MSCT: AN EFFICIENT DATA COLLECTION HEURISTIC FOR WIRELESS SENSOR NETWORKS WITH LIMITED SENSOR MEMORY CAPACITY

Murat Karakaya[1]
[1] Computer Engineering Department, Atilim University,
Incek, Ankara - Turkey
[e-mail: murat.karakaya@atilim.edu.tr]
*Corresponding author: Murat Karakaya

Received May 28, 2015; revised August 12, 2015; accepted August 15, 2015; published September 30, 2015

Abstract

Sensors used in Wireless Sensor Networks (WSN) have mostly limited capacity which affects the performance of their applications. One of the data-gathering methods is to use mobile sinks to visit these sensors so that they can save their limited battery energies from forwarding data packages to static sinks. The main disadvantage of employing mobile sinks is the delay of data collection due to relative low speed of mobile sinks. Since sensors have very limited memory capacities, whenever a mobile sink is too late to visit a sensor, that sensor's memory would be full, which is called a 'memory overflow', and thus, needs to be purged, which causes loss of collected data. In this work, a method is proposed to generate mobile sink tours, such that the number of overflows and the amount of lost data are minimized. Moreover, the proposed method does not need either the sensor locations or sensor memory status in advance. Hence, the overhead stemmed from the information exchange of these requirements are avoided. The proposed method is compared with a previously published heuristic. The simulation experiment results show the success of the proposed method over the rival heuristic with respect to the considered metrics under various parameters.

Keywords: Wireless Sensor Networks, Routing, Mobile Sink, Data Collection

http://dx.doi.org/10.3837/tiis.2015.09.007 **ISSN : 1976-7277**

1. Introduction

Wireless Sensor Networks (WSN) have been welcomed in modern life due to their advantages in implementation, service, and maintenance [1][2]. Many different applications depend on the services provided by WSN. Mostly, WSN collects environmental data via the deployed sensor nodes in the area of interest. Sensor nodes have the capability of sensing different kinds of stimulus such as temperature, magnetic, humidity, light, etc. The collected data in the sensor nodes has to be delivered to a special node, called 'sink', which has the capability of transferring the augmented data from WSN to a remote central to be processed. Most WSN applications depend on timely and accurately delivered data from the WSN itself.

There are different approaches proposed for the collection of data from sensor nodes and its transfer to a remote central [3][4][5]. According to the mobility of the sink node, these approaches can be classified into two main groups: Static Sink (SS) and Mobile Sink (MS) [5]. In the first case, the sink node is static and all the sensor nodes have to deliver their data to the SS via wireless communication. In the second, the sensor nodes wait for MS to visit them to transfer the collected data. Each approach has its own advantages and disadvantages. In a nutshell, using MS has the advantage of saving sensor energy spent in data transfer via wireless communication. However, inherently it has the drawback of high delay in data collection since MS has a relatively far slower speed than that of wireless communication.

The delay in data collection causes different problems for WSN applications. In the literature, there are various proposals to deal with these problems [6][7]. One of the problems is sensor memory overflow [8][9]. While a sensor node is sensing, it stores the reading values into its memory. When MS arrives at the sensor node, the sensor node transfers all the data stored in its memory to the MS and resets the memory. However, if MS does not arrive before the sensor memory is completely full; the sensor node faces memory overflow. In this case, all the data is deleted automatically from the memory to open space for new readings. In this way, overflows cause loss of collected data and, as such, the MS must be routed or scheduled to collect data from sensor nodes such that there will be fewer overflows and less data loss in WSN.

In this study, a solution is proposed to generate an MS tour to cover maximum sensor nodes in a way that the number of overflows can be fewer and the amount of collected data larger. The proposed method creates the tour by selecting sensor nodes so that MS can visit them before any node faces an overflow during the tour. After constructing such a tour, MS repeats the tour continuously. If the tour consists of all the sensor nodes in WSN, it is evident that there will be no overflow at all. However, if the tour can only cover some of the sensor nodes, the excluded ones will overflow. Thus, in these cases, it is aimed to generate tours to cover a maximum number of sensor nodes to minimize the overflowing.

The location of sensor nodes can be used to generate such a tour a priori. However, if sensor nodes are deployed in large numbers, their exact location might be unknown. Using localization techniques, sensor nodes can calculate their position and, then, they can share their locations with the MS. Nevertheless, in this case, sensor nodes consume a considerable amount of battery power, increasing the complexity of WSN management and sensor node hardware.

To estimate the remaining time to an overflow, MS should know the current status of each sensor node as well as their sensing rate. These requirements are challenging in practical WSN applications.

These observations motivate the creation of a new solution, which can generate a maximum sensor coverage tour without necessitating these two requirements a priori. Basically, to device a tour covering a maximum number of sensor nodes, the Nearest Neighbor heuristic which is used in solving the traveling salesman problem (TSP) and the vehicle routing problem (VRP) is adopted [10][11][12] and the 2-opt optimization technique [20] [23] is employed to improve the quality of the created tour. Moreover, to compare the success of the proposed method, a heuristic proposed in [8] and [9] is implemented.

The contributions of the present study can be explained in the following. As for the proposed method; it

- requires neither the exact coordinates of sensor locations nor the current state of sensor memory capacities a priori;
- removes the overhead of information exchanges and time synchronization between MS and sensors;
- is practical, lightweight, and scalable in terms of sensor node numbers and deployed area;
- causes fewer numbers of overflow and higher amount of collected data compared to the rival heuristic.

Moreover, by introducing a new metric for monitoring the amount of collected data enhance the performance analysis of the proposed solutions.

This paper is organized as follows. A brief introduction to data collection with MS is provided in Section I. In Section II, related studies are summarized and the problem is described in detail. The proposed method is presented in Section III. Simulation environment and the test results are discussed in Section IV. Finally, conclusions and future direction are provided in the last section.

