Recharging Schedules for Wireless Sensor Networks with Vehicle Movement Costs and Capacity Constraints

Cong Wang[1], Ji Li[1], Fan Ye[2], and Yuanyuan Yang [1]

[1]*Dept. of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794, USA*
[2]*School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, P.R. China*

***Abstract*—Several recent works have studied the schedule for mobile vehicles to recharge sensor nodes via wireless energy transfer technologies. Unfortunately, most of them overlooked the important factors of the vehicles' moving energy consumption and limited recharging capacity. These oversights may lead to problematic schedules or even stranded vehicles. In this paper, we study the recharging schedule that maximizes the recharging profit - the amount of replenished energy less the cost of vehicle movements - under these important constraints. We first derive the minimum number of vehicles needed for energy neutral condition and discover a set of desired network properties. Then we formulate the recharge schedule optimization into a Profitable Traveling Salesmen Problem with capacity and battery deadline constraints, which we prove to be NP-hard. We propose two algorithms to solve the problem. The first one is a greedy algorithm that maximizes the recharge profit at each step; the second one first adaptively partitions the network based on recharge requests, then forms Capacitated Minimum Spanning Tree in each partition followed by route improvements. Finally, we evaluate and compare the performance of proposed algorithms and validate the correctness of theoretical results through extensive simulations. Given a sufficient number of vehicles, the adaptive algorithm can keep the number of nonfunctional nodes at zero. Compared to the greedy algorithm, it reduces the percentage of transient energy depletion by 30-50% with 10-20% energy saving on vehicles.**

***Index Terms*—Wireless rechargeable sensor networks, perpetual operations, data collection, adaptive network partitioning, vehicle scheduling.**

I. Introduction

Wireless energy transfer is a revolutionary method to power sensor nodes and such sensor networks are referred to as *Wireless Rechargeable Sensor Networks (WRSNs)* [1]–[4]. Unlike energy harvesting techniques where the effectiveness of the harvesting method depends on uncertain environmental factors, wireless energy transfer provides a reliable way to rejuvenate nodes. Ideally, the lifetime of a WRSN can be extended to infinitely long, or *perpetual operations*. Charging vehicles (called *SenCar*s) that can approach sensors in close proximity are usually adopted [2]–[4]. To ensure efficient operations, a recharge sequence is calculated such that sensor nodes are recharged before energy depletion [2]–[4]. In [2], a heuristic algorithm that finds the maximum number of nodes to recharge within a bounded tour length was proposed. In [3], the shortest recharge path is found for recharging all the sensor nodes. In [4], an on-line algorithm was provided to schedule multiple SenCars based on real-time energy information gathered.

However, most of the previous works have ignored the energy consumption of the SenCar itself for moving and the limit of its charging capacity. The simplifications bring problems when existing algorithms are applied in reality. First, they may cause impractical schedules where SenCars deplete their energy, become stranded and unable to return to the base station. The network would eventually use up energy and stop operation completely. Second, they lead to overestimation of SenCar's recharge capability and nodes' lifetimes. Real SenCars have limited capacity. They have to spend time returning to the base station for battery replacement and cannot keep recharging nodes continuously. Third, they may also result in inefficient node selection and recharge sequences. Without considerations on SenCars' capacity and moving costs, one may choose nodes faraway to recharge simply because they have lower energy levels. But this can cause SenCars to travel back and forth over long distances and waste significant amount of energy.

In this paper, we study the recharge schedule for SenCars in a practical model where vehicles have limited capacity and their movements consume energy. We propose to maximize the *recharge profit*, the recharged energy less the traveling cost, while meeting sensor nodes' battery deadlines and SenCars' capacity constraints. Considering SenCar's moving cost and capacity brings us new challenges. On one hand, recharging nodes close to a SenCar reduces its moving cost. On the other hand, nodes all over the network, not only those close by, need recharge once in a while. We have to achieve a balance between the need to recharge the whole network and the desire to minimize the traveling cost. In particular, we need to answer the following questions: How many SenCars are needed for perpetual operation given parameters such as a network's size and nodes' battery capacity? How to schedule SenCars so they will not waste energy traveling back and forth over long distances? Which nodes a SenCar should select to recharge while ensuring it has enough energy to return, and in which sequence so as to meet nodes' battery deadlines?

To answer these questions, we first establish a mathematical model based on the energy neutral conditions where the energy consumed by and replenished into the network achieve balance. We adopt results from probability theory to derive the minimum number of SenCars needed. Then, given recharge requests, we formulate the recharge optimization problem into a Profitable Traveling Salesmen Problem with Capacity and Battery Deadline constraints which was studied before but only has computationally intensive solutions.

We propose two algorithms suitable to the context of our problem. The first is a simple *Greedy Algorithm* that maximizes a SenCar's recharging profit at each step. However, it may lead SenCars to travel long distances. We further propose a three-step *Adaptive Algorithm*. After collecting recharge requests, it partitions the network into several regions using the K-means algorithm [14]. Each SenCar is assigned a region and its movements are confined within the region, so long-distance travels are avoided. Then each SenCar works independently to construct Capacitated Minimum Spanning Trees in its designated region where edges in the tree have the minimum traveling cost. This ensures that the SenCar's capacity is not exceeded so it can return to its starting position. Finally, the algorithm performs route improvements to meet nodes' battery deadlines. It categorizes nodes according to their lifetimes. An initial route containing nodes that do not need prioritized recharge is first constructed using Traveling Salesmen Problem algorithms. Then it inserts nodes that need prioritized recharge into the route while ensuring each insertion retains time feasibility of the whole recharge sequence.

We make several contributions in this paper. First, we point out the oversight of existing works on important constraints of SenCars' moving cost and limited capacity, and its impact on existing recharge algorithms. We establish a mathematical model to characterize energy consumptions in a practical network and derive the minimum number of charging vehicles needed. We also present several theoretical results such as minimum node lifetime and adaptive recharge threshold. Second, we formulate recharge optimization into a Profitable Traveling Salesmen Problem with Capacity and Battery Deadline constraints, and propose two algorithms. The Adaptive Algorithm takes a systematic approach to capture the constraints in the problem. Finally, we conduct extensive simulations comparing the performance of the proposed algorithms. The results have shown that when the number of SenCars is sufficient, the Adaptive Algorithm can keep all the nodes alive at all times. Compared to the Greedy Algorithm, the Adaptive Algorithm can also reduce nonfunctional nodes by 30-50% while saving 10-20% energy on the SenCars. We also validate our theoretical results and demonstrate the trade-off between minimizing SenCars' energy cost and improving network performance. To the best of our knowledge, this is the first work to explore optimal recharge sequences when both SenCars' energy and dynamic sensor battery deadlines are considered. This is also the first work that provides a mathematical model to calculate the minimum number of charging vehicles in a network where detailed sensing and communication energy consumption is modeled.

The rest of the paper is organized as follows. Section II outlines the framework, network components and assumptions. Section III establishes a mathematical model and derives a set of theoretical results. Section IV formalizes the recharge optimization problem and proposes two algorithms. Finally, Section V provides the evaluation results and Section VI concludes the paper.

II. Preliminaries

In this section, we present an overview of the components, network model and assumptions.

