2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems

Dynamic Mobile Charger Scheduling in Heterogeneous Wireless Sensor Networks

Hua Huang* Shan Lin* Lin Chen† Jie Gao‡ Anwar Mamat§ Jie Wu¶

* *Department of Electrical and Computer Engineering, Stony Brook University. {hua.huang, shan.x.lin}@stonybrook.edu*
† *Department of Computer Science, The University of Paris-Sud. chen@lri.fr*
‡ *Department of Computer Science, Stony Brook University. jgao@cs.stonybrook.edu*
§ *Department of Computer Science, University of Maryland College Park. anwar@cs.umd.edu*
¶ *Department of Computer and Information Sciences, Temple University. jiewu@temple.edu*

***Abstract*—Recent advances in energy transfer technology is boosting the development of renewable sensor networks. To sustain such a network, a mobile robot travels from node to node to recharge each sensor before its battery runs out. Consider each node's recharge as a real-time task; the robot needs to serve these tasks by their deadlines. This represents a class of challenging mobility scheduling problems, where the nodes' deadlines and spatial distribution are often at odds with each other. In this paper, we focus on the scenario where nodes have heterogeneous energy consumption rates, and our goal is to maximize the percentage of nodes alive. We formulate this scheduling problem and prove its NP-completeness. To solve this problem, we propose a spatial dependent task scheduling algorithm, which quantifies the impact of scheduling proximate tasks on the other tasks. With extensive simulations, we reveal the trade-offs of existing solutions under a wide range of network scenarios. Our evaluation results show that our algorithms out-perform classical TSP scheduler by up to 10% and 85% in terms of coverage ratio and average tardiness, respectively.**

***Keywords*-Mobile Charger Scheduling, Real-Time, Spatial Dependent Task**

I. Introduction

Mobile robots are widely used in wireless sensor networks [1, 2], such as data mules [3] and event response [4]. Typically, a robot processes tasks by visiting nodes one by one in the network. In this paper, we investigate one instance of such mobility scheduling problems: the mobile charging application, where a mobile robot recharges each sensor node to replenish its battery storage. Sensor network applications usually require a certain percentage of alive nodes and nodes may have different energy consumption rates. To address these realistic issues, we consider the mobile charger scheduling problem: given each recharge as a real-time task, find the traveling path that allows the mobile robot to process these tasks so that the percentage of alive nodes is maximized. As both deadlines and spatial distributions of nodes affect the selection of traveling path, this problem is very challenging.

We firstly formulate the mobile charger scheduling problem, then prove its NP-completeness. The solution to this problem depends on the relative ratio between the traveling time of the charger and the battery lives of the sensor nodes. If the traveling time to traverse the network is much shorter than the battery lives of the sensor nodes, then no nodes require more than one recharge in each round. The scheduling problem is reduced to the Traveling Salesman Problem with Time Window[5–8], which was studied before. On the other hand, if the traveling time is larger than some or all of the sensor nodes' battery lives, this problem gets more complicated because each time a sensor node is charged, its deadline is reset. In this case, we aim at finding schedules responding to such dynamically generated deadlines[3], and maintain high percentage of alive nodes over time. To reveal the complexity of the problem, we investigate the performances of existing solutions, such as the Earliest Deadline First (EDF) and Traveling Salesman (TSP) types of algorithms. We show EDF scheduler rescues urgent nodes, but does not take spatial distribution of nodes into account. Whereas, TSP scheduler minimizes the traveling cost, but fails to respond to the most urgent node in time. Therefore, it is essential to consider spatial and temporal constraints together in this scenario.

To address this problem, we propose a spatial dependent task model to quantify the impact of scheduling proximate tasks on the other tasks. Intuitively, the traveling time for recharging nearby urgent nodes is less than faraway urgent nodes. Therefore, we can optimize the recharging schedule based on the proximity and density of nodes. With the spatial dependent task model, we design the Spatial Dependent Task scheduler. It first identifies clusters of nodes such that recharging them maximizes the network energy level, then computes the traveling path to reach these clusters, such that more nodes can be recharged without incurring much detour.

We implemented a comprehensive simulation frame-

978-1-4673-9101-6/15 $31.00 © 2015 IEEE
DOI 10.1109/MASS.2015.26

work, and conducted extensive trace-driven experiments to test different scheduling algorithms under various settings. The simulation uses energy consumption data traces, which are collected from real sensor network deployments [9]. Experimental results show that our SDT scheduler outperforms the classical shortest route scheduler (TSP) by up to 10% coverage ratio, and reduces the average tardiness by up to 85%. The contributions of this paper are listed as follows:

- We define the mobile charger scheduling problem based on realistic sensor network constraints, and prove its NP-completeness.
- We propose the Spatial Dependent Task scheduling algorithm that finds a good trade-off between spatial distribution and real-time urgency of recharge requests.
- With energy consumption traces from real sensor network deployments, extensive simulations are conducted to compare our algorithms with five representative baseline solutions. Significant improvements in both coverage ratio and average delay are achieved.

II. Related Work

Mobile charger scheduling algorithms were proposed to sustain the operation of sensor networks [10–12]. There have been many variants of the mobile charger scheduling problem, depending on the sets of assumptions and optimization goals adopted. Zhang et al. [10] propose algorithms to minimize the number of mobile chargers necessary to keep all the sensor nodes alive in a 1-d space. In [11], the authors assume that the charging time for sensor nodes is much longer than the mobile charger's traveling time, and thus the scheduling algorithms are less focused on the spatial aspects of the problem. In [12], the authors assume that the mobile charger can recharge all the sensor nodes lying within a distance, and their optimization goal is to minimize the time it takes for each round of recharge. In their recent paper [13], He et al. consider the heterogeneous power of the sensor nodes. They divide the sensor nodes into different groups, and apply TSP algorithms to recharge nodes within each group. In [3, 14], Somasundara et al. formulate the mobile element scheduling problem, which aims at minimizing the overall tardiness. They formulate it into an integer programming problem, and propose several heuristic algorithms.