2. Related Work

Somasundara et al. point out a data collection problem which occurs because of the slow speed of mobile sink (MS) and limited capacity of sensor memory [8]. They call it Mobile Element Scheduling (MES) problem and show that it is an NP-complete problem. To propose a solution to the MES problem, Somasundara et al. designed various heuristics, which aim to select the next sensor to be visited by MS, such that the overall number of memory overflow incidents is minimized.

The first heuristic, Earliest Deadline First (EDF), suggests moving to the sensor node which has the least remaining time to a memory overflow. However, an explicit disadvantage of the EDF heuristic is that it only considers the overflow times of sensors, but not the travel times to them to select the next node. The researchers propose another heuristic, Minimum Weighted Sum First (MWSF), in which the next sensor node is selected according to a score given in **Equation 1**. The score of each sensor node is calculated based on a given weight parameter (α), remaining overflow time (ROT), and travel time (TT) to the sensor node. The one with the minimum score is selected as the next sensor node.

$$MWSF = \alpha * ROT + (1- \alpha) TT \quad (1)$$

The weight (α) is set a number between 0 and 1. According to simulation tests, researchers reported better results for the MWSF heuristic when α is set at 0.1. Moreover, it is observed that the MWSF heuristic causes fewer overflows than those of the EDF heuristic. However, in the experiments in this study, the fraction of sensor nodes which experience an overflow is the only performance metric.

In a follow-up study [9], the researchers extended the suggested heuristics to schedule multiple MS and compare the proposed heuristics with an adapted version of the Insertion Heuristic [13]. They also introduce a new performance metric, latency, defined as "the time taken between the generation of a sample and the time of collection by the mobile element" [9]. According to reported simulation results, the MWSF heuristic leads to fewer number of overflows and less latency compared to the EDF and Insertion heuristics.

As a summary, [8] and [9] report success of the MWSF heuristic against the EDF and Insertion heuristics with respect to the observed overflow numbers and experienced latency. However, the authors do not select the amount of the collected data as a performance metric and fail to provide any results of these heuristics. Indeed, for WSN applications, the amount of the collected data is believed to have the utmost importance. Therefore, in [14], I have proposed an implementation of the Ant Colony Optimization (ACO) meta-heuristic for considering two important performance metrics together. The proposed meta-heuristic, the Mobile Element Scheduling with Time Sensitive ACO (MES/TSACO), attempts to create and improve MS schedules based on these performance metrics. It is shown that the MES/TSACO performs better than the MWSF heuristic under various simulation conditions for both performance metrics [14].

All the related works stated here assume that MS knows the exact location and the exact memory capacity status of sensor nodes all the time to prepare a schedule [8][9][14]. On the contrary, in real life applications, these requirements cannot be met easily. Thus, the implementation of these proposed solutions is limited and unpractical as discussed in the following.

In order to know the location of numerous sensor nodes, one way is to embed GPS receivers in each sensor. It is clear that this solution is expensive and impractical considering the limited battery of sensor nodes. Another solution would be to implement localization methods. In this solution, some sensor nodes either know their coordinates, or they are equipped with GPS receivers. These sensor nodes transmit beacons with their coordinates in order to assist other nodes to calculate their own locations [15]. Whatever solution is used, at the end of the localization process, all sensor nodes have to transmit their calculated coordinates to the ME. Thus, localization of sensor nodes and broadcasting of the coordinates of all the sensor nodes require a considerable amount of communication. However, this is against the main motivation and expected benefit of using mobile elements in WSN which is to prevent or limit these kinds of communications to save the sensor battery and extend the WSN's lifespan [5][16]. Hence, proposing solutions to this MES problem based on sensor locations is not practical.

The other assumption, knowing in time and accurately the memory status of all sensor nodes, has its own challenges as well since it requires that sensor nodes broadcast their memory status to MS, causing communication overheads. Moreover, the proposed solutions presume that the sensing rate of the nodes is pre-determined and fixed. However, there are various techniques proposed in the literature to manage this rate for energy conservation [17]. These techniques aim to reduce energy consumption in sensing by decreasing sensing frequency dynamically and according to the application purposes and sensed phenomena. That

is, the assumption of steady sensing rate for all applications is not practical either. To implement the proposed solutions when the sensing rate changes dynamically, all sensor nodes have to broadcast their current memory status to ME which causes communication storm in WSN and diminishes sensor battery drastically.

Besides these requirements, the MES/TSACO and the MWSF heuristic involve computations increasing with the number of sensors which could be a bottleneck as WSN have large number of sensor nodes in real deployments [1][4].

To relax these impractical assumptions of the proposed algorithms and to eliminate their drawbacks, a novel approach which does not need information exchanges for the locations and memory capacity status of the sensor nodes is proposed. In addition, it does not cause heavy computation. In the next section, the details of the proposed solution are explained.

3. Maximum Sensor Coverage Tour Heuristic

The proposed Maximum Sensor Coverage Tour (MSCT) method is designed based on the following observations. Assume that a sensor node has a memory capacity of MC KB and a sensing rate of SR B/sec . If the sensor node begins with an empty memory, the time duration of filling the memory completely is computed according to the formula given in **Equation 2**.

$$MTD=MC/(SR/1024) \quad (2)$$

Thus, after visiting a sensor node, MS has maximum MTD seconds to return to this node before its memory is overflowed. This duration is called Maximum Tour Duration (MTD). Based on this observation, the MSCT method creates a tour in which MS begins from the SS, visits some of the sensor nodes and returns to the SS before MTD expires. Thus, any sensor nodes included in this tour is immune from any overflow. The MSCT method aims to include as many sensor nodes as possible to this tour to minimize the number of overflows in WSN. In essence, if the MSCT method is able to create a tour including all the nodes in the WSN during MTD, there will be no overflow at all. However, in practice, MTD would limit the number of nodes to be included; in that case, the problem is now to create an efficient tour to consist of a maximum number of nodes. Moreover, due to the topology of sensors, the realized tour would take less time than MTD.