A. Network Components

Fig. 1 gives an illustration of the network we consider. *SenCars* perform wireless energy replenishment for sensor nodes one after another following a recharge sequence. Sensory data is generated at nodes and delivered to the *Base Station* in a multi-hop fashion. The base station also collects energy information periodically and it is where battery replacement for the SenCars is conducted. In the Adaptive Algorithm, the base station performs network partition and informs the SenCars of recharge requests in their partitions using long range radios. Besides data collection, sensors also monitor a number of targets that appear randomly in the sensing field, stay at a location for a random time before disappearing (e.g., meteorologic phenomena such as lightening).

B. Network Model and Assumptions

We assume that N_s sensor nodes are uniformly randomly scattered in a circular sensing field with radius r. Node density

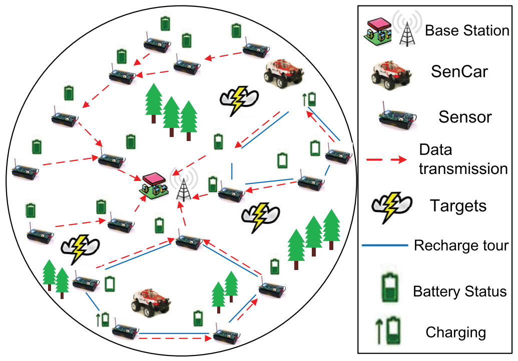


Fig. 1. Illustration of the network architecture and components.

TABLE I
Notations

Notation	Definition
N_s	the number of sensor nodes
N_t	the number of target nodes
r	radius of the circular sensing field
λ	traffic rate of each node
d_r	transmission range of sensor nodes
C_b	battery capacity of sensor node
C_s	recharge capacity of a SenCar
t_r	maximum recharge time of a node
v	speed of SenCars
e_s	energy consumed due to sensing per time slot
e_t, e_r	energy consumed to transmit, receive a data message

of the network is $\rho = \frac{N_s}{\pi r^2}$. The base station is deployed at the center of the circle. Time is equally slotted. In the first time slot, N_t targets appear independently at random locations in the field. For the next time slot, the target either stays at the location with probability p_s or appears at a new location with probability $1 - p_s$. Each node has the same sensing range d_s. For simplicity, in this work we do not consider dynamic sensor activation/deactivation schemes that exploit spatial diversity. We denote p as the probability that a randomly chosen point is within a given sensor's sensing range. Hence, $p = \frac{d_s^2}{r^2}$.

Energy is consumed when nodes sense targets or transmit/receive data messages. When targets are in a sensor node's sensing range, it consumes e_s energy in each time slot to detect the targets. Sensor nodes perform basic functions to capture environmental data in each time slot and generate data messages. They have constant data generation rate λ. All sensors transmit at the same power level with fixed transmission range d_r. The energy consumed for transmitting and receiving a data message is denoted as e_t and e_r, respectively. Energy information is collected periodically by the base station, using communication protocols such as the method developed in [4]. Since the size of energy information is much smaller than data messages and it is collected less frequently, we mainly consider energy consumptions due to transmission of data messages. [1]

There are m SenCars with recharge capacity C_s and they consume energy while moving around at a rate of e_c J/m. To model the relation between battery capacity C_b and total recharge time t_r, we use recharge curves of a Panasonic Ni-MH AAA batteries with 780 mAh from available data sheets [5]. Table I summarizes major notations and their corresponding definitions in this paper.

[1]Energy information is usually a few bits and gathered every 3-6 hours compared to data messages more than tens of bytes generated at minute-level time slots.

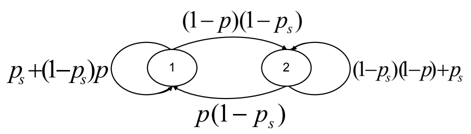


Fig. 2. State transition probabilities to detect a given target in a time slot.

III. Theoretical Analysis

In this section, we analyze the minimum number of SenCars required for perpetual operation, and derive minimum node lifetimes and adaptive recharge thresholds for nodes in the network.

A. The Minimum Number of SenCars

In an earlier work [4], we have proposed the energy neutral condition that must hold in a long time period for the perpetual operation of the network,

$$E(T) \leq R(T) + E_0 \tag{1}$$

in which T is a large time, $E(T)$ is the total energy consumption of the network up to T, $R(T)$ is the total energy replenished into the network by the SenCars up to T and E_0 is the initial energy of all the sensor nodes. The energy neutral condition states that the energy consumption of all the sensor nodes must be less than or equal to the total energy available in long term. Otherwise, sensor nodes would eventually deplete energy.

Our objective is to obtain the minimum number of SenCars m needed for Eq. (1) to hold. First, we estimate the average of $R(T)$ which is the amount of energy that can be replenished into the network. The maximum recharge capacity of a SenCar is achieved when it recharges sensor nodes continuously without any idling time. To model the moving time between two consecutive locations, we find the average traveling distance between two random locations in the circular field of radius r, which is shown [6] to be $\overline{d} = \frac{128}{45\pi} r \approx 0.9r$ using Crofton's mean value theorem.

Therefore, to recharge a node to full capacity, on average, it takes $\overline{d}/v + t_r$ time. Then the total energy for the m SenCars to replenish into the network in T time is,

$$R(T) = \frac{mC_bT}{\overline{d}/v + t_r}. \tag{2}$$

Next, let us derive $E(T)$ on the left hand side of Eq. (1), which is determined by sensing, transmission and reception activities. The transmission/receiption energy depends on data generation rate and sensing energy depends on the spatial distribution of targets and sensing range. Since sensing energy is consumed when there exists any target in the sensing range, we first calculate the probability that at least one target is detected in a time slot.

Given a sensor node, let P_1 and P_2 denote the probabilities that a certain target is *in* and *out* of its sensing range for a time slot, respectively. The state transitions between P_1 and P_2 can be modeled as a Markov chain shown in Fig. 2. Using the equilibrium conditions on the Markov chain, we can easily derive $P_1 = \frac{p-p_sp}{1-p_s} = p$, and $P_2 = \frac{1-p_s-p+p_sp}{1-p_s} = 1-p$. Since there are N_t targets, the probability that there is at least one target in a node's sensing range is,

$$P_t = 1-(1-P_1)^{N_t} = 1-(1-p)^{N_t} \tag{3}$$

For N_s sensor nodes, the probability that there are k sensor nodes detecting the target follows a binomial distribution $\sim B(N_s, P_t)$. Thus for time period T, the average energy consumption due to sensing can be written as,

$$\overline{E_s(T)} = e_sN_sTP_t = e_sN_sT(1-(1-p)^{N_t}) \tag{4}$$

Next, let us estimate energy consumption due to transmission/reception of data messages. As studied in [7], a network with h hops can be closely approximated by a circle with radius $r = hd_r$ with h concentric rings. Based on the structure of the network, each ring carries traffic from all outer rings [7]. Since nodes are uniformly randomly distributed, we can obtain energy consumption rate at the i-th ring,

$$\mu_i = \big((h^2-i^2)(e_t+e_r) + (2i-1)e_t\big)d_r^2\pi\rho\lambda, 0 < i \leq h \tag{5}$$

To elaborate the above formula, nodes in the outmost ring ($i = h$) only need to transmit their own data whereas nodes at inner rings need to forward all traffic from outer rings and transmit their own data messages. Using (5), we can calculate the total energy consumption due to data collection in time T,

$$\begin{aligned} E_d(T) \quad &= (\textstyle\sum_{i=1}^{h}\mu_i)T = \big(\sum_{i=1}^{h-1}[(h^2-i^2)(e_t+e_r) \\ &\quad +(2i-1)e_t]d_r^2\pi\rho + (2h-1)e_t)d_r^2\pi\rho\big)\lambda T \\ &= \big((\tfrac{2}{3}h^3 - \tfrac{1}{2}h^2 - \tfrac{1}{6}h)(e_t+e_r) + h^2e_t\big)d_r^2\pi\rho\lambda T \end{aligned} \tag{6}$$

Summing up Eq. (4) and Eq. (6), we obtain the average value of total energy consumption in T,

$$\overline{E(T)} = \overline{E_s(T)} + E_d(T) \tag{7}$$

Lemma 1: The probability for the energy neutral condition to hold is,

$$P_{neu} = \Phi\left(\frac{\frac{mCT}{\overline{d}/v+t_r} + E_0 - \overline{E(T)}}{\sqrt{e_sN_sTP_t(1-P_t)}}\right) \tag{8}$$

where $\Phi(\cdot)$ denotes the standard Normal cumulative distribution function, $\overline{E(T)}$ is obtained from Eq. (7) and E_0 is the initial energy of the network.