In most of the previous research, the goals are to optimize recharge efficiency while regarding the sensor node deadlines as hard deadlines. However, in many realistic scenarios, especially when the charging workload is heavy, not all the deadlines of sensor nodes can be satisfied. In this scenario, the deadlines are considered soft, and our algorithm design goal is to maximize the percentage of active nodes. Besides, we assume that the charging time is zero, meaning that the mobile charger finishes charging a node as soon as it arrives at its location. In this setting, we the focus of scheduler design is to find trade-offs between the spatial and temporal constraints. Our algorithms can be easily extended to constant non-zero charging times, by increasing the traveling time by a constant amount.

Traveling Salesman Problem is one of the most intensively studied problems in Computational Geometry [15]. The one variant that relates to our problem most closely is the Traveling Salesman Problem with Time Window (TSP-TW). In [16], the author summarizes the insertion and mutation heuristics to find schedules. In [7], Bar-Yehuda et al. firstly provide an $O(\log n)$ approximation algorithm in 1-d case, then design an algorithm that depends on how densely the vertices are located. In [5] and [17], the authors study the Max-Prize Path (Orienteering) problem, where the problem goal is to find a path that visits the maximum number of nodes before a common deadline. In [18], Bansal et al. design an $O(\log n)$ algorithm for the deadline-TSP, which is a more general problem than the orienteering problem, since each node has its own deadline.

However, the mobile charger scheduling problem is different from the TSP-TW. In the TSP-TW, all the deadlines are known and fixed before scheduling. Many existing approximation solutions rely heavily on this assumption [7, 5, 17, 18]. On the other hand, for the mobile charger scheduling problem, the future deadlines are not known as a priori. Before scheduling, the mobile charger only knows the current deadlines, which are the time moments when the sensors drain out their residual energy. However the next deadlines are determined only after the sensor nodes receive recharge. As a result, the solutions to TSP-TW cannot achieve the same performance bound when applied in our problem. In spite of that, the solutions to the TSP-TW problem provide valuable inspirations for us. We adopt the basic ideas of schedule insertion, density- and path-based target search in our SDT scheduler design.

III. Problem Description

A. Problem Formulation

Network Model: The wireless sensor network is modeled by a list of nodes $v_i \in V, i = 1, 2, ..., N$, where $N = |V|$. The network is deployed in a 2-D plain. The distances between any two nodes are given in the matrix D as a priori, and d_{ij} represents the traveling cost between the nodes v_i and v_j. There is a single mobile charger with unit traveling speed

traversing the network to conduct recharge. We use a vector $\mathbf{x}^k$ to represent the location of the mobile charger at the end of time interval k. $x_i^k = 1$ if the mobile charger recharges node v_i, and $x_j^k = 0$ for $j \neq i$.

The length of time interval k is represented by $\Delta t^k = \mathbf{x}^{k-1T} \cdot D \cdot \mathbf{x}^k$, which is the mobile charger's traveling time. Depending on the different distances between two nodes, the interval length Δt^k also changes. $k = 1, 2, ..., W$, where W represents the number of steps we conduct scheduling.

We use e_i^k to represent the residual energy storage of node i at the end of time interval k, which is calculated by $e_i^k = max(0, e_i^{k-1} - \Delta t^k r_i^k)$ when it isn't recharged. During the time interval k, we assume a node's energy consumption rate r_i^k to be a constant.

We adopt the assumption that the charging time is zero, so when the mobile charger arrives at a node v_i, it charges c_k units of energy to the node immediately, which is calculated by $c_k = E_{max} - e_i^k$. In sum, the residual energy e_i^k of a sensor node i at the end of time interval k is described using the following equation:

$$e_i^k = max(0, e_i^{k-1} - \mathbf{x}^{k-1T} \cdot D \cdot \mathbf{x}^k r_i^k) + x_i^k c_k. \quad (1)$$

Optimization Goal: A sensor network needs to maintain certain percentage of active nodes in order to achieve high level of monitoring. In this paper, our primary goal of algorithm design is to maximize the average coverage ratio of the sensor network over any given any scheduling window $[0, T_W]$. In other words, we continue scheduling until $\sum_{k=1}^{W} \Delta t^k \geq T_W$ and $\sum_{k=1}^{W-1} \Delta t^k < T_W$. The *coverage ratio* is defined as the percentage of nodes that have non-zero battery storage. Specifically, we calculate the coverage ratio at the end of each scheduling interval k, and calculate the overall average using the following equation:

$$B_C = \sum_{k=1}^{W} |\mathbf{e}^k|_0 / N. \quad (2)$$

In some cases, when the charging workload is small, the mobile charger is able to ensure 100% coverage ratio. Under this condition, the optimization goal will be to maximize the overall residual energy of the network, which is shown in the following equation:

$$B_E = \sum_{k=1}^{W} |\mathbf{e}^k|_1 / N. \quad (3)$$

Since our primary optimization goal is coverage ratio B_C, we will assign a small weight to the overall residual energy B_E. When the coverage ratio is maintained to be 100%, the overall residual energy gains more importance.

