used to compute the success of the tour (schedule) T^k and the amount of pheromone to deposit on the path.

Updating Pheromone Trails
After all the ants have constructed their tours, the pheromone trails are updated. This is done by first decreasing the pheromone value on all the edges using a constant factor ρ, that is pheromone evaporation, implemented by

$$\tau_{i,j,t} \leftarrow (1-\rho)\tau_{i,j,t}, \forall(i,j) \in L \tag{4}$$

where $0 < \rho \leq 1$ is the pheromone evaporation rate.

The cost c of a generated solution is defined as the number of overflows (ON) occurred during an ant's tour as follows.

$$c = ON \tag{5}$$

The objective function f is based on the cost c as below:

$$f = \frac{1}{c+1} \tag{6}$$

Thus, we would like to maximize the objective function f by minimizing the cost c.

Using the cost of the generated tour, the amount of the additional pheromone is calculated and added to the edges which the ants have visited during the tour. The amount of pheromone is set according to the below formula:

$$\tau_{i,j,t} \leftarrow \frac{1}{c+1} + \tau_{i,j,t}, \forall(i,j) \in T^k \tag{7}$$

where T^k is the tour traversed by the ant k. As a result of Formula 7, tours with less cost will cause more pheromone additions on their edges as a positive feedback.

Regarding the cost function values of the tours, we select the minimum cost tour for the current round and compare it with the one found so far to decide the best tour (T_{Best}). Then ants begin a new round.

Selecting the best solution
After all the ants finish a predefined number of rounds, T_{Best} is output as the MS visit schedule to the WSN simulator to test its accuracy and compare the performance results.

4 SIMULATION MODEL AND RESULTS

This section presents the evaluation of MES/TSACO with respect to several performance metrics and the other scheduling heuristic. Below we first discuss the simulation model by explaining model parameters in details. Then, performance metrics are defined to compare the the success of the proposed algorithm. Next, we explain how to decide the default values of the ACO parameter values using simulation tests. Finally, the results of various experiments are reported.

4.1 Simulation Model

For simulation experiments, we implemented the TSACO and WSN using MASON discrete-event multi-agent simulation library [15]. For a given experiment setting, first, the TSACO heuristic is run to find an optimum schedule. Then, the WSN simulator is activated to apply the optimum schedule and to observe the performance results.

Table 1 summarizes important parameters of the simulation model and their default values. Below, important parameters are defined along with the default values.

WSN: It is assumed that the monitored field size is 500x500 meters, and that the sensors are deployed in a grid topology to cover the monitored field. MS is initially located at a random node on the field.

Sensors: The number of sensors is 625 as default. Each sensor has 4KB of data memory and a sensing rate of 1B/sec. At the beginning of simulation each sensor memory is filled with a random amount of data.

Simulation: In each simulation, MS runs for 20 minutes to collect data. To find the average performance results, each experiment is executed 40 times.

MS: MS can move freely in the monitored field to visit any sensor according to the given schedule with a fixed speed at 40km/h. MS approaches each sensor to collect data - that is, at a time MS can get connected with only one sensor. It is assumed that the data transfer time from any sensor to MS is negligible.

4.2 Performance Metrics

Ideally, MS should collect data from all sensors before any sensor memory overflows. However, this cannot be always accomplished due to different parameters, such as memory size, sensing rate, number of sensors, distance among sensors, MS speed, etc. Also, the scheduling algorithm should be able to perform with minimal overflows. Apart from this, during a tour, MS should collect as much data as possible. Motivated by these observations, the following performance metrics are defined.

- *Overflow Number (ON)*: Number of overflow incidents occurred during the MS tour.

Parameters	Definition	Default Setting
W	Width of monitored field	500m
H	Height of monitored field	500m
N	Number of sensors	625
ST	Sensor Topology	Grid
TT	Tour Time	1200s
ML	MS initial location	Random
DR	Sensing rate	1B/s
MC	Memory capacity	4KB
IM	Initial memory capacity	Random
SP	MS speed	40km/h

TABLE 1
The parameters and default values for WSN simulator.

- *Collected Data (CD)*: Total amount of the data collected from sensors and uploaded to SS by MS at the end of the tour.