Another important observation is that the amount of expected collected data (ECD) from each visited sensor depends on the realized tour duration (RTD) and the sensing rate (SR), as given in **Equation 3**.

$$ECD= RTD*SR \quad (3)$$

If RTD is very close to MTD, MS collects as much data as the sensor memory capacity allows. That is, MS arrives at the sensors just before their memory is completely full. Therefore, it is expected that the MSCT method is able to collect more data as well. Moreover, since MS periodically visits all the sensor nodes included in the tour and the amount of the data collected during visit times are fixed, the MSCT method does not need to be aware of the current memory status of each sensor. In order to overcome the requirement of knowing the sensor locations in advance, the MSCT method implements three phases: Localization Discovery, Tour Construction, and Data Collection Phase. In the first phase, MS discovers the exact locations of the sensors, which is followed by the second phase where initial tour is constructed and improved. In the last phase, MS continously collects data following the optimized tour. In the following, the details of these phases are provided.

3.1 Localization Discovery Phase

For localization of sensor nodes in WSN, various techniques are proposed in the literature [23]. Since MSCT employs a Mobile Sink (MS) to collect data from static sensor nodes, path planning localization algorithms such as [18][19][25] are suitable to be implemented in the proposed solution. In general, in path planning localization algorithms, there is a mobile landmark which knows its exact location, usually via an embedded GPS receiver. The mobile landmark travels on a specific trajectory and broadcast its location beacon. One main problem is to devise a method to create the mobile landmark trajectory such that discovery of the sensor nodes is efficient. In [18], the authors propose to use a spiral trajectory whereas in [19] the authors suggest to use four different trajectories: SCAN, HILBERT, CIRCLES, and S-Curves. Unlike [18] and [19], MSCT does not require MS to follow a specific pre-determined fixed trajectory in the proposed solution. Instead, MS creates its own route dynamically based on the local information obtained from the surrounding sensor nodes following the Nearest Neighbor heuristic as discussed in Section 3.2.

Upon receiving the mobile landmark location beacon, sensor nodes can estimate their position. There are several solutions such as [25][26][27][28] among others to estimate a sensor node's location with location beacons. Among them, the proposed localization method in [27] can easily be implemented into MSCT because this method only requires a single mobile landmark broadcast beacon messages. Moreover, this method provides energy saving and scalability by providing a means to nodes to compute their own position using only local information. In [27], the authors propose two methods (parametric and non-parametric) based on Time of arrival (TOA) of the received signal for ranging and maximum likelihood estimator (MLE) technique for localization. Using frequent location beacons, sensor nodes continuously improve MLE and, hence, increase the location precision.

In our case, mobile landmark is MS. Sensor nodes are supposed to listen to MS's location beacons after deployment. MS starts its tour from a static sink (SS) by continuously broadcasting its current location. The sensor nodes which receive the location beacon calculate their position according to the non-parametric estimation formulation given in [27], and broadcast their estimated positions to nearby MS. Having learning the locations of surrounding sensor nodes, MS selects the nearest sensor node according to its current location and estimated sensor locations. MS heads toward the selected sensor node and, upon arriving, it stores the exact sensor location by using its embedded GPS. MS also transfers all the data stored in the sensor memory and resets it. The visited sensor node does not further listen to MS location beacons to save its battery. MS continues to broadcast its location to discover new sensor nodes during visiting sensor nodes.

3.2 Tour Construction Phase

MS continues its tour by selecting the nearest sensor node which has not been visited yet. Before moving toward the selected nearest sensor node, it calculates the remaining MTD and compares it with the time required to travel to the SS via the selected node. If there is not enough time, MS selects another sensor node to check if it is possible to continue the tour by visiting that sensor before MTD expires. If there is no such sensor node, then MS directly returns to the SS.

Fig. 1. 2-Opt removes intersecting sub-tours (on the left) and may produce shorter path (on the right)

As a result, the first round of Tour Construction Phase is finalized by constructing the initial tour. Before beginning the second round, MCST optimizes the generated tour by applying the 2-Opt optimization technique. 2-Opt is a local search method that was proposed by Croes in 1958 [20] to rearrange a given path such that intersecting sub-tours are removed with new edges which can shorten the total length of the path (see **Fig. 1**). That is, for a given tour (provided by MSCT in our case), 2-Opt iteratively reverses one of the two intersecting sub-tours and if the length of the new tour is shorter than one of the original one, the tour is rearranged with this order. 2-Opt repeats this process until reaching a local optimum in which no further improving is possible. One popular application area of 2-opt is TSP based problems to produce shorter path from an existing path [23]. In this work, 2-opt technique is applied to the tours generated at each round of the Network Discovery phase to decrease the tour time by rearranging the order of visited sensors so that a new sensor node can be appended to the rearranged path in the next round.

Thus, for the next round, MS follows the optimized tour. If the 2-Opt optimization technique produces a shorter tour, MS follows the optimized tour until reaching the final sensor. Before returning to the SS, it attempts to extend the tour by appending new sensor nodes to the current tour as explained in Section 3.1 and 3.2. In the end, MS returns to the SS and finalizes the second round.

MS continues this phase with new rounds as long as the 2-Opt optimization technique can produce shorter tours. However, if this technique fails to optimize the existing tour any longer, then the Network Discovery and Tour Construction Phase is terminated.

3.3 Data Collection Phase

Once the Network Discovery and Tour Construction Phase sets up an optimized tour, in the Data Collection Phase, MS follows this tour repeatedly to collect data.

4. Simulation and Test Results

WSN environment and the proposed solutions are simulated using MASON simulation library in Java [21]. Basically, two sets of simulation scenarios have been experimented: in the first set of experiments, a smaller topology with 52 sensors nodes is used whereas in the second set of experiments a larger topology including 76 sensor nodes are employed to observe the scalability of the proposed heuristic. Below, the details of these parameters and experiment setups are given.

4.1 Simulation Parameters

The simulation parameters and their values are given in **Table 1**. Unless stated explicitly, the default values of the parameters are used in experiments.