Mobile Charger Scheduling Problem: Given a sensor network V with parameter E_{max}, N, r_i^k, and D, and a single mobile charger with unit traveling speed, find a schedule of the mobile charger that maximizes the average coverage ratio over any given time window $[0, \sum_{k=1}^{W} \Delta t^k]$. The mobile charger scheduling problem is formulated as follows:

$$\begin{aligned} \max_{\mathbf{x}^k, c_k} \quad & = B_C + \beta B_E \\ & = \textstyle\sum_{k=1}^{W} (|\mathbf{e}^k|_0 + \beta |\mathbf{e}^k|_1)/N \\ s.t. \quad & e_i^0 = E_{max}, i = 1, 2, ...N \\ & e_i^k = max(0, e_i^{k-1} - \mathbf{x}^{k-1T} \cdot D \cdot \mathbf{x}^k r_i^k) + x_i^k c_k, \\ & i = 1, 2, ..., N, k = 1, 2, ..., W, \\ & 0 \leq e_i^k \leq E_{max}, i = 1, 2, ..., N, k = 1, 2, ..., W, \\ & \mathbf{1}_N^T \mathbf{x}^k = 1, k = 1, 2, ..., W \\ & x_i^k \in \{0, 1\}, i = 1, 2, ..., N, k = 1, 2, ..., W, \\ & 0 \leq c_k \leq E_{max}, k = 1, 2, ..., W. \end{aligned} \quad (4)$$

This is a problem of maximizing a nonlinear convex function over a convex set, with discrete constraint $x_i^k \in \{0, 1\}, \forall k$. So this problem cannot be solved using standard optimization techniques. Furthermore, due to the changes in environment in realistic applications, it will be difficult to have an accurate estimation of the energy consumption rates r_i^k. Therefore, the mobile charger needs to be able to adjust its schedules adaptively according to the network's needs in the run time. In this paper, we aim to design light-weighted algorithms that respond to the environment dynamics automatically and achieves good performance.

B. Problem Hardness

To prove the NP-completeness of the mobile charger scheduling problem, we reduce the Traveling Salesman Problem to it. The decision version of the mobile charger scheduling problem and the Traveling Salesman Problem are stated as follows.

Decision Version of the mobile charger scheduling problem: given a set of sensor nodes V deployed in a 2-D plain, each node $v_i \in V$ has battery capacity E_{max} and constant energy consumption rate r_i. At time zero, all the sensor nodes have full battery storage, and the mobile charger starts at node v_0. The question asks whether or not there exists a schedule such that the mobile charger can charge and maintain all the nodes alive during the time window $[0, T_W]$.

Decision Version of the Traveling Salesman Problem: Given a list of vertices V', a starting point $v_0' \in V'$, and a distance limit L, the question asks whether there exists a route no greater than L that visits each vertex exactly once.

Theorem 1. *The Decision Version of the mobile charger scheduling problem is NP-complete.*

Proof: Firstly, we show that this problem belongs to NP. Given any schedule, we calculate the times

of recharge $t_1, t_2, ...t_W$ for each a node i. We define a virtual recharge time $t_0 = (e_i^0 - E_{max})/r_i$. Then we proceed to compare the adjacent recharge gaps $t_k - t_{k-1}, k = 1, 2, ...W$ with the node's battery life E_{max}/r_i. If all the recharge gaps are smaller than the battery life, it means the node v_i will never run out of energy. We repeat this process for all nodes in V to determine whether the schedule can maintain 100% coverage ratio. This certifier has a polynomial-time computational complexity of $O(|V|W)$, so the mobile charger scheduling problem belongs to NP.

Then we conduct the reduction from the TSP problem. Given any instance of Traveling Salesman Problem V', L and v_0', we construct an instance of mobile charger scheduling problem V, r_i in the following way: Set $V = V'$, and the distances between nodes in V are set to be the same as those in V'. The energy consumption rates of all the nodes in V are set to be E_{max}/L, and the time window is set to $T_W = L$.

In what follows we show that the solutions to the two problems are equivalent in two steps. Step 1: If there exists a solution C' to the TSP problem instance V', L, we apply the same route C' to the mobile charger scheduling problem V. Following the schedule C', the mobile charger visits each sensor node exactly once during the scheduling time window, which takes time L. Since all the sensor nodes have the same battery life L, the schedule C' ensures that all the sensor nodes are recharged before their batteries drain out. In other words, C' is a feasible solution for the mobile charger scheduling problem V.

Step 2: If C is a feasible solution for the mobile charger scheduling problem V, we can see that each sensor node must be recharged at least once. Otherwise the coverage ratio will be smaller than 100%. Then we make a minor change to route C to make it a feasible solution to the TSP instance V'. We firstly scan the route C to see if any vertex is visited more than once. For any vertex v_i' that is visited before, we update the route C by making a short cut that connects its previous vertex v_i^p and next vertex v_i^n directly. According to the triangle inequality, this operation will not increase the length of C. We repeat this process to update C until no more duplicated vertices exist. This process has polynomial time complexity of $O(|V'|)$. In this way, we ensure that C visits all nodes in V' exactly once, and the total time spent on C is no more than L. Therefore C is a feasible solution to the TSP problem instance V'. This way we show that the mobile charger scheduling problem is NP-complete. ■

IV. Motivating Example

In this section, by testing the classical EDF and TSP schedulers using simple examples, we demonstrate that

