Collaborative Mobile Charging

Sheng Zhang, *Student Member, IEEE*, Jie Wu, *Fellow, IEEE*, and Sanglu Lu, *Member, IEEE*

Abstract—The limited battery capacity of sensor nodes has become one of the most critical impediments that stunt the deployment of wireless sensor networks (WSNs). Recent breakthroughs in wireless energy transfer and rechargeable lithium batteries provide a promising alternative to power WSNs: mobile vehicles/robots carrying high volume batteries serve as mobile chargers to periodically deliver energy to sensor nodes. In this paper, we consider how to schedule multiple mobile chargers to optimize energy usage effectiveness, such that every sensor will not run out of energy. We introduce a novel charging paradigm, *collaborative mobile charging*, where mobile chargers are allowed to intentionally transfer energy between themselves. To provide some intuitive insights into the problem structure, we first consider a scenario that satisfies three conditions, and propose a scheduling algorithm, `PushWait`, which is proven to be optimal and can cover a one-dimensional WSN of infinite length. Then, we remove the conditions one by one, investigating chargers' scheduling in a series of scenarios ranging from the most restricted one to a general 2D WSN. Through theoretical analysis and simulations, we demonstrate the advantages of the proposed algorithms in energy usage effectiveness and charging coverage.

Index Terms—Collaborative mobile charging, wireless energy transfer, wireless sensor networks

1 Introduction

Wireless sensor networks (WSNs) have been used extensively in many fields, from rainfed agriculture [1] to forest fire detection [2], to structural health monitoring [3], and to home automation [4]. Due to the limited energy capacity of the battery used at each sensor node, WSNs can only remain operational for a limited amount of time, which becomes by far one of the most critical impediments that stunt the growth of WSNs. Studies on extracting energy from the environment [5], [6], [7], [8], [9] have shown great promise of addressing this issue, however, they remain limited in practice due to the partial predictability of harvesting and the relatively large sizes of harvesting devices [10].

Recent breakthroughs in *wireless energy transfer* and *rechargeable lithium batteries* provide a promising alternative to power WSNs. Kurs et al. [11] experimentally demonstrated that energy can be efficiently transmitted between magnetically resonant objects without any interconnecting conductors. Kang et al. [12] showed rechargeable lithium batteries with high energy densities and high charge/discharge capabilities.

Inspired by these enabling technologies, prior studies [10], [13], [14] envisioned employing a mobile vehicle or robot [15] carrying a high volume battery as a mobile charger to periodically deliver energy to sensor nodes; they mainly focused on maximizing the ratio of the charger's vacation time (time spent at the home service station) over the cycle time [10], [13], or concentrated on minimizing the total delay to charge all nodes in a WSN above their energy threshold [14]. However, we observed that, most of them assumed that a mobile charger has a sufficient amount of energy to not only replenish an entire WSN, but also to make a round-trip back to the base station. This model becomes invalid when there is a remote area that even a dedicated charger with a full battery cannot reach before running out of energy.

In this paper, we do not impose any restrictions on the size of a WSN or the charger's capacity, and we consider how to optimize energy usage effectiveness (EUE). The energy consumed in replenishing a WSN can be classified into three categories: the energy eventually obtained by sensor nodes, which is considered as payload energy; the energy used by mobile chargers for moving; and the energy loss during wireless energy transfer. To maximize energy usage effectiveness, which is defined as the ratio of the amount of payload energy to total energy, we introduce a novel *collaborative mobile charging* paradigm [16], which allows energy transfer between mobile chargers. By careful selection of the location of and the amount of energy transferred at each rendezvous point, not only is the energy usage effectiveness improved, but the charging coverage is also enlarged.