Thus, to evaluate the success of our algorithm, MS/TSACO, we implemented another MS scheduling algorithm, the MWSF, introduced in [19]. For both approaches, all parameter values are identical in the respective simulation experiments. To compare the results of the heuristics, we opt to present the performance improvement in percentages for the number of overflows and the collected data metrics as in Formulas 8 and 9, respectively.

$$Improvement_{ON} = \frac{ON_{MWSF} - ON_{TSACO}}{ON_{MWSF}} * 100 \qquad (8)$$

$$Improvement_{CD} = \frac{CD_{TSACO} - CD_{MWSF}}{CD_{MWSF}} * 100 \qquad (9)$$

4.3 MES/TSACO Parameter settings

ACO is known to be sensitive to parameter settings to find an optimum solution for a given optimization problem, e.g. [2], [3], [4], [6], [9] and [17]. Therefore, we first conduct a set of experiments by varying the initial values of the important parameters, namely number of ants, number of ant tours, and alpha and beta, to observe their effects on the overall success of the proposed scheduling algorithm. After observing the effects of the varying values of each parameter, the default values of the MES/TSACO to be used in the comparisons in Sections 4.5, 4.6, and 4.7 are determined.

The initial values of the ACO parameters, given in Table 2, are selected similar to those used in related studies such as [10], [16], [19], [20]. The simulation parameter values given in Table 1 and the MES/TSACO parameters' initial values in Table 2 are effective in the following experiments whose results are presented in the Figures 3, 4, 5, and 6 . In these figures, the error bar indicates 1 standard deviation (σ) from the mean.

Parameters	Definition	Initial Settings
m	Number of ants	20
NR	Number of rounds	10
α	Relative importance of pheromone	3.0
β	Relative importance of heuristic	5.0
τ_{init}	Initial level of pheromone	2.0
ρ	Evaporation rate	0.01

TABLE 2
The parameters and their initial values for the MES/TSACO heuristic.

Number of ants

For the simulation settings given in Table 1 and 2, different ant populations are tested. As seen in Fig. 3, as the ant number increases, the quality of the solution found improves. Using 45 ants instead of 5 improves the overflow number about 15%. However, upon reaching a certain level, increasing the ant number does not affect the result more. Therefore, employing 30 ants seems to be sufficient to reach a better optimal solution with the current settings.

Number of rounds

The number of rounds that each ant should execute is another important parameter. As the ants execute more iterations, they may make better use of the pheromone trails to search for higher quality solutions. This expectation is supported by the experiments as shown in Fig.4. Higher number of rounds produces higher quality solutions. For example, when 40 iterations are applied, the number of overflow incidents decreases more than 10% compared to the case when 5 iterations are used. However, increasing the number of rounds results in more computation time. Therefore, we choose 30 iterations to be used as the default value rather than 50 since the overflow numbers are very close to each other.

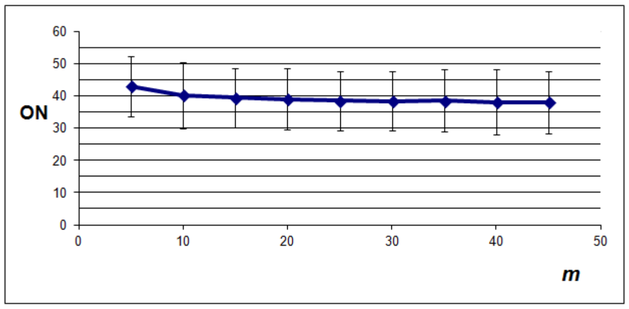


FIGURE 3
Impact of number of ants on overflow.

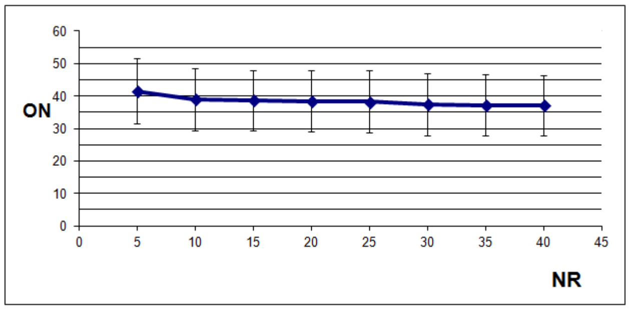


FIGURE 4
Impact of number of rounds on overflows.