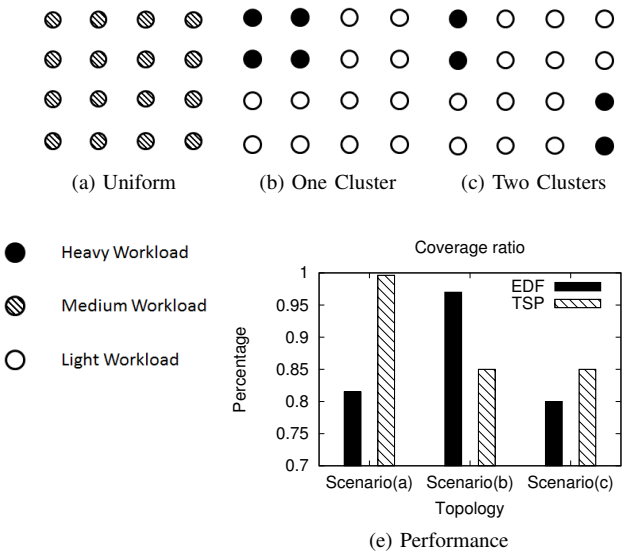


Figure 1: Motivating Examples

both spatial and temporal closeness must be considered in the mobile charger scheduling problem. Figure 1a, 1b, and 1c show three scenarios: (a) a network of nodes with the same energy consumption rate, (b) a network with a cluster of nodes that have high energy consumption rates, and (c) a network with two clusters of nodes that have high energy consumption rates. The result is shown in Figure 1e. The EDF and TSP schedulers are defined as follows:

Earliest Deadline First (EDF) scheduler: In the EDF[19] scheduler, the mobile charger always recharges the node v_i with the shortest deadline e_i^k/r_i^k.

Shortest Path (TSP) scheduler: The TSP Scheduler[20]firstly uses a heuristic to generates an approximately shortest path that visits all nodes, then the mobile charger follows this path to recharge each node it encounters.

Spatial Considerations: To achieve good scheduling performance, the scheduler needs to find short routes that reduce the traveling time and improves recharge efficiency. This is illustrated in Figure 1a, where all the sensor nodes have the same energy consumption rate. In this case, the TSP scheduler achieves 99% coverage ratio. However, if the EDF scheduler is applied instead, the average coverage ratio will be degraded to around 82%. This is because the mobile charger's schedule is driven purely by deadlines. It travels back and forth to save the dying nodes without considering the nodes' location information.

Temporal Considerations: The mobile charger scheduling problem is also a real time scheduling problem because each sensor node has its own deadline that needs to be satisfied. Therefore, a good scheduler needs to take into account the time constraints. As an example, we apply the EDF scheduler to the scenario

in Figure 1b. In this case, the mobile charger spends most of its time recharging nodes with high energy consumption rates, and achieves good performance. However, if we apply the TSP scheduler in this same network, the coverage ratio is reduced to around 84%. This is because it fails to take into account the different urgency of nodes.

In scenario 1c, which has two clusters of nodes with high energy consumption rates, neither scheduler performs well. When the EDF scheduler is applied, the mobile charger travels between the two clusters back and forth, which results in long traveling times. After some time, many low-power nodes in the network run out of battery, and a dramatic drop in coverage ratio occurs. On the other hand, when the TSP scheduler is applied, the high power nodes do not receive enough recharge, while the lower power nodes receive recharge more than needed. As a result, the coverage ratio fluctuates at a lower level. This example demonstrates that both spatial and temporal aspects of the network need to considered when designing a scheduler. Otherwise the recharge performance will be greatly compromised.

V. Algorithm Design

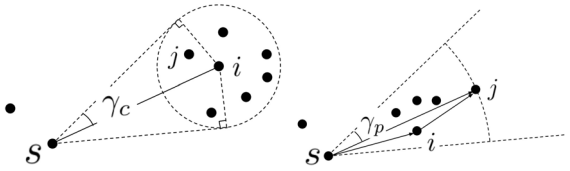
In this section we present the development of the innovative Spatial Dependent Task (SDT) Scheduler. We firstly quantify the mutual influence among spatially distributed recharge tasks using Cosine Rule. We discover that there exist two conditions, termed *Node Cluster* and *Nodes Near Path*, which can be used to improve scheduling performance. Based on these observations, we design the high performance Spatial Dependent Task (SDT) Scheduler.

A. Spatial Dependent Task Model

In the mobile charger scheduling problem, the processing times (traveling time) of different recharge tasks are mutually dependent, which distinguishes it from many other real time scheduling problems. We can quantify the mutual dependency of the spatially distributed recharge tasks. As we can see in Figure 2a, the traveling time for node v_j changes from d_{sj} to d_{ij} if the mobile charger moves from node v_s to v_i first. According to Cosine Rule, the value of d_{ij} is determined by the following equation:

$$d_{ij} = \sqrt{d_{si}^2 + d_{sj}^2 - 2d_{si}d_{sj}cos\gamma}. \tag{5}$$

Equation 5 can be applied to determine how the traveling times of the recharge tasks evolve as the mobile charger travels in the network. In what follows, we introduce the *node cluster* and *nodes near path* based on this equation.



(a) Cluster Dependency (b) Path Dependency

Figure 2: Task Spatial Dependency

1) Cluster Dependency: When the sensor nodes are densely distributed in a small area, recharging one node can reduce the traveling time of nearby nodes. For instance, in Figure 2a, if the mobile charger recharges v_i, then the traveling times of all other nodes around v_i will be greatly reduced as well. We take the node v_j in this figure as an example. We can see $\angle v_i v_s v_j \approx 0$ and $d_{sj} \approx d_{si}$. According to Equation 5, $d_{ij} \approx 0$. This means that the traveling time for node v_j will be decreased to near zero if the mobile charger recharges node v_i first.