To provide some intuitive insights into the structure of our problem, we first consider the scenario that satisfies the following three conditions: ($K1$) all sensor nodes are distributed along a one-dimensional (1D) line; ($K2$) the recharging cycles of all sensor nodes are the same; ($K3$) there is no energy loss during any energy transfers. For this scenario, we propose an algorithm called *PushWait*, which is proven to be optimal and can cover a 1D WSN of any length. We then remove these conditions one by one, and finally investigate the mobile charging scheduling problem (MCS) in general two-dimensional (2D) WSNs. The contributions of this paper are summarized as follows:

1) To the best of our knowledge, we are the first to introduce collaborative mobile charging. Through

- *S. Zhang and S. Lu are with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China. E-mail: zhangsheng@dislab.nju.edu.cn, sanglu@nju.edu.cn.*
- *J. Wu is with the Computer and Information Sciences Department, Temple University, Philadelphia, PA 19122. E-mail: jiewu@temple.edu.*

Manuscript received 30 Apr. 2013; revised 12 Nov. 2013; accepted 29 Dec. 2013. Date of publication 8 Jan. 2014; date of current version 11 Feb. 2015. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TC.2013.2297926

theoretical analysis and simulations, this paper demonstrates its advantages in energy usage effectiveness and charging coverage.

2) We develop a set of scheduling algorithms for five different scenarios ranging from the simplest one in Section 4 to the hardest one in Section 7, in an effort to provide some potential guidelines for the future design of charging scheduling.
3) We perform extensive theoretical analysis and simulation studies to verify the performance of the proposed set of algorithms.

2 Related Work

There is a rich heritage of work in prolonging the lifetime of WSNs that informed and inspired our algorithms. We describe a subset of these related efforts in this section.

Energy harvesting. Energy harvesting [5] extracts environmental energy (e.g., solar, wind, vibration) to replenish sensor nodes. Cammarano et al. [6] developed an accurate energy prediction model for solar/wind energy harvesting. Kansal and Srivastava [7] proposed to proactively learn energy environment for performance adaption. Solar-aware routing was presented in [8]. Further, Kansal et al. [9] provides methods to systematically utilize environmental resources in a performance-aware manner. While these works are orthogonal to the major focus of this paper, they complements each other, showing great promise of addressing the energy problem in WSNs.

Energy conservation. Bhattacharya et al. [17] proposed to cache mutable data at some locations to control data retrieval rate, for slowing down energy consumption rate. Dunkels et al. [18] incorporated cross-layer information-sharing in their proposed architecture. Wang et al. [19] proposed using resource-rich mobile nodes as sinks/relays to balance energy usages. In general, conservation cannot compensate for depletion; we focus on optimizing chargers' scheduling. A combination of energy conservation and charger scheduling can further improve the energy usage effectiveness.

Wireless energy transfer. The wireless power consortium [20] defines the inter-operability standards of wireless energy transfer based on magnetic induction. Peng et al. [21] focused on maximizing network lifetime through find an optimal charging sequence. Li et al. [22] considered the same goal, but additionally took routing into account. Tong et al. [23] evaluated the performance of multi-node simultaneous charging, and focused on optimizing sensor deployment and packet routing to improve energy efficiency. Shi et al. [10] assumed that the mobile charger has unbounded energy, and investigated the problem of periodically charging sensors to maximize the ratio of the charger's vacation time (time spent at the home service station) over the cycle time. They further considered the multi-node simultaneous charging scenario [13]. He et al. [24] investigated the energy provision problem of finding the minimum number of RFID readers to cover a given network. Fu et al. [14] focused on minimizing the total delay of replenishing all sensor nodes in a network. Comparatively, we consider a different scenario in which a single mobile charger may not have enough energy to cover the entire network. We introduce the collaborative mobile charging paradigm, and develop a set of scheduling algorithms for maximizing the energy usage effectiveness.

3 Problem Formulation

3.1 Sensors and Chargers

We consider N stationary sensor nodes distributed over a two-dimensional area. The location of the ith node s_i is denoted as (x_i, y_i). The battery capacity of s_i is b_i. Each node consumes energy for sensing, data reception, and transmission. We represent the average energy consuming rate of s_i as r_i. The recharging cycle of a sensor is defined as the time period that the sensor with a full battery can survive without being charged. Let τ_i be the recharging cycle of s_i, we have $\tau_i = b_i/r_i$. We further denote $(x_1, x_2, \ldots, x_N)$ by $\mathbf{X}$, $(y_1, y_2, \ldots, y_N)$ by $\mathbf{Y}$, $(b_1, b_2, \ldots, b_N)$ by $\mathbf{B}$, and $(\tau_1, \tau_2, \ldots, \tau_N)$ by $\mathbf{T}$.