Proof: In $E(T) = E_s(T) + E_d(T)$, only $E_s(T)$ is a binomial random variable with mean $e_sN_sTP_t$ and variance $e_sN_sTP_t(1-P_t)$, and $E_d(T)$ is deterministic. Since we evaluate the energy neutral condition in a long term, T is a large number, we can approximate binomial distribution closely with a normal distribution $\mathcal{N}(e_sN_sTP_t + E_d(T), e_sN_sTP_t(1-P_t))$ [9]. Thus the probability for the energy neutral condition to hold is $P_{neu} = Pr\{R(T) + E_0 > E(T)\}$. ■

Proposition 1: The minimum number of SenCars required to achieve perpetual operation is,

$$m = \left\lceil \frac{(\Phi^{-1}(\epsilon)\sqrt{e_sN_sTP_t(1-P_t)} + \overline{E(T)} - E_0)(\overline{d}/v + t_r)}{C_bT} \right\rceil \tag{9}$$

where $\Phi^{-1}(\epsilon)$ is the inverse cumulative distribution function of Normal distribution, ϵ is a value very close to 1 but not equal to 1, $\overline{E(T)}$ is obtained from Eq. (7) and E_0 is the initial energy of the network.

Proof: Because $\Phi^{-1}(1) \to \infty$, we consider the energy neutral condition holds with probability ϵ, a value very close to 1 but not equal to 1. Take $\epsilon = 0.99$ as an example. To ensure $P_{neu} \geq 0.99$, since $\Phi^{-1}(0.99) = 2.33$, we have $\frac{\frac{mCT}{\overline{d}/v+t_r} + E_0 - \overline{E(T)}}{\sqrt{e_sN_sTP_t(1-P_t)}} \geq \Phi^{-1}(0.99)$. After some mathematical manipulations, we obtain the minimum number of SenCars m. ■

B. Node Lifetime and Adaptive Thresholding

To devise the recharge schedule, we need to know how long a sensor node can survive after it has requested for recharge. Such information is vital in making recharge decisions in the next section. Since a node's energy consumption rate depends on both sensing activities and traffic patterns, it is important for each node to know its number of hops to the base station. This information can be obtained by message propagation from the base station in various routing protocols. To estimate node lifetime we have the following lemma:

Lemma 2: Given a sensor node i that is j hops away from the base station with energy $E_{i|j}$, its minimum residual lifetime $L_{i|j}$ is,

$$L_{i|j} = \begin{cases} E_{i|j}/(e_s + e_t\lambda), j = h \\ E_{i|j}/(e_s + (\frac{(h^2-j^2)(e_t+e_r)}{2i-1} + e_t)\lambda), \\ 0 < j < h \end{cases} \tag{10}$$

Proof: First, the energy consumption due to transmission is deterministic for each time slot. In the outmost ring ($j = h$), nodes only need to transmit their own data messages. Therefore, the maximum energy consumption in a time slot is $e_s + e_t\lambda$. On the other hand, nodes in the inner rings need to forward traffic from all outer rings and their own traffic towards the base station. From Eq. 5, we can get the maximum energy consumption per node in a time slot, $e_s + ((h^2 - j^2)(e_t + e_r)/(2i - 1) + e_t)$. Therefore, we obtain the minimum residual lifetime $L_{i|j}$. ■

Then we have the following proposition.

Proposition 2: Given a set of nodes that need recharge $\mathcal{N}_r = \{1, , \ldots, n_r\}$ where n_r is the number of nodes in the set. For a sensor node $i \in \mathcal{N}_r$, if $L_{i|j} \geq n_r t_r + (n_r - 1)(2r/v)$, no matter where the node is placed in the recharge sequence, it will not deplete battery energy before its recharging starts.

Proof: The worst case occurs when the node is placed at the end of the recharge sequence. The maximum waiting time to get recharged is $(n_r - 1)(t_r + 2r/v)$ since there are $n_r - 1$ nodes ahead. $2r/v$ is the maximum traveling time between two sensor nodes since $2r$ is the diameter of the field. Since the minimum lifetime $L_{i|j} \geq n_r t_r + (n_r - 1)(2r/v) > (n_r - 1)t_r + (n_r - 1)(2r/v)$, it is guaranteed to recharge the node before it depletes battery energy. ■

From *Proposition 2*, given a recharge sequence, we can calculate the possibility that a node can survive the entire recharge process. This lays the theoretical foundations to solve the recharge optimization problem in the next section.

We have observed that due to higher data traffic for nodes in inner rings, they consume more energy than those in outer rings. A fixed recharge threshold treats nodes at different rings the same way, thus it is not fair for nodes with higher consumption rates. To recharge these nodes more frequently, the recharge thresholds should be made adaptive and proportional to energy consumption rates at different rings. In other words, nodes closer to the base station should request recharge more frequently than others.

Let $\tau_i (0 < \tau_i < 1)$ denote the recharge thresholds for nodes in the i-th ring. Since targets appear in the field uniformly randomly, the differences of energy consumptions between nodes at different locations is mainly caused by data communications. We make the ratio between the recharge thresholds of ring i and j equal to that between their energy consumptions due to data transmission. Assume we have set the recharge threshold of the first ring to be τ_1. The thresholds for other rings

$$\tau_i = \frac{(h^2 - i^2)(e_t + e_r) + e_t(2i - 1)}{(h^2 - 1)(e_t + e_r) + e_t}\tau_1 \approx \frac{2h^2 - (i-1)^2 - i^2}{2h^2 - 1}, \tag{11}$$

where $0 < i < h$. The approximation is due to the fact that $e_t \approx e_r$ in practice. To illustrate Eq. 11, let us consider $h = 5$, after τ_1 is set, we obtain $\tau_2 = \frac{45}{49}\tau_1$, $\tau_3 = \frac{37}{49}\tau_1$, $\tau_4 = \frac{25}{49}\tau_1$ and $\tau_5 = \frac{9}{49}\tau_1$.

IV. Capacitated Multi-Vehicle Recharge Problem with Battery Deadlines

In this section, we study a Capacitated Multi-Vehicle Recharge Problem with Battery Deadlines (CaMP-BaD) and consider practical constraints from real sensing applications. The first challenge is the constant changes (i.e., decrease) of SenCars' energy due to moving and recharging sensor nodes. The recharge route should be planned carefully to reflect SenCars's current energy status and traveling costs to nodes' locations. The second challenge is the nonuniform energy consumption due to data transmissions. Some nodes consume energy at higher rates and should be taken care of more frequently than others to maintain the functionality of the network. Further, the random appearances of targets trigger sensing and data traffic, adding to the dynamics of energy consumption. The recharge routes should reflect all aforementioned concerns.