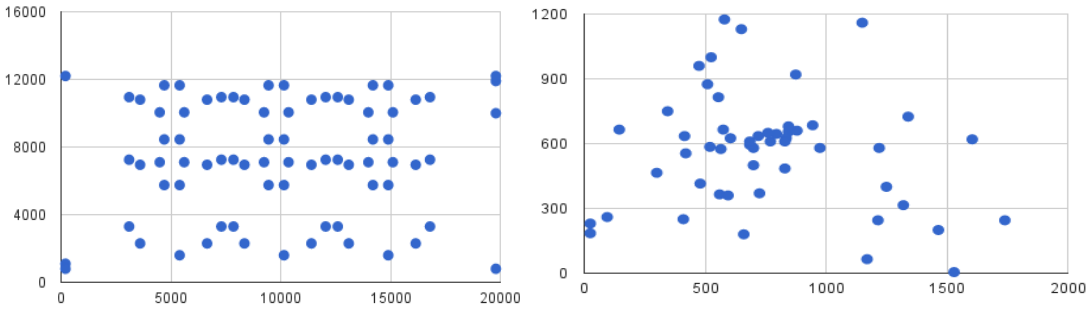

Fig. 2. Node topology of PR76 (on the left) and Berlin52 (on the right) TSP file

Table 1. Simulation parameters

Parameter	Default	Range / Alternative	Notes
MS Speed	4	4 - 36	km/h
Sensor Memory	1	1-9	KB
Sensing Rate	1	-	KB/sec
Initial Sensor Memory State	Empty	-	
Simulation Duration	(Memory Capacity / Sensing Rate) *20	-	seconds
WSN Topology	Berlin52	PR76	TSP files
Warm-up Duration	(Memory Capacity / Sensing Rate) *3		

WSN Topology: The locations of sensor nodes are taken from two TSPLIB benchmark problems available in [22]. The first file, Berlin52, has 52 coordinates dispersed on an area of 1.2 x 2 kilometers whereas the other file, PR76, has almost 50% more nodes (76 coordinates) which are scattered in a hundred times larger area than that of Berlin52 as seen in **Fig. 2**.

Simulation Duration: Simulation duration is determined as a function of sensor memory capacity and sensing rate to have adequate observation time of about 20 overflows for each sensor node to guarantee stable performance results. For instance, if the sensor memory capacity is 1KB with the sensing rate 1 B/sec, one overflow takes about 17 (1024/60) minutes. Thus, simulation duration is set for 5.6 (17*20) hours.

Number of Runs: A single simulation test is repeated as the number of nodes in the selected TSP benchmark problem by selecting each node as a Static Sink (SS). In each test, starting from the SS, the mobile sink (MS) visits the nodes according to the selected method until the end of the simulation duration. Then, the results of all the runs are taken into account to report the average result for the test.

Alternative Method: For the purpose of the comparison, the Minimum Weighted Sum First (MWSF) heuristic is also implemented. During the experiments, α value is set at 0.1 as it has been reported to produce better results in previous studies [8] [9].

Warm-up Duration: As the proposed method includes the Network Discovery and Tour Construction Phase, it requires some time to prepare an optimum tour. Thus, the warm-up duration is reserved before collecting the data of the observed metrics.

4.2 Performance Metrics

Number of overflows and amount of the collected data are the main performance metrics. To compare the proposed solutions, an improvement ratio for number of overflows (ON) and amount of collected data (CD) is also defined as in **Equation (4)** and **(5)**.

$$ON = \frac{ON_{MWSF} - ON_{MSCT}}{ON_{MWSF}} \quad (4)$$

$$CD = \frac{CD_{MCST} - CD_{MWSF}}{CD_{MWSF}} \quad (5)$$

The results of the first experiment set for Berlin52 topology are provided in details in Sections 4.3, 4.4, and 4.5. In Section 4.6, results for the PR76 topology are given. In each set of experiment, first, the success of the proposed solution with respect to the number of sensor nodes covered in these tours is discussed. Then, the overflow incidents and the collected data of the MSCT and MWSF methods are compared. Lastly, the impact of sensor memory capacity, MS speed, and different sensor topology on the observed performance metrics are investigated.

4.3 Evaluation of Created Tours

The number of nodes in a tour has a crucial importance on the success of the proposed method. Therefore, the relationship among the number of sensors included, memory capacity, and MS speed is examined. The experiment results are presented in **Fig. 3** for different memory capacities and for different MS velocities. To make the results more readable, the Sensor Coverage (SC) ratio is defined as follows.

$$SC = \frac{RT}{WSN} * 100 \quad (6)$$

In **Equation 6**, RT is the number of sensor nodes in the realized tour and WSN is the number of sensor nodes in WSN.

As seen on the left in **Fig. 3**, sensor coverage is increased as the sensor node memory capacity increases. Thus, larger memory capacity enables the MS to visit more sensors before any overflow occurs at the visited nodes. For the given topology in the TSP benchmark file, if the sensor nodes have a memory capacity of 9KB and MS speed is 4 km/h, no overflow is expected to happen at all since all the sensor nodes in the WSN are covered.

Similarly, on the right in **Fig. 3**, it is seen that MS can visit more nodes if its speed increases while the memory capacity of sensors is kept as 1KB. By travelling at 36km/h MS is able to visit all the nodes in WSN, which implies that none of the sensor nodes experience memory overflow.