Multiple mobile chargers are employed to replenish a given WSN. The ith mobile charger is written as $\mathcal{C}_i$ for short. We assume mobile chargers are homogeneous: for every charger, the battery capacity is P, the travelling speed is v, and energy consumed by travelling one unit distance is c. Both travelling and wireless charging share the same battery of a mobile charger.

The base station BS serves as data sink as well as energy source. Without loss of generality, we assume BS is located at $(0, 0)$. Mobile chargers start from the BS with full batteries; when they collaboratively finish the charging task, they will return to the BS to be serviced (e.g., recharging their own batteries). We assume that charging can happen only when two objects share the same location, and leave as future work the case where multiple sensor nodes are charged simultaneously.

We denote by η_1 the wireless charging efficiency between a charger and a sensor node, i.e., a charger $\mathcal{C}$ consumes one unit of energy while a sensor can only receive η_1 units of energy. Similarly, denote by η_2 the efficiency between two chargers. We further denote by $\mathcal{C}_i \xrightarrow{e} \mathcal{C}_j$ (resp. $\mathcal{C}_i \xrightarrow{e} s_j$) the event that charger $\mathcal{C}_i$ transfers e units of energy to charger $\mathcal{C}_j$ (resp. sensor s_j), which receives only $\eta_2 e$ (resp. $\eta_1 e$) units of energy.

For simplicity of exposition, we discuss our solutions with the following assumptions. *Short duration (SD)*: the duration of a charging is negligible compared to the traveling time of mobile chargers, and *Long Cycle (LC)*: the recharging cycle of a sensor node is longer than a charging round.[1] Our solutions can be applied to contexts without these two assumptions, see Section 8.

3.2 The Mobile Charging Scheduling Problem

The task of scheduling is to decide the actions (e.g., charging a sensor/charger, being charged, waiting) of each charger in its respective time-space trajectory. In a *feasible* scheduling, every sensor node can get charged before running out of energy, and every charger is able to return to the BS to be serviced.

1. Thus, any two consecutive charging rounds have no intersections, i.e., mobile chargers can always accomplish a charging round, return to the BS, and wait for another charging round.

We define the *scheduling cycle* of a feasible scheduling as the time interval between two consecutive time points when each sensor has the same battery level. To evaluate the long-term energy efficiency of a scheduling, we only have to consider the energy usage in a scheduling cycle.

In a scheduling cycle, the energy consumed in replenishing a WSN contains three parts: the energy eventually obtained by sensors, the energy consumed by chargers' travelling, and the energy loss during charging. We define the first part as *payload energy* (E^{pl}), and the sum of the second and third parts as *overhead energy* (E^{oh}). The *energy usage effectiveness* metric is defined as:

$$EUE = \frac{E^{pl}}{E^{pl} + E^{oh}}. \quad (1)$$

Problem 1 (Mobile charging scheduling problem). *Given a WSN* $(\mathbf{X}, \mathbf{Y}, \mathbf{B}, \mathbf{T})$ *and a charging model* $(P, c, v, \eta_1, \eta_2)$*, we must discover how to schedule chargers to replenish the WSN, such that EUE is maximized.*

3.3 The Organization

To better understand the problem structure, we first consider some special scenarios, then we use the knowledge obtained from these scenarios to find solutions to more general ones. We define three conditions as follows:

(K1) All sensor nodes are distributed along a 1D line.[2] Without loss of generality, we let $\forall 1 \leq i \leq N$, $y_i = 0$.

(K2) All sensor nodes have the same recharging cycle, i.e., $\forall 1 \leq i \leq N$, $\tau_i = \tau$.

(K3) Wireless energy transfer has no energy loss, i.e., $\eta_1 = \eta_2 = 1$.

We further use $\overline{Kj}$ to indicate that Kj does not hold, $j \in \{1, 2, 3\}$. For example, $K1K2\overline{K3}$ represent the scenario that $K1$ and $K2$ hold, while $K3$ does not hold. We consider $K1K2K3$ and $K1K2\overline{K3}$ in Section 4 and Section 5, respectively. Both $K1\overline{K2}K3$ and $K1\overline{K2K3}$ are discussed in Section 6. We finally consider MCS in a general 2D WSN, i.e., $\overline{K1K2K3}$, in Section 7. Extensions and remarks are provided in Section 8. Simulation results are presented in Section 9. We conclude this paper in Section 10.