Inspired by this observation, we propose that when looking for targets, the mobile charger should take into account the energy information of the nodes' neighbors. The closer two nodes are located to each other, the larger mutual influence they have. Specifically, the cluster priority of a node v_i is defined as follows:

$$P_{v_i}^c(k) = \frac{\sum_{v_j \in cluster(v_i)} w(j) * (E_{max} - e_j^k)}{\sum_{v_j \in cluster(v_i)} w(j)} / d_{si}^{\alpha}. \tag{6}$$

In this equation, $E_{max} - e_j^k$ is an estimation of the amount of energy node v_j can receive. This is then weighted by the term $w(j) = (d_{cluster} - d_{ij})/d_{cluster}$, which is an estimation of the mutual influence on traveling time for neighboring nodes. $cluster(v_i)$ is defined as the set of nodes located in the neighborhood of node v_i. In this paper, the neighborhood is defined as a circular disk with radius $d_{cluster}$ centered around v_i. Finally, the cluster priority is divided by d_{si}^{α}, which is to penalize clusters lying far away. We use an adjustable parameter α to determine how heavy the penalty is.

2) Path Dependency: When nodes are lying close to the mobile charger's traveling path, as shown in Figure 2b, these nodes can be inserted to the recharge schedule without incurring much extra overhead. For example, if the mobile charger travels from node v_s to v_j directly, the traveling time is d_{sj}. Instead, if the mobile charger recharges both node v_i and v_j, then the total traveling time is $d_{si}+d_{ij}$, which is approximately the same as d_{sj}. Specifically, according to Equation 5, when $\gamma_P = \angle v_s v_i v_j \approx \pi$, we have:

$$\begin{aligned} d_{sj} &= \sqrt{d_{si}^2 + d_{ij}^2 - 2d_{si}d_{ij}cos\gamma_P} \\ &\approx \sqrt{d_{si}^2 + d_{ij}^2 + 2d_{si}d_{ij}} \\ &= d_{si} + d_{ij}. \end{aligned} \tag{7}$$

This means node v_i can be recharged incidentally with small overhead when the mobile charger is traveling to node v_j. In order to quantify the benefits, we need to consider both the distances between nodes and their battery storage. We quantify the benefit of recharging a node v_j near the path as $E_{max} - e_j^k - \mathbf{x}^{k-1T} D \mathbf{x}^k |\mathbf{r}^k|_1$. In this equation, $E_{max} - e_j^k$ is the amount of energy that node v_j can receive. D is the node distance matrix, and $\mathbf{x}^{k-1T} D \mathbf{x}^k$ is the traveling time for node v_j. $|\mathbf{r}^k|_1$ represents energy consumption rate of the entire network. The physical interpretation of this function is the net increase of the energy of the network when the mobile charger charges node v_j. Based on this function, we can define the *path priority* and the mobile charger's *traveling graph*. Assume the mobile charger recharges nodes along the path $\mathbf{l} = [v_1^*, v_2^*, ..., v_n^*]$, and the distance between node v_i^* and v_j^* be $d_{v_i^*, v_j^*}$ then the priority of this path $\mathbf{l}$ is defined as follows:

$$P_{\mathbf{l}}^p(k) = \sum_{v_j \in \mathbf{l}} (E_{max} - e_j^k - d_{v_{j-1}^*, v_j^*} |\mathbf{r}^k|_1). \quad (8)$$

Given a target node v_t, our next step is to find a path for the mobile charger to travel. Our goal of path-finding is to maximize the benefit defined in Equation 8, while not incurring much detour from reaching v_t. To achieve this goal, we define the *traveling graph* that the mobile charger can travel so that length of detours can be reduced. Specifically, assume the mobile charger is located at node v_s and has a schedule to recharge node v_t. A directed edge $edge(v_i, v_j)$ exists from node v_i to v_j if and only if $\angle tij < \gamma_p$ and $d_{jt} < d_{it}$. Intuitively, this means the mobile charger can only travel to nodes lying in the sector area centered around the segment $\overline{v_i v_t}$, and has closer distance to the target node v_t. Using this definition, the *traveling graph* is a Directed Acyclic Graph (DAG). Finally, we assign the *path priority* defined in Equation 8 to each of these edges, and apply the critical path algorithm on this graph to find the path that has maximum overall *path priority*. Note that although polynomial-time longest path algorithm does not exist in general graphs, a linear time complexity longest path algorithm, which is based on Dijkstra's algorithm, is available for DAG [21].

B. Spatial Dependent Task Scheduler

Based on the *cluster priority* and *path priority*, we design the Spatial Dependent Task scheduler. It consists of two basic steps: 1) create a schedule using the *cluster priority*, and 2) optimize the schedule by selecting travelling path with highest *path priority*. Specifically, the SDT scheduler is defined in Algorithm 1.

The SDT scheduler firstly searches for the node v_t with highest *cluster priority* defined in Equation 6.

Algorithm 1 *Spatial Dependent Task Scheduler*

Input: Initial location of MC s
Output: schedule: $v_1, v_2 ... v_t$
1: **for all** $v_i \in V$ **do**
2: calculate $P_{v_i}^c(k)$
3: **end for**
4: $v_t \leftarrow arc \max_{v_i \in V} P_{v_i}^c(k)$
5: Update edges of travelling graph
6: Use Longest Path algorithm to find highest priority path $\mathbf{l}$ from v_s to v_t in V
7: **return** $[\mathbf{l}, v_t]$

After v_t is found, the edges of the *travelling graph* is updated, and priorities of the edges are computed according to the definition of *travelling graph* and *path priority* in Section V-A2. Finally, Critical Path Algorithm (longest path algorithm) is applied to search for the path that has the maximum *path priority* from the mobile charger's location v_s to the target node v_t. The mobile charger will follow this schedule to recharge sensor nodes. When it arrives at the node v_t, it will invoke the SDT scheduler again for the next schedule.