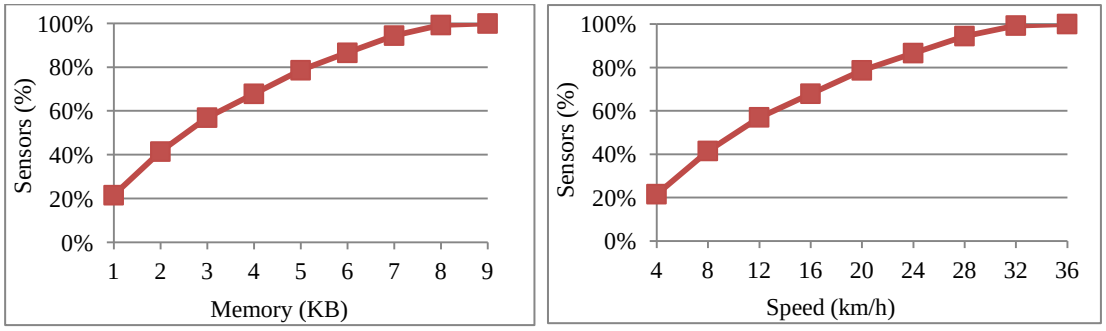

Fig. 3. Sensor coverage of created tours for different memory capacity (on the left) and MS speed values (on the right)

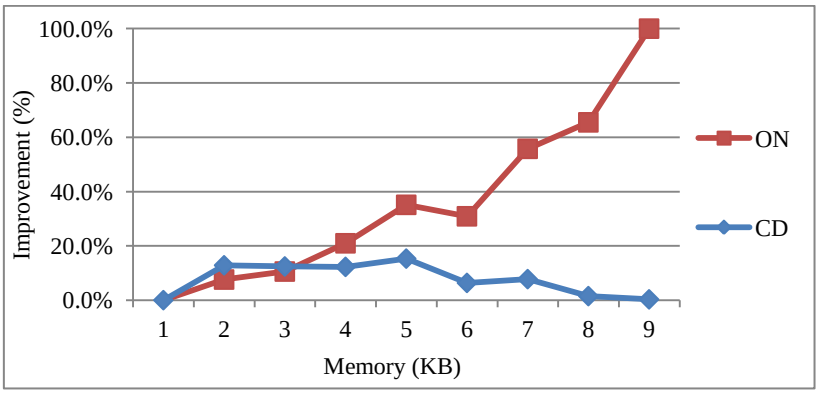

Fig. 4. . Improvement in overflow number (ON) and amount of collected data (CD) for different memory capacity values

Table 2. Number of overflows and amount of collected data for different memory capacity values

Memory (KB)	Average Overflow Number		Amount of Average Collected Data (MB)	
	MSCT	MWSF	MSCT	MWSF
1	693,26	693,16	24,17	24,15
2	524,60	568,01	89,97	79,70
3	357,76	400,10	189,26	168,29
4	267,20	338,18	300,73	267,80
5	177,60	273,82	435,05	377,16
6	111,36	161,20	575,64	541,03
7	46,08	104,18	732,53	679,55
8	5,76	16,70	878,31	865,00
9	0	1,56	994,15	990,79

Fig. 3 indicates that it is possible to create tours to visit all nodes without any overflows provided that memory capacity and MS speed are set accordingly. In the following experiments, the result of the cases when it is not possible to create such tours is observed.

4.4 Impact of Memory Capacity on Performance Metrics

In Section 4.1, the findings show that, if the sensor nodes have a larger memory capacity or the MS travels at a higher speed, then it is possible to create a tour to cover all the nodes and to be completed before any overflow occurs. However, in real life applications, memory capacity is a scarce resource and cannot be made available in abundance. To investigate the impact of insufficient memory, several experiments are conducted. The resulting overflow numbers and the amount of collected data are presented in **Table 2** for different memory capacities. In general, the MSCT improves the overflow number (ON) and amount of collected data (CD) compared to the MSWF heuristic as seen in **Fig. 4**.

According to **Table 2** and **Fig. 4**, the MSCT causes considerably fewer number of overflows compared to those of MWSF at all memory capacities. For example, when sensors have 5 KB memory capacity, the MSCT method causes about 35% fewer cases of overflow than that of the MWSF heuristic. Moreover, to attain a similar number of overflows, the MWSF heuristic requires more memory than that of the MSCT method. For instance, when memory capacity is 4 KB and the MSCT method is applied, 267 overflows happen on average. However, to decrease the number of overflows to this level, the MWSF heuristic requires 5

KB of memory, which is 20% more than that of the MSCT method.

Table 2 also provides the amount of collected data by both heuristics. The MSCT method collects more data than that of the MWSF heuristic for all memory capacities. Another observation is that both heuristics collect a maximum amount of data when memory capacity prevents any overflow from occurring. However, this happens at 9 KB memory for the MSCT method, but at 10 KB memory for the MWSF heuristic. That is, the MWSF heuristic requires more memory capacity to collect a similar amount of data as the MSCT method does.

The stated results show that the MSCT heuristic performs well under different memory capacities by creating fewer overflows and collecting more data compared to the MWSF heuristic. Moreover, the MSCT heuristic needs less memory to create an overflow-free WSN. In the following section, the effects of different MS velocities on the performance of both heuristics are investigated.

4.5 Impact of MS Speed on Performance Metrics

As discussed in Section 4.1., increasing MS speed helps to create tours including more sensor nodes. Unfortunately, in real life, the MS speed may not be always increased to desired levels due to environmental constraints. Therefore, understanding the impact of speed on the success of the proposed method is important. In this section, the results of applying different MS velocities while maintaining the sensor memory at 4 KB are provided.

In **Table 3**, the number of overflows is decreased as MS speed increases. This is due to the fact that when MS has a higher speed; it increases its chances to visit more sensor nodes for both solutions. However, the MSCT method causes substantially fewer overflows compared to the MWSF heuristic, especially at higher speeds. For instance, when MS has a speed of 28 km/h, about 40% less overflow occurs if the MSCT method is applied. This observation is in line with the sensor coverage ratios reported in Section 4.1. As a result, one can argue that the MWSF heuristic entails MS to travel at a relatively higher speed to reduce the overflow incidents with respect to the MSCT method (see **Fig. 5**).

For the amount of data collected from the sensor nodes, parallel to the decreasing number of overflows, more data is collected as MS travels at a higher speed, as seen in **Table 3**. It is also observed that, in lower speeds, the MSCT method performs far better than the MWSF heuristic. For instance, speed values between 8-20 km/h, the MSCT method collects about 12% more data than MWSF. In higher speeds, nevertheless, such improvement becomes smaller due to the fact that both methods have abundant time to visit all sensors before any of them overflows.