Next we show the recharge problem can be formulated into a *Profitable Traveling Salesmen Problem with Capacity and Battery Deadline constraints* (PTSPCBD). In the Profitable Traveling Salesmen Problem [10], a reward is collected by visiting a city while the objective is to maximize the profit which is defined as the reward minus cost. In our problem, the reward represents the amount of energy that can be replenished into a sensor node and the cost measures the energy cost in traveling to that node's location.

To tackle the problem, we first present a straightforward *Greedy Algorithm* (GA). After realizing that the greedy algorithm might incur extra movements of SenCars, we further propose a three-step *Adaptive Algorithm* (AA) through 1) adaptive network partition using K-means algorithm, 2) Capacitated Minimum Spanning Tree (CMST) formation and 3) route improvements using node insertions. By partitioning the network, the SenCars are confined in their own regions so traveling back and forth through the entire field is avoided. Then we form CMST for each SenCar. The trees indicate which subset of sensor nodes the SenCar should select to minimize traveling cost and ensure the total weight of the tree is within the SenCar's recharge capacity. After that, we perform route improvements on nodes in CMST to capture sensor nodes' dynamic battery deadlines. We also analyze the complexity of the proposed algorithms.

A. Problem Formulation

The recharge optimization problem can be defined as follows. Given a set of SenCars $\mathcal{S} = \{1, 2, \ldots, m\}$ and a set of recharge node list $\mathcal{N}_r = \{1, 2, \ldots, n_r\}$, we formulate the CaMP-BaD problem into a PTSPCBD problem. Consider a graph $G = (V, E)$, where V_i ($i \in \mathcal{N}_r$) is the location of sensor node i to be visited, and E is the set of edges. Each edge E_{ij} is associated with a traveling energy cost c_{ij}, which is proportional to the distance between nodes i and j. A SenCar has recharge capacity C_a that determines the maximum number of nodes it can recharge before it goes back to the base station for its own battery replacement. Different SenCars could have different recharge capacities during the run. Each sensor node i has lifetime l_i and demand (reward) for energy recharge r_i (demand equals the total battery capacity of a sensor node minus its residual energy). A_i specifies the arrival time for a vehicle at sensor node i.

We introduce two decision variables x_{ij} for edge E_{ij} and y_{ia} for vertex V_i. The decision variable x_{ij} is 1 if an edge is visited, otherwise it is 0. The decision variable y_{ia} is 1 if and only if vertex i is served by vehicle a, otherwise it is 0. u_i is the position of vertex i in the path. Our objective is to maximize the total amount of energy recharged minus total traveling energy cost of the SenCars while ensuring the recharge capacities of SenCars are not exceeded and no sensor node depletes battery energy.

$$\mathbf{P1}: \quad \max\left\{\sum_{a=1}^{m}\sum_{i=1}^{n_r} r_i y_{ia} - \sum_{i=1}^{n_r}\sum_{j=1}^{n_r} c_{ij}x_{ij}\right\} \tag{12}$$

Subject to

$$\sum_{j=1}^{n_r} x_{0j}^{(a)} = \sum_{i=1}^{n_r} x_{i0}^{(a)} = 1, \forall a = 1, 2, \ldots, m \tag{13}$$

$$\sum_{i=1}^{n_r} x_{ik} = \sum_{j=1}^{n_r} x_{kj} = 1; \forall k = 2, \ldots, n_r \tag{14}$$

$$\sum_{i=1}^{n_r} r_i y_{ia} \leq C_a, \forall a = 1, 2, \ldots, m \tag{15}$$

$$\sum_{a=1}^{m} y_{ia} = 1, \forall i = 1, 2, \ldots, m \tag{16}$$

$$A_i \leq l_i; \forall i = 1, 2, \ldots, n_r \tag{17}$$

$$x_{ij} \in \{0, 1\}; \forall i, j = 1, 2, \ldots, n_r, \tag{18}$$

$$y_{ia} \in \{0, 1\}; \forall i = 1, 2, \ldots, n_r, \forall a = 1, 2, \ldots, m \tag{19}$$

$$2 \leq u_i \leq n_r; \forall i = 2, 3, \ldots, n_r \tag{20}$$

$$u_i - u_j + (n_r - m)x_{ij} \leq n_r - m - 1 \quad \forall i, j = 2, 3, \ldots, n_r, i \neq j \tag{21}$$

In the above formulation, constraint (13) states that the recharge tour for each SenCar starts at position 0 and finishes at position 0[2]. Constraint (14) ensures the connectivity of the path and every vertex is visited at most once. Constraints (15) and (16) guarantee the vehicle's capacity is not violated and each vertex is visited by only one SenCar. Constraint (17) guarantees the arrival time of the SenCar is within each sensor's residual lifetime. Constraints (18) and (19) impose x_{ij} and y_{ia} to be 0-1 valued. Constraints (20) and (21) eliminate the subtour in the planned route, which is formulated according to [11]. The classic TSP with Profits can be considered as a special case of CaMP-BaD with unlimited capacity and deadlines. Since TSP with Profits is known to be NP-hard [10], CaMP-BaD is also NP-hard.

Due to the nature of our problem, it is not realistic to use standard optimization techniques [12], [13] because these methods deal with datasets of static inputs and the optimization is usually done offline through a one-time effort. In contrast, energy consumption in our framework is probabilistic in nature. A SenCar's recharge capacity declines after recharging sensor nodes, so the input to our problem is more dynamic than that most existing solutions have considered. Furthermore, existing algorithms require high computation power that may not be available on SenCars. Therefore, we need to design algorithms suitable to our problem context. Next, we present two such algorithms.

[2]Position 0 here is the starting position of the SenCar

B. Greedy Algorithm (GA)

The simplest approach is a greedy algorithm which selects the node with the maximal recharge profit (i.e., recharge reward less traveling cost) for each node selection. After a SenCar finishes recharging a node, it picks the next available node with the maximal profit. When the SenCar's energy falls below a threshold χ, it returns to the base station for battery replacement and then resumes recharge in the same fashion.

Despite of its simplicity, GA may have some problems in practice. The first problem is that the SenCar might move back and forth over long distances, thereby increasing the traveling energy cost. This happens when the node with the maximum profit lies faraway, and the energy efficiency of SenCars can deteriorate. Second, because the only consideration is profit, it may not fulfill a recharge request in a fixed time. These observations offer us room for further improvements. To prevent SenCars from traveling long distances, we can confine the scope of movements by partitioning the network into several regions and assigning each SenCar to one of the regions. Second, a more sophisticated scheduling method should be developed to capture SenCars' capacity as well as sensors' battery deadline constraints. In the next subsection, we will introduce an Adaptive Algorithm (AA) to address the limitations in GA.

C. Recharge by Adaptive Algorithm (AA)

1) Adaptive Network Partitioning: In the first step, the base station requests sensor nodes for energy information periodically using the method in [4]. Then it adaptively partitions the network into m *regions* according to the originating locations of requests. The result of partitions is disseminated to the SenCars using long range radio. We utilize the well-known K-means algorithm to perform the partition [14]. Using the K-means algorithm would allow the SenCars to adaptively select a subset of nodes with their square sum of distance minimized regarding to the centroid of each region so the SenCar would only move in a confined scope, and most likely with less distances. For each region, our objective is to minimize the intra-region square sum of inter-node distance. Now we briefly explain the partitioning process.