α and β values

For α and β values, it is observed that the heuristic parameter (β) has much more influence than the pheromone parameter (α) for locating better solutions as seen in Fig. 5 and 6. This observation is in line with the conclusions provided by the previous works. For example, Colorni et.al. examined the α parameter best value for different combinatorial problems and proposed to use values between 1 and 1.5 as an optimal range in their original works such as [2], [3], and [4]. Similarly, in [17], it is concluded that the α parameter value does not have a significant effect on the quality of the generated solution. Therefore, the common value of α parameter is fixed and set 1 in many works, such as [2], [16], etc.. Moreover, in some works, e.g. [9], α parameter is removed since setting it to 1 makes effectively the parameter redundant. Considering the results of above experiment and the previous works, 1 is chosen for the α parameter as the default value.

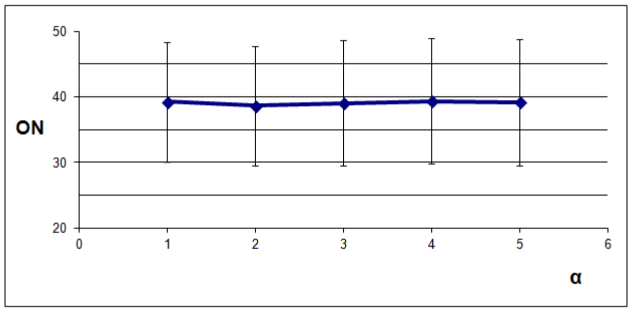


FIGURE 5
Impact of α value on overflows.

Contrary to α parameter, the heuristic parameter (β) has a significant effect on the success of the MES/TSACO algorithm. Higher values of β produce better results. However, since β is an exponential parameter of the real value heuristic, larger β values increase the computation cost. Since the gain of the performance results of 10 and 7 is very similar, 7 is opted for as the β default value.

Determining the default values of the MES/TSACO heuristic parameters

After observing the effect of the important parameters on the success of the proposed algorithm with the initial settings, we attempt to combine a best

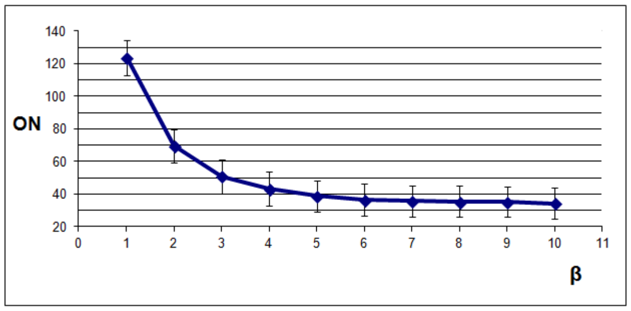


FIGURE 6
Impact of β value on overflows.

combination of the parameter values to find better results. After conducting a set of experiments, the parameter values given in Table 3 are chosen as the default values to be used in the following performance tests.

Parameters	Definition	Default Settings
m	Number of ants	30
NR	Number of rounds	30
α	Relative importance of pheromone	1.0
β	Relative importance of heuristic	7.0
τ_{init}	Initial level of pheromone	2.0
ρ	Evaporation rate	0.01

TABLE 3
The parameters and their default values for the MES/TSACO heuristic.

4.4 MWSF Parameter setting

The MWSF heuristic has a single parameter which is used to give relative weights for the sensor deadline and the travel time to the sensor to calculate a score and prioritize the next visiting node. As the MWSF heuristic is reported to perform the best when the balancing parameter is set 0.1 in [19], in this work, the MWSF heuristic is run with the same parameter value as well.

4.5 Base Experiment Results

After working on the impact of TSACO parameters on the quality of the produced solution, various experiments were conducted to compare the MES/TSACO heuristic with the MWSF one. The obtained results are presented as mean values (μ) of the 30 time-run simulations along with the standard deviations (σ) in Tables 4 and 5 when the default values of the simulation parameters in Table 1 and TSACO parameters in Table 3 are effective.