VI. Evaluation

A. Experiment Set-Up

In order to test the mobile charger schedulers, we implement a simulation framework, with the parameters collected from realistic systems. At the start of each scheduling interval, we randomly pick an consumption rate value from the history record of real sensors from [9]. We assume the energy consumption rates remain static during each scheduling interval. The average energy consumption rate is around $0.12W$. For high workload sensors, we scale the energy consumption rate to 6 times of normal ones. We assume the battery capacity is $10KJ$, and the mobile charger is moving at a constant speed of $0.35m/s$.

To achieve comprehensive evaluation, we test the scheduling algorithms under various different scenarios. Each network has 225 sensor nodes, and is deployed in grid topology. The areas of deployment regions range from $0.25km^2$ to $4km^2$. To simulate the performances under high workloads, we divide the network into 9 equal-size clusters, and assign from 0 to 4 clusters of nodes to have high workloads. In each experiment, we conduct $W = 10000$ times of scheduling. The coverage ratio and other evaluation metrics are recorded at the end of each scheduling interval, which are used for the calculation of the statistic values like average and standard deviation of each experiment. Each experiment is repeated three

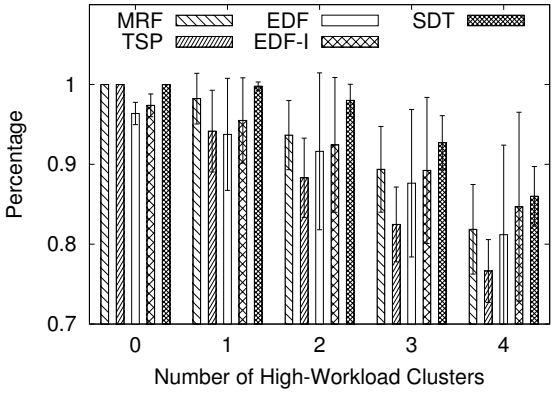


Figure 3: The Influence of High-Workload Clusters

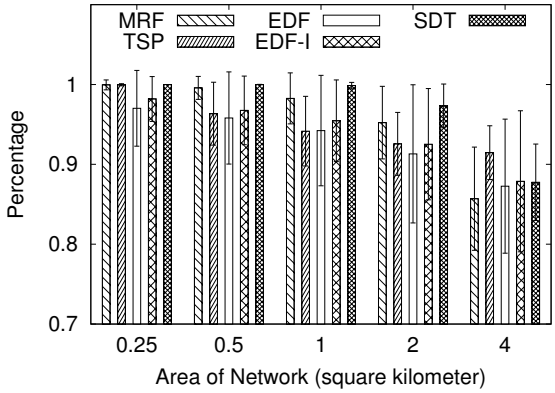


Figure 4: The Influence of Network Scale

times to derive the average of these statistical metrics in order to reduce the effects of random cases. The parameters of the SDT algorithm are listed as follows: $d_{cluster} = 13.125m$, $\alpha = 0.5$, and $\gamma_p = \pi/4$.

B. Baseline Algorithms

Nearest Insertion The Nearest Insertion Algorithm is applied to improve the EDF scheduler, which is termed EDF-I. Specifically, assume the mobile charger is located at node v_s and the target node found by EDF is v_i. Then EDF-I searches the node v_j that minimizes the detour $W_j = d_{sj}+d_{ji}$. If W_j is smaller than the life time of v_i, then the node v_j will be recharged before v_i. Otherwise the mobile charger will not insert any node.

Maximum Response ratio First (MRF) In the MRF scheduler [19], we define a node v_i's priority using the response ratio $W_i = wt_i^k/d_{si}$, where the waiting time wt_i^k is defined as the length of time since the previous recharge.

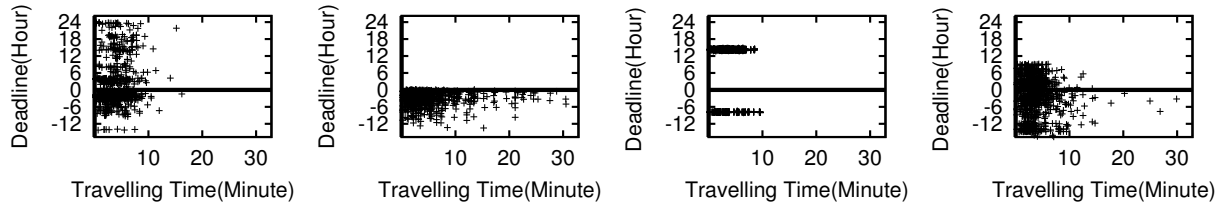
C. Coverage Ratio

1) The Influence of Workload: We firstly evaluate the algorithm performance when the networks are deployed in an area of $1km^2$, with from 0 to 4 clusters of heavy-workload nodes. The results are shown in Figure 3. When the number of clusters increases, we can easily find that TSP scheduler's performance degrades seriously, from 100% of coverage ratio to 76%. The reason is that TSP scheduler recharges all sensor nodes at the same frequency, regardless of the fact that they have different energy consumption rates. When all the nodes have the same energy consumption rate i.e., the case with zero cluster of heavy-workload nodes, TSP achieves good performance because it reduces the overall traveling time. However, when the nodes have increasingly polarized energy consumption rates, i.e., the case with four clusters, more and more nodes fail to receive sufficient energy and thus run out of energy, while others are recharged more than needed. As a result, TSP's performance drops dramatically. On the other hand, we can see that when the number of clusters ranges from 2 to 4, SDT scheduler outperform TSP by 10%. This shows that when the workloads in certain areas of the network increase, SDT scheduler is able to adjust schedules accordingly, and spends most of the time on the nodes that have heavy workloads. As a result, the coverage ratio is improved.