Initially, we select a number of m sensor nodes with the minimum lifetime from $\mathcal{N}_r$ to be the centroid of regions. We assign each node to the closest centroid. After all the nodes have been associated with a centroid, we re-calculate centroid positions taking the average value of x and y coordinates of nodes in the region. This process is repeated until the centroids no longer change. After the partition, the centroid of each region represents a virtual position that has the minimal sum of distances to all the nodes in its region. This position can be used as the starting position for the SenCar to recharge nodes in its region. As an example of $m = 3$, Fig. 3(a) shows all the nodes requesting for recharge. Fig. 3(b) is the result of K-means partitions.

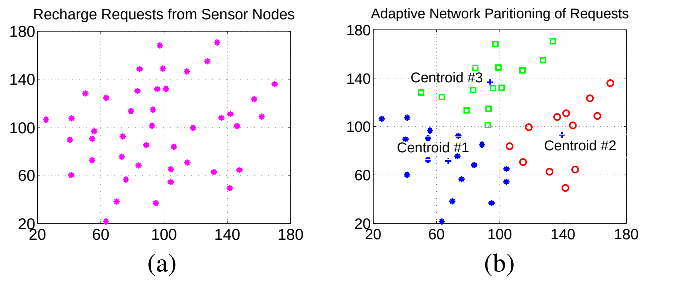


Fig. 3. Procedures of AA (partition). (a) Receive recharge requests. (b) Adaptive network partition.

2) Generating Capacitated Minimum Spanning Tree: In the first step, m regions are generated and each SenCar only needs to take care of the nodes in its region. To decide the route to recharge sensor nodes, we need to ensure each SenCar's recharge capacity is not exceeded (Eq. (15)). On the other hand, we want to minimize the traveling energy cost for the SenCar. These requirements lead to finding Capacitated Minimum Spanning Tree (CMST) [15] where the total sum of demands from nodes does not exceed the SenCar's capacity and the minimum traveling energy cost can be found by constructing the minimum spanning tree.

The exact solution to CMST requires us to go over all possible tree setups and pick the one with the lowest cost, which involves exponential computation. Fortunately, an efficient algorithm by Esau-Williams(EW) can find a suboptimal solution very close to the exact solution in polynomial time [15]. The EW algorithm merges any two subtrees when there is a "saving" in the total cost of two trees.

Nevertheless, there are some limitations of the original EW algorithm when applied for our problem. First, when determining whether two subtrees can be merged, only the demands from sensor nodes are counted whereas the traveling costs on edges are not considered. Second, multiple such trees can be generated. How does the SenCar decide which tree to pick? To overcome these limitations, we extend the original EW algorithm. First, in [17], a deterministic upper bound on the shortest tour length is developed as $\sqrt{2(n-2)ab} + 2(a+b)$ for a rectangle of side length a, b and n nodes. If we consider the circular field as inscribed in a square with side length $2r$, we have a loose upper bound on the traveling cost of nodes in the subtree with n_b nodes, $2r(\sqrt{2(n_b-2)}+4)e_c$. Second, when multiple trees are generated, we select a tree that maximizes the ratio of total energy demand to traveling cost. In this way, we can exploit limited resources on SenCars better and improve energy efficiency of the network.

Next, we explain our extension to the EW algorithm in detail. Each SenCar computes CMST independently by iteratively updating a distance matrix. The distance matrix facilitates the computation process by maintaining costs of tree nodes. Let us denote recharge set $\mathcal{N}_a$ with n_a nodes for SenCar a ($\bigcup_{a=1}^{m} \mathcal{N}_a = \mathcal{N}_r$). We define trade-off function t_i for each node in its recharge set $\mathcal{N}_a$, $t_i = \min(c_{ij}^{(a)}) - c_{0i}^{(a)}$ and $j \in P_i$, where P_i is the neighboring set of node i, $\min(c_{ij}^{(a)})$ finds the minimum cost from node i to its neighbor j in P_i and $c_{0i}^{(a)}$ is the cost from node i to SenCar's starting position (i.e., the root)[3]. The trade-off function evaluates whether it is beneficial to merge subtrees of nodes i and j. A positive t_i indicates that it incurs smaller cost for the SenCar to directly travel from the root to node i so merging subtrees of nodes i and j is not preferred. A negative t_i indicates how much it can be saved by connecting subtrees of i and j. Thus the most negative t_i results in the most savings in an iteration.

[3]In order to reduce intra-region traveling cost, we set the centroid output from K-means algorithm to be the root.

After t_i has been computed, we search through all trade-offs $t_i (\forall i = 1, \ldots, n_a)$, looking for the minimum trade-off (i.e., the most negative value). Assume t_k is the most negative trade-off and j is k's minimum cost neighbor. To capture SenCar's capacity constraint in Eq. (15), if the sum of total demands from the subtrees of k and j plus upper bound of their traveling cost is less than the recharge capacity (which means we can cover the subtrees of k and j under the current recharge capacity), we merge the subtrees of k and j. Since the action of merging k and j has resulted in a lower total traveling cost to k, direct traveling from the root to reach k has higher cost and should be avoid. So we remove the edge from node k to the root by setting $c_{0k}^{(a)}$ in the distance matrix to ∞.

At this point, two subtrees satisfying the recharge capacity with minimum sum of cost have been merged, and we need to update the minimum cost of the newly merged tree to the root. It is done by updating the minimum cost in the distance matrix from the tree to the root by setting the value to $\min(c_{0i}^{(a)})$, where i is the node in the newly merged tree.

On the other hand, if merging subtrees of k and j violates SenCar's recharge capacity, we need to restrict any further actions to merge j to k because these two trees cannot be covered by the SenCar in a single run. Then we recompute the trade-off function t_k to search for the next neighboring node that results in minimum trade-off until the next valid neighboring node j is found and merged to the existing trees. The iteration continues until all the trade-offs become nonnegative, in other words, no more saving can be made.

After the CMST has been generated, the SenCar selects a tree with the maximal ratio of recharge demand to traveling cost and utilize the route improvement algorithm to form the final recharge sequence among the tree nodes. After the SenCar finishes recharging nodes in a tree, it checks whether its energy falls below a threshold. If so, it returns to the base station for battery replacement. Table II shows the pseudo-code of our extended EW algorithm. Fig. 4(a) shows an example of CMST formed in each region.

3) Insertion Algorithm for Route Improvement: After the CMST has been obtained, next we want to produce a recharge sequence for nodes such that for each node the SenCar arrives before its battery deadline. Let us denote the result from CMST to be a recharge node set $\mathcal{N}_r^{(a)}$ ($\mathcal{N}_r^{(a)} \subseteq \mathcal{N}_a$). Recall that if the condition in Proposition 2 is satisfied, a node can be placed anywhere in the recharge sequence. We call such a set of nodes a *feasible node set* $\mathcal{N}_f^{(a)}$. Otherwise, a node may need prioritized treatment to meet its battery deadline. We denote such a set of nodes as a prioritized set $\mathcal{N}_p^{(a)}$ ($\mathcal{N}_f^{(a)} \cup \mathcal{N}_p^{(a)} = \mathcal{N}_r^{(a)}$).