As seen in Table 4, the MES/TSACO heuristic causes fewer number of memory overflows compared to the MWSF heuristic for a varying number of sensors deployed on a fixed size field. However, the improvement in overflows changes with the number of sensors deployed inversely. As the deployed sensor number increases, the number of possible schedules increases exponentially. Thus, the MES/TSACO heuristic can explore only a small portion of them and has a lower chance to converge to an optimal one. On the other hand, for the current simulation setting, the monitored field has an area of 500x500 meters. To create a uniformly distributed grid of 900 sensors, each sensor should be deployed from other neighboring sensors about 17 meters away. It is assumed that the communication range between the MS and sensors would be longer than this distance in actual applications. Therefore, in a denser deployment of sensors with a longer range of communication, actually MS is in fact not to visit each sensor by getting so close to its location. As an option it can group sensors, which can be covered by the same communication range at a single stop. As a result, it is anticipated that the MES/TSACO heuristic would considerably result in fewer incidents of overflows compared to the MWSF heuristic in practice.

	TSACO		MWSF		
N	μ	σ	μ	σ	$Improvement_{ON}$
900	93.23	14.64	109.33	12.4	15%
625	32.13	8.50	47.83	10.41	33%
400	6.76	2.48	11.23	3.86	40%
225	2.43	1.25	3.03	1.69	20%
100	0.96	0.88	1.06	0.98	9%

TABLE 4
Average Number (μ) of Overflows caused by TSACO and MWSF heuristics for different numbers of sensors (N).

Table 5 summarizes the results for the second performance metric. By collecting data from the sensors before being purged due to an overflow, the MES/TSACO heuristic is able to gather more data. Aside from this anticipated advantage, even with lower overflow numbers, the schedule created by the MES/TSACO heuristic results in higher amounts of collected data. For example, for the 225 sensor deployment experiments, the MES/TSACO heuristic can prevent about 1 overflow out of 3 caused by the MWSF heuristic (see Table 4) . Since a sensor data memory is assumed to be 4KB in the simulation setting, if the increase in the amount of data collected according to the MES/TSACO schedule is due only to the prevented overflows, this increase is expected to be around 4KB. However, as seen in Table 5, the difference between the amount of data collected by these two heuristics is more than 130KB. Therefore, the MES/TSACO heuristic does not only reduce the number of overflows, but also increases the amount of the collected data considerably.

	TSACO		MWSF		
N	μ	σ	μ	σ	$Improvement_{CD}$
900	721.52	24.31	646.05	24.42	12%
625	688.65	33.05	569.08	19.80	21%
400	603.29	38.89	471.35	20.22	28%
225	487.53	23.06	356.81	15.04	37%
100	266.49	12.44	234.72	10.54	14%

TABLE 5
Average (μ) amount of data in KB collected by TSACO and MWSF heuristics for different number of sensors (N).

As a result of the base experiments it can be argued that the MES/TSACO heuristic is capable of producing schedules with fewer overflows and more collected data compared to the other heuristic. To validate this argument, we check the statistical significance of the differences between the produced results by the two algorithms applying the paired-samples T test. The test shows that the results produced by the MES/TSACO are significantly better than the ones generated by the MWSF. For example, for the default settings of the simulation and the ACO parameters, the obtained 30 sample pairs of both algorithms are used to calculate the value of t as the significance level is set 0.01 and two-tailed hypothesis is opted out. The value of t is calculated as 18.300693 and the power value (P) is found less than 0.00001. That is, the difference between the results generated by the heuristics is statistically significant.

Presented below are the results of the experiments carried out to observe the effects of the other simulation parameters on the performance metrics.

4.6 Experiment Results For Various Memory Capacities

For different values of memory size, the number of overflow incidents and the amount of collected data are observed for both heuristics. In Fig. 7 the improvements realized by the MES/TSACO over the results of the MWSF heuristic are depicted.

Generally speaking, when the sensors have more memory with the same sensing rate, MS has more time to visit sensors. The advantage of having more time to collect data can be viewed in a drop of number of overflows and in a surge of collected data for both scheduling algorithms. Similarly, once the memory capacity is reduced and the sensing rate is maintained, the sensors have less time to get overflowed which decreases the chances of MS to catch up with all overflow incidents.