Table 3. Number of overflows and amount of collected data for different MS speed values

Speed (km/h)	Average Overflow Number		Amount of Average Collected Data (MB)	
	MSCT	MWSF	MSCT	MWSF
4	693,26	693,16	24,27	24,15
8	524,62	566,12	44,98	40,05
12	380,12	419,34	63,08	54,94
16	296,82	350,54	73,55	63,52
20	203,20	282,80	85,18	75,47
24	118,32	167,42	95,92	90,10
28	65,96	108,90	102,52	97,08
32	6,12	17,80	109,77	108,16
36	0	2,48	110,45	110,02

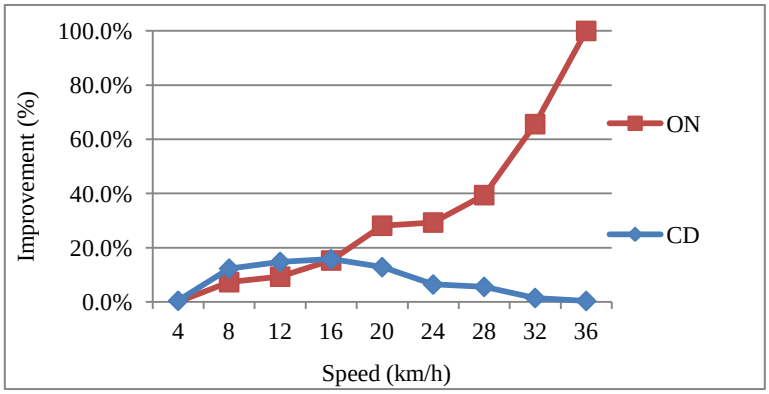

Fig. 5. Improvement in overflow number (ON) and amount of collected data (CD) for different MS speed values

4.6 Impact of Different Sensor Topology on Performance Metrics

So far, the experiments have been conducted using the given topology in the Berlin52 TSP benchmark file. The topology and the distances among the sensors can affect the results. Therefore, another TSP benchmark file is used to experiment the proposed method. Since the topology and the distances among the sensors are different, here the results of experiments with different values of sensor memory capacity and MS speed are reported.

Fig. 6 depicts the sensor coverage ratios according to different memory capacity (above) and MS speed values (below) for PR76 topology. These results show that the MSCT method can create tours to cover a maximum number of sensors for a different topology as well.

In **Table 4**, the results for various memory capacities are presented when MS moves at a speed at 32 km/h. **Fig. 7** and **Fig. 8** illustrates the gained improvement thanks to the MSCT method for various MS speed and sensor memory capacity values.

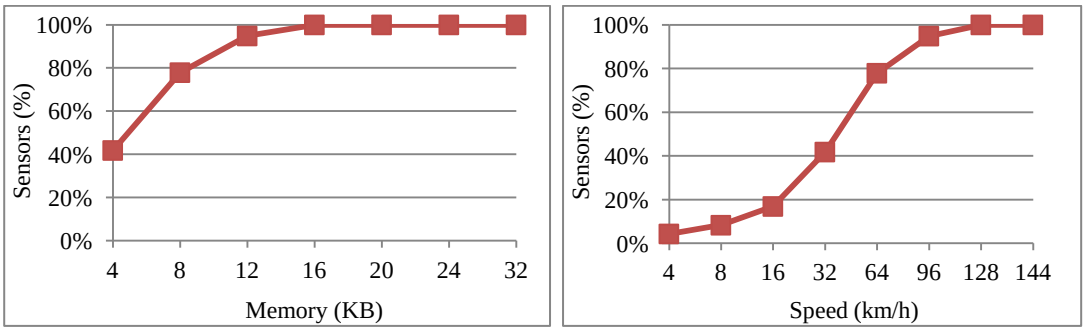

Fig. 6. Sensor coverage of created tours for different memory capacity **(left)** and MS speed values **(right)** for PR76 topology

Table 4. Number of overflows and amount of collected data for different MS speed values

Memory (KB)	Average Overflow Number		Amount of Average Collected Data (MB)	
	MSCT	MWSF	MSCT	MWSF
1	1184,9	1219,3	13,40	9,64
4	707,84	830,92	270,24	218,03
8	269,89	479,44	1007,54	802,15
12	62,40	111,98	1840,54	1762,35
16	0	58,60	2583,40	2462,07
20	0	16,50	3230,82	3183,47
24	0	4,76	3878,20	3858,42
32	0	0	5166,68	5163,28

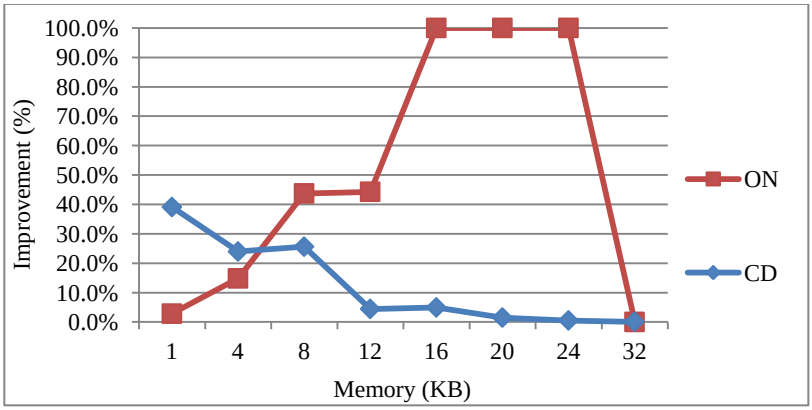

Fig. 7. Improvement in overflow number (ON) and amount of collected data (CD) for different sensor memory capacity values when PR76 topology used