Intuitively, we first use a Traveling Salesman Problem algorithm (e.g., the $\mathcal{O}(n^2)$ nearest neighbor heuristic algorithm [16], where n is the number of nodes) to find a feasible solution as the initial sequence Ψ for nodes in the feasible set $\mathcal{N}_f^{(a)}$. Then we

TABLE II
EXTENDED ESAU-WILLIAMS ALGORITHM

```
input: recharging node set $\mathcal{N}_r$, distance matrix $D^{(a)}$,
recharge capacity $C_a$, demand of nodes $d_i$, $i \in \mathcal{N}_a$.
output: CMST nodes need to recharge.
Initialize $t^{(a)} < 0$, weight of tree, $C^{(a)} = 0$.
while ($t^{(a)} < 0$)
  Find neighbor $m_i$ of $i$ results min cost, $\min_{m_i} D^{(a)}(i, m_i)$.
  Compute trade-off value list $t_i^{(a)} = D^{(a)}(i, m_i) - D^{(a)}(1, i)$.
  Find $k$ and $j$ resulting most negative trade-off value,
  $k \leftarrow \min_i(t^{(a)}), j \leftarrow m_k$.
   do
    Add new nodes $N_{new} \leftarrow k + j$ if not exist in current trees
    if weight of merging subtree of $N_{new} < C_a$
     Add $N_{new}$ to corresponding tree $i$
     Update cumulative weight of corresponding tree $i$, $C_i^{(a)}$.
     Declare $N_{new}$ is accepted.
     else
     update $D^{(a)}(k, j) \leftarrow \infty$
     Search for next min cost neighbor for $k$,
     $m_k \leftarrow \min_{m_k} D^{(a)}(k, m_k)$.
     Recompute trade-off for $k$, $t_k^{(a)} = D^{(a)}(k, m_k) - D^{(a)}(1, k)$.
     Declare $N_{new}$ rejected.
    until ($N_{new}$ is accepted) or (all $t_i^{(a)} \geq 0$)
end while
Select a tree results maximum energy efficiency.
```

TABLE III
INSERTION ALGORITHM

```
input: CMST $\mathcal{N}_r^{(a)}$, lifetime $l_i$ and recharge time $t_i$, $i \in \mathcal{N}_r^{(a)}$,
distance matrix $D^{(a)}$, feasible set $\mathcal{N}_f^{(a)}$ satisfying Proposition 2.
output: resultant recharge sequence $\Psi$.
Compute shortest path in the feasible set, $\Psi \leftarrow$ TSP($\mathcal{N}_f^{(a)}$)
Sort $\mathcal{N}_p^{(a)}$ in a descending order of lifetime as $\Omega$
Initialize $i \leftarrow 1$, last step node position $k \leftarrow \infty$.
while $\Omega \neq \emptyset$
  Find temporary max position $m_t$ in $\Psi$ such that
  $A_{m_t} \leq l_{\Omega_i}$ and $A_{m_t+1} > l_{\Omega_i}$
  Find the max insertion position $m$ such that
  $A_m \leq A_{m_t} - \sum_{k=i+1}^{n_r^p} t_k$ and $A_{m+1} > A_{m_t} - \sum_{k=i+1}^{n_r^p} t_k$.
   if Cannot find $m \geq 0$. break,return infeasible and report.end if
  Set minimum cost $c_{min} \leftarrow \infty$.
   for $x$ from 0 to $m$
    Insert node $\Omega_i$ into $\Psi$, get temporary sequence $\Psi_t$
    Calculate cost $c \leftarrow \sum_{j=1}^{|\Psi_t|-1} D^a(j, j+1)$.
    if $c < c_{min}$, $\Psi \leftarrow \Psi_t$, $c_{min} \leftarrow c$, $k \leftarrow x$. end if
   end for
  $i \leftarrow i + 1$, update $\Omega \leftarrow \Omega - i$
end while
Return recharge sequence $\Psi$, minimum cost $c_{min}$.
```

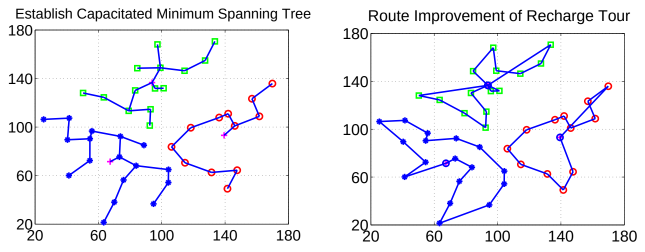
insert nodes from the prioritized set $\mathcal{N}_p^{(a)}$ into Ψ while ensuring the battery deadline in Eq. (17) for all nodes are still met. To this end, we sort the nodes in $\mathcal{N}_p^{(a)}$ in a descending order of residual lifetimes and denote the sorted sequence as Ω. We insert these nodes starting from the first node Ω_1. Let A_i denote the arrival time of the SenCar at the i-th node in the shortest path Ψ, $i = \{1, 2, \ldots, n_f^{(a)}\}$.

To insert the j-th node Ω_j from Ω into Ψ, we first find position m_t in Ψ such that $A_{m_t} \leq l_{\Omega_j}$ and $A_{m_t+1} > l_{\Omega_j}$ where l_{Ω_j} is Ω_j's lifetime. We call m_t the *temporary maximum position* to insert Ω_j. It indicates the maximum number of nodes in Ψ that can be served before node Ω_j depletes its battery. To accommodate the remaining $|\Omega| - j$ nodes, we need to find a position such that even all the remaining nodes are inserted before Ω_j, Ω_j can still meet its battery deadline. We find the maximum position m such that $A_m \leq A_{m_t} - \sum_{i=j+1}^{n_p^a} t_i$ and $A_{m+1} > A_{m_t} - \sum_{i=j+1}^{n_p^a} t_i$, where t_i is the recharge time of Ω_j. Now, the maximum position m represents the rightmost position Ω_j can be inserted if all remaining nodes are later inserted before Ω_j.

For each of the m possible positions that Ω_j can be inserted, a total traveling cost is computed and the one that minimizes the traveling cost is selected as the final insertion position for Ω_j. Then we obtain a new sequence Ψ and remove Ω_j from Ω. The iteration continues until we exhaust Ω or an infeasible situation is encountered. Table III shows the pseudo-code of the insertion algorithm. Fig. 4(b) demonstrates the recharge tours for the SenCars in their designated regions.

D. Complexity Analysis

We now analyze the complexity of our algorithms. The complexity of the greedy algorithm is $\mathcal{O}(n_r)$ because it only selects the maximum profitable node at each step. For the



(a) (b)

Fig. 4. Procedures of Adaptive Algorithm (CMST and recharge route). (a) Establish CMST. (b) Improve recharge route.

adaptive algorithm, the base station has abundant resources and it performs the k-means algorithm. So we focus on the computing burdens on SenCars for calculating CMST and route improvements. In the worst case, there is only one SenCar to recharge n_r nodes. For the extended EW algorithm, finding the minimum trade-off value requires $n_r^2 + 2n_r$ iterations at the outer loop. In the inner loop, the worst case is that for a node with the minimum trade-off value, every minimum-cost neighbor is rejected due to capacity violations. So n_r iterations are required. In sum, its time complexity is $\mathcal{O}(n_r^3)$.