In the experiments this expected trend for both heuristics are observed as seen in Tables 6 and 7. In Tables 6, for the incremented values of MC, both heuristics cause less memory overflows. For the amount of collected data, Table 7 confirms that an increase in MC ends up with more collected data.

When two heuristics are compared, it can be observed that the MES/TSACO exploits the increase in the memory capacity better than the MWSF heuristic as seen in Fig. 7. The improvements generated by the MES/TSACO on the collected data are almost in linear with the increase of the memory capacity. However, the improvement in the number of overflow incidents does not follow a linear pattern. As the memory increased up to 5 KB, the MES/TSACO heuristic success of minimizing the number of overflows compared to the MWSF heuristic is increasing. But, then, this improvement begins to decrease. It can be argued that, after some memory capacity, in our case 5 KB, as the time that an overflow takes place is so long that very small number of sensors face the overflow incident, and thus there is not much room to optimize the solution. As an extreme case, when the sensors have unlimited memory capacity any method would come up with a zero overflow number. Therefore, the pattern in Fig. 7 depicts this phenomena.

4.7 Experiment Results For Various Tour Durations

MS is assumed to start from an initial location in the monitored field to collect data from sensors until a designated tour time elapsed. Thus, the heuristics

	TSACO		MWSF		
MC (KB)	μ	σ	μ	σ	$Improvement_{ON}$
1	509.10	9.57	510.40	11.79	0%
2	181.06	9.24	188.73	9.41	4%
3	80.83	12.45	93.86	9.02	14%
4	32.13	8.5	47.83	10.41	33%
5	15.36	5.89	25.36	7.88	39%
6	8.53	3.85	13.7	5.55	38%
7	5.53	2.93	7.83	3.80	29%
8	3.93	2.28	5.46	2.95	28%
9	3.2	2.02	4.33	2.53	26%
10	2.7	1.76	3.46	1.94	22%

TABLE 6
Average (μ) Number of Overflows caused by TSACO and MWSF heuristics for different sensor memory capacities (MC).

	TSACO		MWSF		
MC (KB)	μ	σ	μ	σ	$Improvement_{CD}$
1	226.48	6.81	225.30	6.21	0%
2	375.67	10.76	360.23	13.41	4%
3	523.48	17.60	472.81	12.96	11%
4	688.65	33.05	569.08	19.80	21%
5	841.32	60.09	653.49	30.84	29%
6	1012.75	83.49	728.14	30.88	39%
7	1167.68	96.80	799.32	32.17	46%
8	1342.95	104.73	871.37	32.06	54%
9	1551.36	121.33	942.48	32.42	65%
10	1762.95	137.72	1004.55	33.01	75%

TABLE 7
Average (μ) amount of data in KB collected by TSACO and MWSF heuristics for different sensor memory capacities (MC).

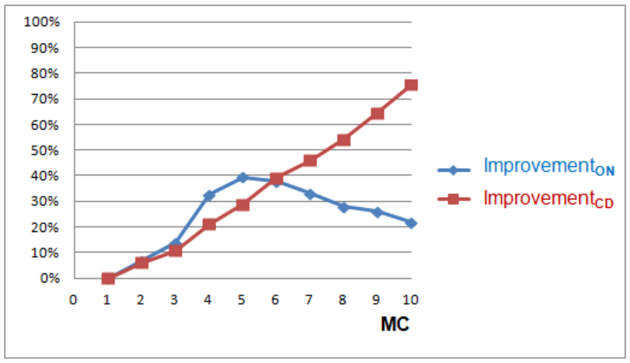


FIGURE 7
Impact of memory capacity (MC) on the MES/TSACO improvements for overflow and collected data metrics.

aim to create a schedule to visit sensors for the specified tour time. A robust scheduling algorithm should produce similar quality solutions for different tour durations. As the tour time extends, the number of overflows and the amount of collected data are expected to be increased. The experiment results given in Tables 8 and 9 apparently support the expectation.

	TSACO		MWSF		
TT (Sec.)	μ	σ	μ	σ	$Improvement_{ON}$
600	22.63	5.48	28.53	6.15	21%
900	28.43	7.22	38.3	8.44	26%
1200	32.13	8.5	47.83	10.41	33%
1500	38.26	10.46	59.16	10.79	35%
1800	41.73	11.70	68.03	9.68	39%

TABLE 8
Average (μ) Number of Overflows caused by TSACO and MWSF heuristics for different tour times (TT).