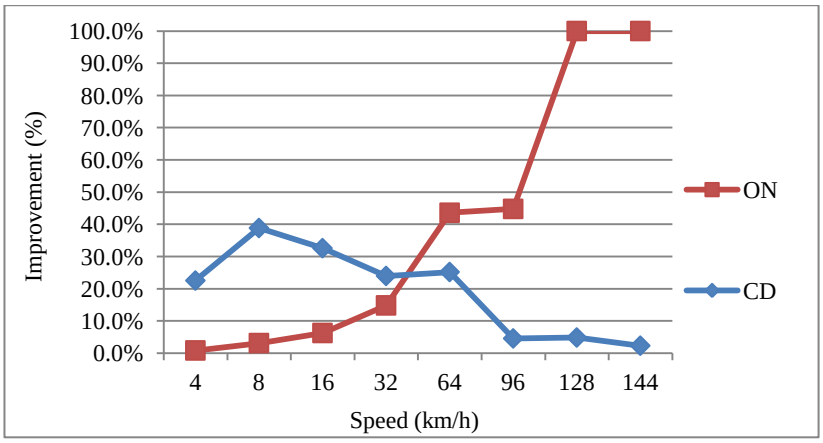

Fig. 8. Improvement in overflow number (ON) and amount of collected data (CD) for different MS speed values when PR76 topology used

In **Table 5**, the test results for the PR76 TSP benchmark problem is provided for different MS speeds assuming that sensor memory capacity is 4 KB. The first observation in these

Table 5. Number of overflows and amount of collected data for different MS speed values when PR76 topology used

Speed (km/h)	Average Overflow Number		Amount of Average Collected Data (MB)	
	MSCT	MWSF	MSCT	MWSF
4	1163,84	1173,54	27,77	22,66
8	1115,20	1151,02	53,60	38,61
16	1010,88	1078,22	109,22	82,38
32	707,84	830,92	270,24	218,03
64	269,12	476,90	503,79	402,46
96	62,40	112,98	613,52	586,79
128	0	57,62	645,87	616,03
144	0	27,10	645,48	630,93

results is the fact that the success trend of the MSCT method over the MWSF heuristic is evident. Thus, the MSCT method is considered robust in the presence of a different topology as discussed below.

In both results, the MSCT method performs far better than the MWSF heuristic for the low values of sensor memory capacity and MS speed with respect to the amount of collected data. For example, when memory is less than 8 KB or MS speed is less than 64 km/h, the MSCT method collects 25 -30 % more data compared to the MWSF heuristic. As WSN designers prefer low sensor memory capacity due to the cost and low MS speed due to terrain limitations, this property is an obvious advantage.

When sensor memory capacity or MS speed is set at higher values, the MSCT method improves the MWSF heuristic results slightly. This is due to the fact that these conditions extend the sensor overflow time considerably and both methods exploit this extension to cover most or all of the sensors before any overflow. In reality though, one cannot expect that either of sensor memory capacity or MS speed to be plentiful.

Another important observation is that, for the very low values of memory capacity or MS speed, even the improvement in the overflow numbers is limited while the improvement in the amount of collected data is considerably large. It can be concluded that even though the overflow incidents happen at similar levels for both methods, the MSCT method is able to collect more data as expected and explained in Section 3. This is because the MSCT method schedules the MS to re-visit sensor nodes just before a memory overflow happens. That is, the MS collects almost 100% of the data from each sensor. On the other hand, the accumulated data in sensor memory has a weight of 90% on the MWSF heuristic's sensor selection criteria as given in **Equation 1**. As a result, the MWSF heuristic can prefer a sensor node with less data, but closer to the MS location.

5. Conclusion

In this study, the Maximum Sensor Coverage Tour (MSCT) method is introduced to efficiently collect data with a mobile sink in a WSN composed of sensors with limited memory. The proposed method does not require information exchanges about sensor locations and memory status. Instead, the MSCT method discovers the deployed sensors and generates a tour such that the sensor nodes in the tour are re-visited before any memory overflow occurs. To incorporate a maximum number of sensor nodes into the tour, the MSCT method employs the Nearest Neighbor heuristic and the 2-opt optimization technique.

In the experiments, it is observed that the proposed heuristic can generate such tours successfully. Moreover, the MSCT method outperforms the alternative heuristic for different memory capacities, MS speeds, and sensor topologies with respect to the number of overflows and the amount of collected data. The improvements on these performance metrics are essential for data collection in WSN applications.

As a future study, we would like to extend the study by scheduling multiple MS and introducing uncertainty in travelling time, dynamic sensing rate, and mobile sensors.

References

[1] Prathap, U.; Shenoy, D.P., Venugopal, K.R., Patnaik, L.M., "Wireless sensor networks applications and routing protocols: survey and research challenges," in *Proc. of Cloud and Services Computing (ISCOS)*, *2012 International Symposium on.* IEEE, 2012. Article (CrossRef Link)

[2] Durisic, Milica Pejanovic, et al., "A survey of military applications of wireless sensor networks," in *Proc. of Embedded Computing (MECO), 2012 Mediterranean Conference on*. IEEE, 2012.

[3] Han, Kai, Jun Luo, Yang Liu, and Athanasios V. Vasilakos. "Algorithm design for data communications in duty-cycled wireless sensor networks: A survey," *Communications Magazine*, IEEE 51.7 (2013): 107-113. Article (CrossRef Link)

[4] Wang, Feng, and Jiangchuan Liu, "Networked wireless sensor data collection: issues, challenges, and approaches," *Communications Surveys & Tutorials*, IEEE 13.4, 673-687, 2011. Article (CrossRef Link)

[5] Di Francesco, Mario, Sajal K. Das, and Giuseppe Anastasi, "Data collection in wireless sensor networks with mobile elements: A survey," *ACM Transactions on Sensor Networks (TOSN)* 8.1, 7, 2011. Article (CrossRef Link)

[6] Javaid, Nadeem, et al., "SRP-MS: A new routing protocol for delay tolerant Wireless Sensor Networks," in *Proc. of Electrical and Computer Engineering (CCECE), 26th Annual IEEE Canadian Conference on. IEEE*, 2013. Article (CrossRef Link)