For the route improvement algorithm, running a TSP algorithm requires $\mathcal{O}(n_r^2)$ time. Sorting nodes' lifetimes requires $\mathcal{O}(n_r \log n_r)$ time. Then, insertion requires $\mathcal{O}(n_r^2)$ time. Hence, the total time complexity of route improvement algorithm is $\mathcal{O}(n_r^2)$ and the adaptive algorithm takes $\mathcal{O}(n_r^3)$ time.

V. PERFORMANCE EVALUATIONS

We have conducted extensive simulations to evaluate our algorithms using a discrete event-driven simulator. We examine a network of 500 sensor nodes uniformly randomly distributed over a circular field with radius $r = 100m, h = 5$. Dijkstra's shortest path routing algorithm is used to send messages to the base station at rate $\lambda = 10$ pkt/min. Each time slot is 1 min. Sensor nodes are equipped with CC2430 communication module that draws 27mA current when either transmitting or receiving at a voltage of 3V. Messages have the same length at 30 bytes and the bit rate is 1 Kbps ($e_t = e_r = 1.94mJ$). The sensing unit comprises of a plugged-in Passive Infra-Red module which on average draws 5mA at 3V. When there are targets in a node's sensing range, it consumes 0.9 J/min (5mA $\times$ 3V $\times$ 60s). Since a higher initial energy takes longer time for the network to achieve equilibrium, we set all the sensors to have 50% of their full battery initially to make the network enter equilibrium faster.

Sensor nodes have adaptive recharge thresholds at different rings according to Eq. (11), with $\tau_1 \sim \tau_5 = 0.95, 0.87, 0.72, 0.48, 0.18$, respectively. The battery's recharge time is modeled from [5]. We assume SenCars are electric-powered vehicles carrying FPGAs (for computation), communication modules and high density battery packs (e.g., 12A, 5V standard ones can easily satisfy our needs). The vehicle can weight tens of pounds and we assume it is 20 lbs. Using the method in [8], we estimate that a vehicle consumes energy at a rate of 5.59 J/m. To evaluate how the number of SenCars affects system performance as well as validate theoretical results in Proposition 1, we vary the number of SenCars m from 2 to 4 and set the simulation time to 4 months.

A. *Nonfunctional Nodes and Energy Consumption vs. Recharge*

First, we examine the evolution of the number of nonfunctional nodes. When a sensor node depletes its battery energy, it becomes nonfunctional until recharged by SenCars. Fig. 5 presents the results of nonfunctional nodes by proposed algorithms.

For the Greedy algorithm (GA), when $m = 2$, the number of nonfunctional nodes surges intensively around 22 days to over 50% until it slowly decreases to within 10% around 42 days. This is because the SenCars find out that nodes closer to the base station have more profits. Thus they do not serve nodes in the outmost ring (36% of nodes) immediately even after their requests. SenCars only cover them when their batteries nearly deplete. By then, SenCars' recharge capacity ($m = 2$) is temporarily exceeded, which causes the big spike. Although the two SenCars can gradually resolve most nonfunctional nodes, it is observed that there is persistently more than 5% nonfunctional nodes. In contrast, the Adaptive Algorithm (AA) provides more stability. When $m = 2$, there is no such huge spike. Nonfunctional nodes are well within 10% all the time. This is because AA captures the sensor battery deadlines. For $m = 3, 4$, AA also yields better results. When $m = 4$, AA can reduce the nonfunctional nodes to zero.

We observe that $m = 3$ is likely to be a threshold since two SenCars are definitely not enough. From Proposition 1, after plugging in the experimental parameters, we obtain $m = \lceil 3.15 \rceil = 4$. Thus $m = 3$ can barely satisfy the energy neutral condition. This calculation matches our observations in Fig. 5 (b), validating the correctness of our theoretical results.

Fig. 6 shows the cumulative energy consumption vs. replenishment. For clarity and better observing the gaps and intersections between curves, we plot 2 months' simulation time and magnify a portion of the curves. The curves represent the overall energy status of the network. If the energy replenishment curve is above the energy consumption curve, more energy has been refilled into the network than consumed, and vice versa. First, in Fig. 6(a), the replenishment curves when $m = 3, 4$ are well above the energy consumption curves for GA. A larger gap is observed with $m = 4$ representing higher energy replenishment capabilities. In

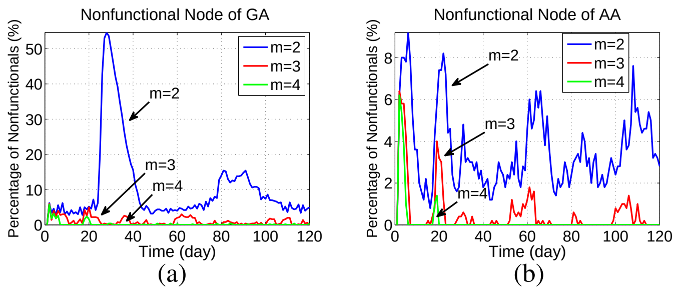


Fig. 5. Evolution of nonfunctional nodes. (a) GA. (b) AA.

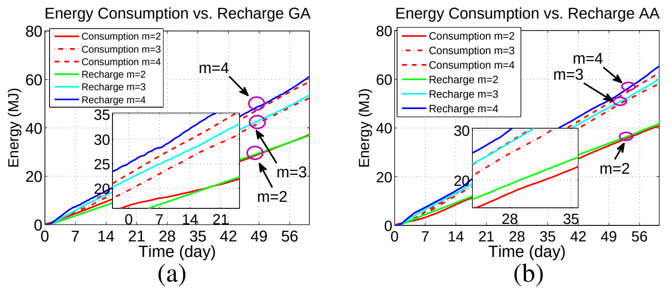


Fig. 6. Cumulative energy consumption vs. replenishment. (a) GA. (b) AA.

contrast, with $m = 2$, the energy consumption curve stays above energy replenishment curve until the two curves start to merge around 22 days. This is because from the very beginning, more energy is consumed than replenished. Around 22 days, a large number of nodes have depleted energy and stopped consuming energy, which brings down the consumption curve. In Fig. 6(b), all the replenishment curves are above their consumption curves indicating that AA can put more energy into the network than GA.

B. *Energy Cost of SenCars and Data Collection Latency*

Next we evaluate the traveling energy cost of SenCars. Table IV compares the average traveling cost per SenCar for the proposed algorithms. When $m = 2$, a little more energy cost is observed with AA. Since there are only two partitions when $m = 2$, the SenCar still needs to travel on half of the sensing field. AA takes care of nodes in the outmost ring preemptively before they deplete energy, thus more energy cost is observed. With GA, the SenCars travel to the outmost ring only when the recharge profits there are larger, but at that time, those nodes nearly deplete their energy and many of them become nonfunctional as observed earlier. Although GA has lower traveling cost when $m = 2$, the network performance deteriorates greatly. When $m = 3, 4$, we can partition the network into more regions with smaller sizes so the movements of SenCars can be confined in smaller regions. This brings down the energy costs for SenCars. However, as GA still directs SenCars to recharge nodes with the maximal profit, long distance travels are inevitable. Thus AA incurs less energy cost than GA.

To transmit the sensed data to the base station in a timely manner, all the nodes should be functional on a routing path. We assume static shortest routing paths by Dijsktra's algorithm

TABLE IV
COMPARISON OF TRAVELING ENERGY COST ON SENCARS.