4 $K1K2K3$: 1D, Same Recharging Cycle, No Energy Loss

In this section, we first present an argument for simplifying the discussion in this scenario $K1K2K3$, we then show three motivational examples before introducing the proposed solution. We provide two good properties of our algorithm at the end of this section.

We argue that, *given a WSN satisfying K1K2K3, we can maximize EUE through minimizing the sum of distances travelled by chargers*. The main reason is that, *relative to the beginning of each charging round, the time point when each sensor is recharged to its full battery is fixed*. Formally, suppose that a charging round starts at time T_1, and node s_i gets fully charged at time $T_1 + \Delta_i$. We let the next charging round start at time $T_1 + \Delta T$, then node s_i would get fully charged at time $T_1 + \Delta T + \Delta_i$. Since E^{oh} only contains energy consumed by chargers' movement (due to $K3$), to avoid unnecessary movements, ΔT should be as long as τ, and the scheduling cycle is τ too. Thus, the payload energy in a scheduling cycle is $\sum_i r_i(T_1 + \tau + \Delta_i - T_1 + \tau) = \sum_i r_i \tau = \sum_i b_i$, which is fixed for a given WSN. Therefore, maximizing EUE is equivalent to minimizing the sum of distances travelled by chargers in a scheduling cycle.

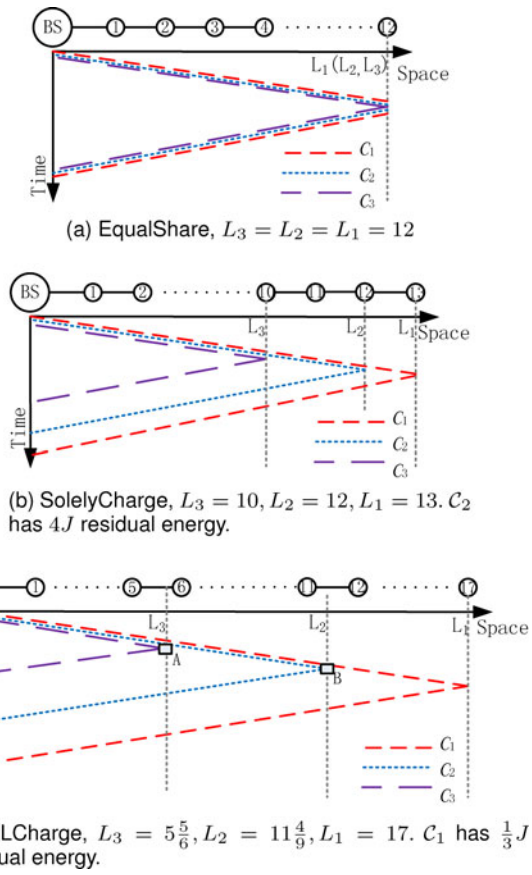


Fig. 1. Time-space views of three scheduling examples for scenario $K1K2K3$ where $x_i = i$, $b_i = 2\ J$, $P = 80\ J$, and $c = 3\ J/m$.

4.1 Motivational Scheduling Examples

We first show three examples to demonstrate the advantages of collaborative mobile charging and to motivate our algorithm design as well.

Suppose that we have the following WSN: for $i \geq 1$, node s_i is located at $(i, 0)$, $b_i = 2\ J$ (J = Joule), and $\tau_i = \tau$. We also have, at hand, three mobile chargers with $P = 80\ J$ and $c = 3\ J/m$. Since the total energy is fixed (which is $80\ J \times 3 = 240\ J$), EUE reaches its maximum when E^{pl} is the largest. Therefore, we just check which algorithm can cover the most sensor nodes.

Fig. 1 shows the time-space views of three simple scheduling heuristics. In the figures, we use L_i $(1 \leq i \leq 3)$ to represent the farthest distance that $\mathcal{C}_i$ travels away from the BS. We also let L_4 be 0 for compatibility.