We can also see that the standard deviations of coverage ratio of MRF, EDF and EDF-I are large. For example, when the network has 1 cluster of high power clusters, MRF, EDF and EDF-I have standard deviations of 4%, 9% and 8%, respectively. This shows that during the recharge process, the coverage ratio is fluctuating. On the other hand, the SDT scheduler achieves only 2% standard deviation, which indicates much better stability in coverage ratio. This reveals the drawbacks of applying greedy-search-based schedulers to the mobile charger scheduling problem: The mobile charger is driven by the local optimum of its goal, while ignoring the other areas of the network. For example, when EDF is applied, it will spend most of its time recharging nodes with high power consumptions, until a large number of lower-power nodes begin to die, which results in a large drop in coverage ratio.

2) The Influence of Network Scale: As the network scale increases, the time it takes for the mobile charger to travel between sensor nodes also increases. In this experiment, we compare the algorithm performance under different network scales from $0.25km^2$ to $4km^2$. We can see that SDT achieves over 97% coverage ratio when the network scale ranges from $0.25km^2$ to $2km^2$, which outperforms all other algorithms. It shows that SDT maintains good performance gracefully under cases that require more traveling time.

When the sensors are deployed in an area of $4km^2$, then the TSP scheduler achieves the best performance among all, which is 91%. This is because when the



(a) SDT (b) EDF (c) TSP (d) MRF

Figure 5: Relation between Deadline and Traveling Distance

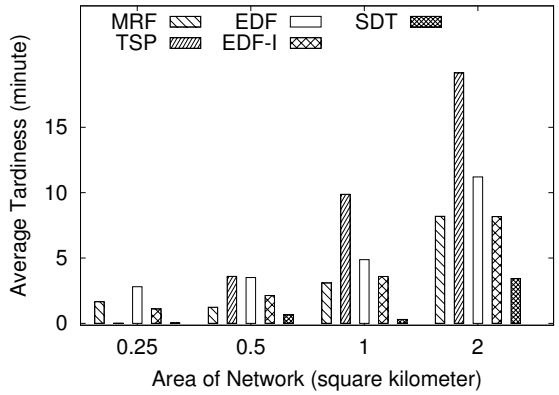


Figure 6: Average Tardiness

TSP scheduler is traveling along efficient routes, which reduces the influence of the increase of traveling distance. This benefit is more evident when the network scale grows large.

We can also see that the Insertion heuristic provides a valuable improvement on scheduling performance. EDF-I has a better performance than EDF when the network scale is from $0.25km^2$ to $2km^2$. When the network scale is $4km^2$, they have similar performance, at 87% of coverage ratio. The reason is that the insertion heuristic is able to improve the recharge efficiency by recharging a node lying close to the mobile charger's existing schedule. However, the costs of detours incurred by insertion are not negligible in large scale networks. Therefore, we can observe performance degradation in such cases.

3) Relation Between Spatial and Temporal Closeness: To further explore the differences among these scheduling algorithms, we deploy sensor nodes randomly within an area of $1km^2$, with 2 clusters of heavy-workload clusters. We plot the relations between deadlines and processing times for SDT, EDF, TSP and MRF in Figure 5. In these figures, each point represents the traveling time and deadline of a node when it is being recharged. We use negative deadlines to indicate that the sensor nodes are already our of battery when it is recharged, and the absolute values of such negative deadlines represent the length of tardiness.

From Figure 5a, we can see that SDT achieves a trade off between processing time and deadline. Due to the application of *cluster priority*, the mobile charger sometimes takes long trips to rescue urgent nodes, while for most of the time it recharges nearby nodes so that the travelling cost can be minimized. Besides, we can also see that the traveling time of SDT is short, which is similar to that of TSP. The reason is that SDT applies *path priority* to optimize its schedules. Using *path priority*, SDT automatically searches for nodes that can be recharged without incurring much extra traveling overhead. As a result, the traveling efficiency is improved.

From Figure 5b, we can see that the EDF scheduler frequently takes long trips to rescue urgent nodes, because it focuses much on deadlines of nodes while ignoring the node distances. EDF-I improves the performance slightly by inserting nodes near the path. However, since it inserts at most one node in each trip, the performance improvement is not significant. On the other hand, SDT inserts large amounts of nodes into its schedulers so that it significantly reduces inefficient trips.

From Figure 5c, we can see that the nodes scheduled by TSP are clustered into two groups: one group of nodes have large negative deadlines, and the other large positive ones. Although TSP ensures short processing times for all the nodes, a large number of nodes are recharged after they have been starved for about 5 hours. The reason is that TSP scheduler fails to respond to these nodes' heavy workloads quickly enough. Instead, it spreads its charging power equally among all the nodes, regardless of their different energy consumption rates.

D. Average Tardiness

In order to evaluate the real-time performance of the schedulers, we apply the average tardiness as a metric. We use t_i^m to denote the moment when node i's residual energy level drops to zero. Then the tardiness $T_i(t)$ of a node i is equal to $t - t_i^m$ if $e_i^k = 0$, and equal to 0 otherwise. We compute the average tardiness over all the sensor nodes and over time. In Figure 6, we show the results of average tardiness for the schedulers

when network scale ranges from $0.25km^2$ to $2km^2$, with two clusters of heavy workload clusters.

In this figure we can see that the SDT scheduler achieves the lowest tardiness among all the schedulers. When the network scale is $0.5km^2$, $1km^2$ and $2km^2$, the SDT scheduler reduces at least 85% of the tardiness of the TSP scheduler. This demonstrates that compared with TSP, SDT responds to the urgent nodes much more quickly.

We can also see TSP scheduler's average tardiness is also significantly larger than MRF, EDF and SDT, especially when the network scale is 1 or 2 km^2. This is because when using TSP under these cases, the nodes with heavy workloads are receiving far less energy than needed. As a result, they die out quickly and receive insufficient recharges, which creates large tardiness.