When the success of two heuristics are compared, it is observed that for different tour durations the MES/TSACO always generates better results than the MWSF heuristic as seen in Fig 8. In general, compared to the alternative

	TSACO		MWSF		
TT (Sec.)	μ	σ	μ	σ	$Improvement_{CD}$
600	334.89	21.10	285.73	15.91	17%
900	508.43	27.38	427.82	16.52	19%
1200	688.65	33.05	569.08	19.80	21%
1500	847.99	41.12	714.06	19.41	19%
1800	1014.65	60.55	854.50	20.35	19%

TABLE 9
Average (μ) amount of data in KB collected by TSACO and MWSF heuristics for different tour times (TT).

heuristic, the MES/TSACO generates schedules causing less overflow numbers ranging from 20% to 40%. Similarly, the amount of data collected by the MES/TSACO schedules is 20% more than that of the MWSF schedules for the experimented tour times.

One interesting observation is the fact that as the tour duration gets longer, the MES/TSACO produces better overflow numbers with respect to the MWSF. The reason for this could be that for a longer tour duration there are more chances for overflow incidents to happen. Therefore, advantages of using a more efficient scheduling algorithm can be observed easily. For example, for a 30-minute tour, the MES/TSACO causes %39 less overflow incidents compared to the MWSF. Consequently, it can be argued that the MES/TSACO is also capable of successfully scheduling longer tours as well.

5 CONCLUSIONS

This paper introduced a scheduling algorithm based on the Ant Colony Optimization (ACO) meta-heuristic for gathering data in Wireless Sensor Networks (WSN) with a mobile sink (MS) following a controlled mobility pattern. In WSN, sensors with a limited memory capacity can store the sensed data to transfer MS for a specified amount of time. MS is tasked to collect data from the sensors upon visiting them. However, if MS is not able to visit all sensors in time, memory overflows can occur and all the stored data can be lost to free the memory.

The proposed mobile element scheduling algorithm (MES), the Time-Sensitive ACO for Mobile Element Scheduling (MES/TSACO), is based on the ACO

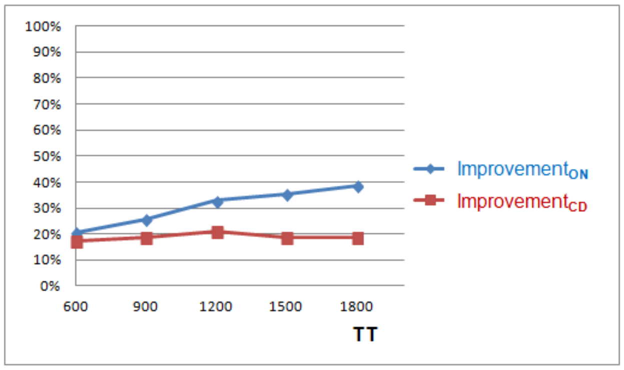


FIGURE 8
Impact of tour time (TT) on overflow and collected data metrics.

meta-heuristic. The MES/TSACO aims at minimizing the number of overflows and increasing the size of the collected data by making use of controlled sink mobility efficiently and effectively. For this reason, the ACO pheromone and heuristic mechanisms are modified to reflect the dynamic nature, and to meet the requirements, of the MES problem.

The MES/TSACO and a WSN were implemented in a simulation environment, and to compare the performance results, the MWSF heuristic was also simulated. Extensive simulation tests were conducted with different simulation parameter values. It is observed that compared to the results produced by the MWSF heuristic, the TSACO heuristic creates MS schedules which considerably reduce overflow incidents while increasing the amount of the collected data for various WSN settings such as sensor numbers, memory capacities and tour times.

As a future work, the author intends to extend this work by scheduling a minimum number of multiple MSs, such that there will be no incidents of overflow at all. Furthermore, it is planed that the scheduling algorithm be adapted so as to control the MS speed in order to minimize the overflows with optimum MS energy consumption.