2. One-dimensional WSNs have a broad array of applications, ranging from oil/gas/water pipeline monitoring [25], to driver-alert systems [26], to international border protection [27].

EqualShare: Each mobile charger transfers $2/3$ unit of energy to each sensor node, and each sensor is charged by all mobile chargers. Thus, 12 sensors can be covered, as shown in Fig. 1a.

SolelyCharge: Each sensor is charged by only one charger. For example, $\mathcal{C}_3$ charges sensor nodes from L_4 to L_3; $\mathcal{C}_2$ charges sensor nodes from L_3 to L_2; and so on. The variables L_3, L_2, and L_1 are carefully chosen, so that each charger returns to the BS with exactly zero energy. Fig. 1b demonstrates that 13 sensors can be covered.

CLCharge (short for CollaborativelyCharge): Each sensor is charged by only one charger, and energy transfer between chargers is allowed. For example, $\mathcal{C}_3$ charges sensors from BS to L_3, then transfers some energy to $\mathcal{C}_2$ and $\mathcal{C}_1$ at L_3, and finally returns to BS. Here, L_3 is carefully chosen, such that (i) $\mathcal{C}_2$ and $\mathcal{C}_1$ have full batteries after $\mathcal{C}_3$ transfers energy to them, and (ii) $\mathcal{C}_3$ returns to BS with exactly zero energy. In Fig. 1c, 17 sensors can be covered.

We notice that CLCharge covers more sensor nodes than the former two heuristics, which do not take advantage of collaboration. CLCharge also provides some intuitions for us to design an optimal scheduling solution to scenario $K1K2K3$, as we show below.

4.2 Our Solution: PushWait

As we mentioned earlier, given a fixed WSN satisfying $K1K2K3$, maximizing EUE is equivalent to minimizing the sum of all chargers' travelling distances. The key idea is that, at every possible moment, we let as few mobile chargers as possible carry the residual energy of all chargers, and move forward. CLCharge in Fig. 1c reflects this intuition: $\mathcal{C}_3$ turns around at $L_3 = 5\frac{5}{6}$, which is smaller than 13 in Fig. 1b. Therefore, the sum of total travelling distances in CLCharge is less than that in SolelyCharge, leading to a higher EUE. The reason that we can safely let $\mathcal{C}_3$ turn around at $L_3 = 5\frac{5}{6}$ is that $\mathcal{C}_2$ and $\mathcal{C}_1$ can carry the residual energy and move forward, instead of having all of the three not-full-battery mobile chargers move forward.

Let us go one step further and think whether we can improve CLCharge. We notice that, in Fig. 1c, when $\mathcal{C}_2$ (resp. $\mathcal{C}_1$) reaches L_3 on its way back to the BS, it has a positive amount of residual energy, since it has to return to the BS. In other words, $\mathcal{C}_2$ (resp. $\mathcal{C}_1$) carries this particular part of energy during its travelling from L_3 to L_2 (resp. L_1), and finally to L_3 again. How about letting $\mathcal{C}_3$ stop moving forward at a place L_3' closer to the BS than $L_3 = 5\frac{5}{6}$? In doing so, $\mathcal{C}_3$ can wait at a place with sufficient energy to support $\mathcal{C}_2$ and $\mathcal{C}_1$'s travelling from L_3' to the BS.

Based on the intuition, we design a scheduling algorithm, *PushWait*. Formally, suppose that we require M chargers to cover a given WSN, then each charger $\mathcal{C}_i$ in PushWait follows iterative process below:

1. $\mathcal{C}_i$ starts from the BS with a full battery, it then gets fully charged at locations $L_M, L_{M-1}, \ldots$, and L_{i+1}.
2. $\mathcal{C}_i$ charges sensor nodes between L_{i+1} and L_i. When it arrives at L_i, it charges $\mathcal{C}_{i-1}, \mathcal{C}_{i-2}, \ldots$, and $\mathcal{C}_1$, such that these $(i-1)$ chargers' batteries are full.
3. $\mathcal{C}_i$ waits at L_i. When all of $\mathcal{C}_1, \mathcal{C}_2, \ldots$, and $\mathcal{C}_{i-1}$ return to L_i, it evenly distributes its residual energy among these i chargers (including $\mathcal{C}_i$ itself).
4. On $\mathcal{C}_i$'s way back to the BS, it gets charged at locations $L_{i+1}, L_{i+2}, \ldots$, and L_M, which makes it have just enough energy to return to the BS.