Interestingly, the MRF scheduler results in large average tardiness in our experiments, although in real-time processor scheduling problem, the MRF scheduler is supposed to eliminate process starvation [19]. When the network is deployed in $0.25km^2$, it has 2 minutes of average tardiness, which is higher than all other schedulers except TSP. The reason is that MRF focuses too much time recharging nearby nodes. From Figure 5d, we can see MRF scheduler spends most of its time recharging nearby nodes, until some faraway urgent nodes force it to take long trips to rescue long starved nodes. Therefore, we can see that although MRF reduces tardiness in a single machine, it is not necessarily able to reduce tardiness under the case that tasks are distributed in space.

In summary, our simulations show that the proposed algorithm SDT performs well in terms of both coverage ratio and average tardiness under a variety of settings. When the recharge workloads are light, TSP achieves good performance. However its performance degrades quickly when the the number of high-power nodes increases. The EDF scheduler suffers from large standard deviation in coverage ratio and tardiness. From the experiment results of EDF-I, we can see that the Insertion heuristic is able to slightly improve the scheduling performance in small scale networks. MRF scheduler has satisfactory performance in coverage ratio, but it is unable to reduce average tardiness.

VII. Conclusion

In this paper, we formulate the mobile charger scheduling problem as an optimization problem. Our primary goal is to maximize the percentage of nodes that are alive for monitoring purposes. We proved that it is NP-complete by reducing it to the Traveling Salesman Problem. In algorithm design, we focus on the case when the traveling time is larger than some of the battery lives of sensor nodes. we propose a spatial dependent task scheduling algorithm, which quantifies the impact of scheduling proximate tasks on the other tasks. With extensive simulations that cover different network scales and workloads, we demonstrate that our algorithms have good performance. Our solution outperforms the classical such as the TSP scheduler by up to 10% and 85% in terms coverage ratio and tardiness, respectively.

References

[1] G. Xing, J. Wang, K. Shen, Q. Huang, X. Jia, and H. C. So, "Mobility-assisted spatiotemporal detection in wireless sensor networks," in *Proceedings of ICDCS*. IEEE, 2008.

[2] L. Luo, C. Huang, T. Abdelzaher, and J. Stankovic, "Envirostore: A cooperative storage system for disconnected operation in sensor networks," in *Proceedings of INFOCOM*. IEEE, 2007.

[3] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines," in *Real-Time Systems Symposium, 2004. Proceedings. 25th IEEE International*. IEEE, 2004, pp. 296–305.

[4] Y. Hu, X. Wang, and X. Gan, "Critical sensing range for mobile heterogeneous camera sensor networks."

[5] A. Blum, S. Chawla, D. R. Karger, T. Lane, A. Meyerson, and M. Minkoff, "Approximation algorithms for orienteering and discounted-reward tsp," *SIAM Journal on Computing*, vol. 37, no. 2, pp. 653–670, 2007.

[6] A. Mingozzi, L. Bianco, and S. Ricciardelli, "Dynamic programming strategies for the traveling salesman problem with time window and precedence constraints," *Operations Research*, 1997.

[7] R. Bar-Yehuda, G. Even, and S. M. Shahar, "On approximating a geometric prize-collecting traveling salesman problem with time windows," *Journal of Algorithms*, vol. 55, no. 1, pp. 76–92, 2005.

[8] Y. Dumas, J. Desrosiers, E. Gelinas, and M. M. Solomon, "An optimal algorithm for the traveling salesman problem with time windows," *Operations research*, 1995.

[9] T. Zhu, Z. Zhong, Y. Gu, T. He, and Z.-L. Zhang, "Leakage-aware energy synchronization for wireless sensor networks," in *Proceedings of MobiSys*. ACM, 2009.

[10] S. Zhang, J. Wu, and S. Lu, "Collaborative mobile charging for sensor networks," in *Proceedings of MASS*. IEEE, 2012.

[11] Y. Peng, Z. Li, W. Zhang, and D. Qiao, "Prolonging sensor network lifetime through wireless charging," in *Proceedings of RTSS*. IEEE, 2010.

[12] L. Xie, Y. Shi, Y. T. Hou, W. Lou, and H. D. Sherali, "On traveling path and related problems for a mobile station in a rechargeable sensor network," in *Proceedings of MobiHoc*. ACM, 2013.

[13] L. He, L. Fu, L. Zheng, Y. Gu, P. Cheng, J. Chen, and J. Pan, "Esync: An energy synchronized charging protocol for rechargeable wireless sensor networks," in *Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2014, pp. 247–256.

[14] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling with dynamic deadlines," *Mobile Computing, IEEE Transactions on*, vol. 6, no. 4, pp. 395–410, 2007.

[15] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, 1992.

[16] H. N. Psaraftis, "Dynamic vehicle routing problems," *Vehicle routing: Methods and studies*, 1988.

[17] C. Chekuri, N. Korula, and M. Pál, "Improved algorithms for orienteering and related problems," *ACM Transactions on Algorithms (TALG)*, vol. 8, no. 3, p. 23, 2012.

[18] N. Bansal, A. Blum, S. Chawla, and A. Meyerson, "Approximation algorithms for deadline-tsp and vehicle routing with time-windows," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. ACM, 2004, pp. 166–174.

[19] P. Brucker, *Scheduling algorithms*. Springer, 2007.

[20] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem." DTIC Document, Tech. Rep., 1976.

[21] J. E. Kelley Jr and M. R. Walker, "Critical-path planning and scheduling," in *Papers presented at the December 1-3, 1959, eastern joint IRE-AIEE-ACM computer conference*. ACM, 1959, pp. 160–173.