[7] Akbar, M., et al., "On modeling geometric joint sink mobility with delay-tolerant cluster-less Wireless Sensor Networks," in *Proc. of Smart Communications in Network Technologies (SaCoNeT)*, *2013 International Conference on.* Vol. 1. IEEE, 2013. Article (CrossRef Link)

[8] Somasundara, Arun A., Aditya Ramamoorthy, and Mani B. Srivastava, "Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines," in *Proc. of Real-Time Systems Symposium, 25th IEEE International*, 2004. Article (CrossRef Link)

[9] Somasundara, Arun A., Aditya Ramamoorthy, and Mani B. Srivastava, "Mobile element scheduling with dynamic deadlines," *Mobile Computing, IEEE Transactions on,* 6.4, 395-410, 2007. Article (CrossRef Link)

[10] Tsai, Cheng-Fa, Chun-Wei Tsai, and Ching-Chang Tseng, "A new hybrid heuristic approach for solving large traveling salesman problem." *Information Sciences* 166.1, 67-81, 2004. Article (CrossRef Link)

[11] Lenstra, Jan Karel, and A. H. G. Kan, "Complexity of vehicle routing and scheduling problems," *Networks* 11.2, 221-227, 1981. Article (CrossRef Link)

[12] Laporte, Gilbert, et al., "Classical and modern heuristics for the vehicle routing problem," *International transactions in operational research* 7.45, 285-300, 2000. Article (CrossRef Link)

[13] Solomon, Marius M., "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations research* 35.2, 254-265, 1987. Article (CrossRef Link)

[14] Karakaya, Murat. "Time-Sensitive Ant Colony Optimization to Schedule A Mobile Sink for Data Collection in Wireless Sensor Networks," *The Ad Hoc & Sensor Wireless Networks*, Vol.28, No.1-2, p. 65-82, 2015.

[15] Han, Guangjie, et al., "Localization algorithms of wireless sensor networks: a survey," *Telecommunication Systems* 52.4, 2419-2436, 2013. Article (CrossRef Link)

[16] Karakaya, Murat, "Deadline-aware energy-efficient query scheduling in wireless sensor networks

with mobile sink," *The Scientific World Journal*, 2013. Article (CrossRef Link)
[17] Anastasi, Giuseppe, et al., "Energy conservation in wireless sensor networks: A survey," *Ad hoc networks* 7.3, 537-568, 2009. Article (CrossRef Link)
[18] Hu, Zhen, et al., "Localization in wireless sensor networks using a mobile anchor node," in *Proc. of Advanced Intelligent Mechatronics*, IEEE/ASME International Conference on. IEEE, 2008. Article (CrossRef Link)
[19] Huang, Rui, and Gergely V. Zaruba, "Static path planning for mobile beacons to localize sensor networks," in *Proc. of Pervasive Computing and Communications Workshops*, *Fifth Annual IEEE International Conference on*. *IEEE*, 2007. Article (CrossRef Link)
[20] Croes, Georges A. "A method for solving traveling-salesman problems," *Operations Research* 6.6, 791-812, 1958. Article (CrossRef Link)
[21] Luke, Sean, et al. "Mason: A multiagent simulation environment," *Simulation 81.7*, 517-527, 2005. Article (CrossRef Link)
[22] TSPLIB web site, http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/, visited February 2015.
[23] Englert, Matthias, Heiko Röglin, and Berthold Vöcking, "Worst case and probabilistic analysis of the 2-Opt algorithm for the TSP," *Algorithmica* 68.1, 190-264, 2014. Article (CrossRef Link)
[24] Han, Guangjie, et al., "Localization algorithms of wireless sensor networks: a survey," *Telecommunication Systems* 52.4, 2419-2436, 2013. Article (CrossRef Link)
[25] Ou, Chia-Ho, and Wei-Lun He, "Path planning algorithm for mobile anchor-based localization in wireless sensor networks," *Sensors Journal*, IEEE 13.2, 466-475, 2013. Article (CrossRef Link)
[26] Xu, Jiang, and Huan-Yan Qian, "Localization of Wireless Sensor Networks with a Mobile Beacon," *Information Technology Journal* 12.11, 2251, 2013. Article (CrossRef Link)
[27] Guo-Lin Sun, Wei Guo, "Comparison of distributed localization algorithms for sensor network with a mobile beacon," in *Proc. of Networking, Sensing and Control*, *IEEE International Conference on*, vol.1, no., pp.536,540 Vol.1, 21-23, March 2004. Article (CrossRef Link)
[28] Kuo-Feng Ssu; Chia-Ho Ou; Jiau, H.C., "Localization with mobile anchor points in wireless sensor networks," *Vehicular Technology*, IEEE Transactions on, vol.54, no.3, pp.1187-1197, May 2005. Article (CrossRef Link)

Murat KARAKAYA received the B.S.E.E. degree in 1991 from the Turkish Military Academy (KHO), Ankara, Turkey, and the M.S. and Ph. D. degrees in Computer Engineering from the Bilkent University, Ankara, Turkey in 2000 and 2008, respectively. From 1992 to 2000, he worked as an engineer at different units in the Turkish Land Forces (KKK), Ankara, Turkey. From 2000 to 2005, he worked as an instructor and software engineer at the Turkish Military Academy (KHO), Ankara, Turkey. Then, during 2005-2008 he worked as IT Project Manager in the North Atlantic Treaty Organization (NATO) Brussels, Belgium. From 2008 to 2012, he worked as an instructor and software engineer at the Turkish Military School of Electronics, Communications and Information Systems (MEBS) and Turkish Military Academy (KHO), Ankara, Turkey. He joined the faculty of Atilim University in 2012 and is currently an Asst. Professor in the department of Computer Engineering, Ankara, Turkey. His research interests are natural computing, sensor networks, peer-to-peer networks, optimization, and communications protocol design