REFERENCES

[1] Khaled Almi'ani, Anastasios Viglas, and Lavy Libman. (2010). Mobile element path planning for time-constrained data gathering in wireless sensor networks. In *Advanced In-*

formation Networking and Applications (AINA), 2010 24th IEEE International Conference on, pages 843–850. IEEE.

[2] Alberto Colorni, Marco Dorigo, Vittorio Maniezzo, *et al.* (1991). Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life*, volume 142, pages 134–142. Paris, France.

[3] Alberto Colorni, Marco Dorigo, Vittorio Maniezzo, *et al.* (1992). An investigation of some properties of an ant algorithm. In *Proceedings of the Parallel Problem Solving from Nature Conference (PPSN 92)*, pages 509–520. Elsevier Publishing.

[4] Alberto Colorni, Marco Dorigo, Vittorio Maniezzo, and Marco Trubian. (1994). Ant system for job-shop scheduling. *Belgian Journal of Operations Research, Statistics and Computer Science*, 34(1):39–53.

[5] Mario Di Francesco, Sajal K. Das, and Giuseppe Anastasi. (August 2011). Data collection in wireless sensor networks with mobile elements: A survey. *ACM Trans. Sen. Netw.*, 8(1):7:1–7:31.

[6] Qiulei Ding, Xiangpei Hu, Lijun Sun, and Yunzeng Wang. (2012). An improved ant colony optimization and its application to vehicle routing problem with time windows. *Neurocomputing*, 98(0):101 – 107. Bio-inspired computing and applications (LSMS-ICSEE ' 2010).

[7] Marco Dorigo and Thomas Stutzle. (2004). *Ant Colony Optimization*. MIT Press.

[8] RJ DSouza and Johny Jose. (2010). Routing approaches in delay tolerant networks: A survey. *International Journal of Computer Applications*, 1(17):8–14.

[9] Dorian Gaertner and Keith Clark. (2005). On optimal parameters for ant colony optimization algorithms. In *Proceedings of the International Conference on Artificial Intelligence 2005*, pages 83–89. CSREA Press.

[10] Yaoyao Gu, Doruk Bozdağ, Robert W Brewer, and Eylem Ekici. (2006). Data harvesting with mobile elements in wireless sensor networks. *Computer Networks*, 50(17):3449–3465.

[11] Sushant Jain, Rahul C Shah, Waylon Brunette, Gaetano Borriello, and Sumit Roy. (2006). Exploiting mobility for energy efficient data collection in wireless sensor networks. *Mobile Networks and Applications*, 11(3):327–339.

[12] David Jea, Arun Somasundara, and Mani Srivastava. (2005). Multiple controlled mobile elements (data mules) for data collection in sensor networks. *Distributed Computing in Sensor Systems*, pages 466–466.

[13] Anil Kumar and Anil Thomas. (2012). Energy efficiency and network lifetime maximization in wireless sensor networks using improved ant colony optimization. *Procedia Engineering*, 38:3797–3805.

[14] Ang Li and Jian Xiao. (2012). Efficient data gathering algorithm in wireless sensor networks with optimal-path mobile sink. *Journal of Computational Information Systems*, 8(22):9269–9279.

[15] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, and Keith Sullivan. (2004). Mason: A new multi-agent simulation toolkit. In *Proceedings of the 2004 SwarmFest Workshop*.

[16] Selcuk Okdem and Dervis Karaboga. (2009). Routing in wireless sensor networks using an ant colony optimization (aco) router chip. *Sensors*, 9(2):909–921.

[17] Enda Ridge and Daniel Kudenko. (2007). Screening the parameters affecting heuristic performance. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 180–180. ACM.

[18] Rahul C Shah, Sumit Roy, Sushant Jain, and Waylon Brunette. (2003). Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks. *Ad Hoc Networks*, 1(2):215–233.

[19] Arun A Somasundara, Aditya Ramamoorthy, and Mani B Srivastava. (2004). Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines. In *Real-Time Systems Symposium, 2004. Proceedings. 25th IEEE International*, pages 296–305. IEEE.

[20] Arun A Somasundara, Aditya Ramamoorthy, and Mani B Srivastava. (2007). Mobile element scheduling with dynamic deadlines. *Mobile Computing, IEEE Transactions on*, 6(4):395–410.