Number of SenCars	$m = 2$	$m = 3$	$m = 4$
GA (KJ)	0.33	0.78	1.19
AA (KJ)	0.41	0.67	0.94

TABLE V
COMPARISON OF AVERAGE DATA COLLECTION LATENCY.

Number of SenCars	$m = 2$	$m = 3$	$m = 4$
GA (hrs)	6.14	4.22	1.9
AA (hrs)	0.74	0.05	0.02

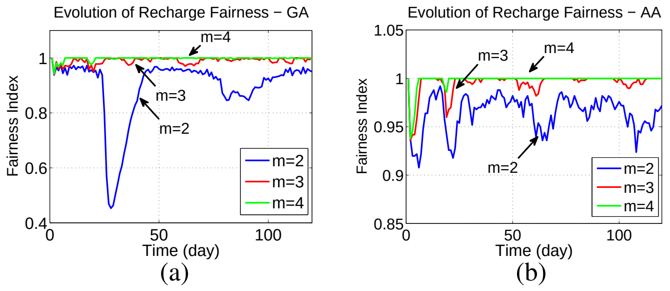


Fig. 7. Comparison of recharge fairness. (a) GA. (b) AA.

are used. If a node depletes energy, all the pending messages are buffered at senders until the path is restored. Table V shows average data latency over the entire simulation time. We can see that AA results in much shorter latency than GA. The significant improvement is due to the route improvement algorithm that treats nodes with lower lifetimes with higher priority.

C. Recharge Fairness and Duration of Nonfunctional Nodes

Recharge fairness indicates whether SenCars can recharge nodes commensurate to their workloads. Those having more workload (e.g., nodes near the base station) should be recharged more frequently. This is reflected from the functional time of sensor nodes. To quantify recharge fairness, we leverage the fairness index from [18],

$$F = \frac{(\sum_{i=1}^{n} x_i)^2}{n\sum_{i=1}^{n}(x_i)^2}, \tag{22}$$

in which x_i is a normalized indicator whether a node is functional in a time slot. x_i equals $1/N_s$ if i is functional in a time slot, otherwise, it is zero. The fairness index F ranges from 0 (worst case if all nodes are nonfunctional) to 1 (best case if all the nodes are functional) here. We can see from Fig.7(b) that AA can distribute energy resources fairly among the nodes especially when $m = 4$. In Fig.7(a), when the nodes in the outmost ring become nonfunctional, the fairness of GA algorithm severely degrades as the SenCars only recharge nodes with maximum profits.

Fig. 8 plots the percentage of nonfunctional durations of nodes as a function of their locations. Using GA, nodes near the base station has a maximum of 5.14% time in nonfunctional states whereas AA only 0.9%. Further, AA spreads nonfunctional durations across the field while the spikes of GA are highly concentrated. This is because that nodes close to the base station consume energy faster and are more prone to become nonfunctional. GA considers profit only and has no measure for battery deadlines. In contrast, AA considers both profit and battery deadlines. Therefore, the duration of nonfunctional nodes with AA is significantly less than that of GA.

VI. CONCLUSIONS

In this paper, we consider two important factors overlooked by previous studies, the charging vehicle's energy consumption and capacity limits in WRSN. We first establish a practical mathematical model, formally derive the probability that the

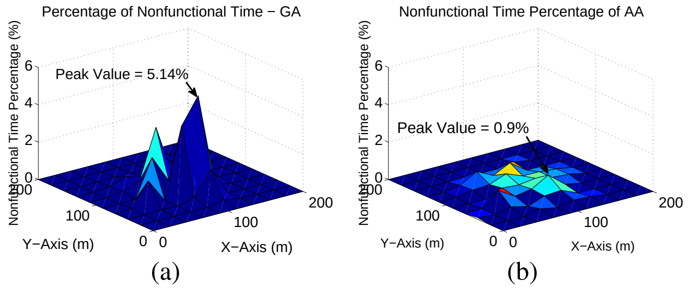


Fig. 8. Comparison of durations for nonfunctional nodes when $m = 4$. (a) GA. (b) AA.

energy neutral condition holds and the minimum number of SenCars needed. We formulate recharge optimization problem into a Profitable Traveling Salesmen Problem with Capacity and Battery Deadline constraints, which is NP-hard. To solve the problem with low complexity, we propose two algorithms. The greedy algorithm maximizes the recharge profit in each step. We further propose a three-step adaptive algorithm that systematically captures the recharge capacity and nodes' battery deadline constraints in the problem while minimizing traveling costs. We evaluate and compare the performance of the proposed algorithms by extensive simulations. They show that the adaptive algorithm can provide better stability for the network by reducing the number of nonfunctional nodes and the length of their nonfunctional durations. We also validate the theoretical results through simulations and demonstrate the adaptive algorithm can save energy on SenCars.

REFERENCES

[1] A. Kurs, A. Karalis, R. Moffatt, J. D. Joannopoulos, P. Fisher and M. Soljacic, "Wireless power transfer via strongly coupled magnetic resonances," *Science*, vol. 317, pp. 83, 2007.
[2] M. Zhao, J. Li and Y. Yang, "Joint mobile energy replenishment and data gathering in wireless rechargeable sensor networks," *IEEE ITC*, 2011.
[3] Y. Shi, L. Xie, T. Hou and H. Sherali, "On renewable sensor networks with wireless energy transfer," *IEEE INFOCOM*, 2011.
[4] C. Wang, J. Li, F. Ye and Y. Yang, "Multi-Vehicle coordination for wireless energy replenishment in sensor networks," *IEEE IPDPS*, 2012.
[5] Panasonic Ni-MH battery handbook, "http://www2.renovaar.ee/userfiles/Panasonic_Ni-MH_Handbook.pdf".
[6] S. Dunbar,"The average distance between points in geometric figures," *The College Mathematics Journal*, vol. 28, no. 3, 1997, pp. 187-197.
[7] X. Wu, G. Chen and S. Das, "Avoiding energy holes in wireless sensor networks with nonuniform node distribution," *IEEE TPDS*, vol.19, no.5, 2008.
[8] Battery Calculator, "http://www.evsource.com/battery_calculator.php."
[9] S. Ross, *A First Course in Probability*, 8th Ed, Prentice Hall, 2009.
[10] D. Feillet, P. Dejax and M. Gendreau, "Traveling salesman problems with profits," *Transportation Science*, vol. 39, no. 2, 2005.
[11] B. Gavish, "A note on the formulation of the m-salesman traveling salesman problem," *Management Science,* 1976.
[12] M. Desrochers, J.K. Lenstra, M.W.P. Savelsbergh and F. Soumis, "Vehicle routing with time windows: optimization and approximation," Elsevier Science Publisher, 1988.
[13] B. Chandran and S. Raghavan, "Modeling and solving the capacitated vehicle routing problem on trees," *The Vehicle Routing Problem: Latest Advances and New Challenges*, pp 239-261, 2008.
[14] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proc. of 5th Berkeley Symposium on Math. Statistics and Probability*, 1967, pp. 281-97.
[15] L.R. Esau and K.C. Williams, "On teleprocessing system design: part II-a method for approximating the optimal network," *IBM Syst Journal*, vol. 5, pp. 142-147, 1966.
[16] T.H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, MIT Press, 2001.
[17] P. Jaillet, "Probabilistic traveling salesman problem," Ph.D. Dissertation, MIT, 1985.
[18] R. Jain, D. M. Chiu, W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," *DEC Research Report TR-301.*