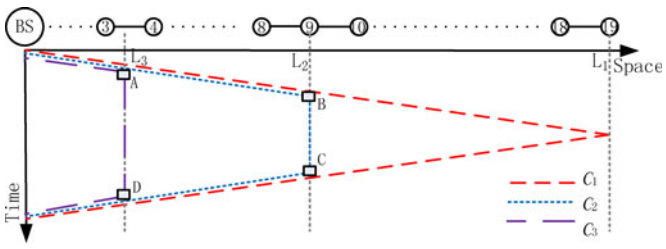


Fig. 2. Time-space view of PushWait for $K1K2K3$ with the same settings as in Fig. 1. We have $L_3 = 3\frac{1}{3}, L_2 = 9, L_1 = 19$. $\mathcal{C}_3$ has $14\ J$ residual energy.

Fig. 2 shows the result of applying PushWait to the aforementioned settings, where 19 sensor nodes are covered. Mobile chargers $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$ start from the BS with $P = 80\ J$ energy. At $L_3 = 3\frac{1}{3}$, both $\mathcal{C}_1$ and $\mathcal{C}_2$ have $80 - 3 \cdot L_3 = 70\ J$ energy, while $\mathcal{C}_3$ has $80 - 3 \cdot L_3 - 2 \times 3 = 64\ J$ energy, because it charges nodes s_1, s_2, and s_3. Then, we let $\mathcal{C}_3 \overset{10}{\rightarrow} \mathcal{C}_2$ and $\mathcal{C}_3 \overset{10}{\rightarrow} \mathcal{C}_1$. After this, $\mathcal{C}_3$ waits at L_3 with $44\ J$ energy. Similarly, after $\mathcal{C}_2$ charges nodes from s_4 to s_9, and charges $\mathcal{C}_1$ to its full battery at $L_2 = 9$, $\mathcal{C}_2$ waits at L_2 with $34\ J$ energy. When $\mathcal{C}_1$ returns to L_2 after charging nodes from s_{10} to s_{19}, as the reader can verify, it has exactly 0 energy. Then, we let $\mathcal{C}_2 \overset{17}{\rightarrow} \mathcal{C}_1$. Note that, $17\ J$ energy is just enough for $\mathcal{C}_1$ or $\mathcal{C}_2$ to move from L_2 to L_3. At L_3, we let $\mathcal{C}_3 \overset{10}{\rightarrow} \mathcal{C}_2$ and $\mathcal{C}_3 \overset{10}{\rightarrow} \mathcal{C}_1$. Again, note that, $10\ J$ energy is just enough for $\mathcal{C}_1$ or $\mathcal{C}_2$ to move from L_3 to the BS. When they return to the BS, only $\mathcal{C}_3$ has $14\ J$ residual energy.

The reason of naming this scheduling after "PushWait" is clear: each charger "pushes" some other chargers to move forward, and "waits" for their returns.

4.2.1 Parameter Optimization

Before we present how to optimize L_i $(1 \leq i \leq M)$ for maximizing EUE, we additionally assume that, from here to the end of Section 4, $\forall 1 \leq i \leq N$, $b_i = b$ and $x_i = i \cdot d$. That is, given d' distance, we approximately have d'/d sensors. This assumption is made for ease of representation. All of the following results and Theorems 1 and 2 are valid, even without this assumption; the corresponding analysis and proofs follow a similar routine as we use below, and are left to the reader.

Let us analyze the interval between L_{i+1} and L_i. $\mathcal{C}_i$ gets fully charged at L_{i+1} and reaches L_{i+1} with 0 energy on its way back to the BS. The full battery P is used up for the following reasons: (i) $\mathcal{C}_i$ charges sensors between L_{i+1} and L_i, (ii) $\mathcal{C}_i$ moves from L_{i+1} to L_i, (iii) $\mathcal{C}_i$ transfers some energy to $\mathcal{C}_1, \mathcal{C}_2, \ldots$, and $\mathcal{C}_{i-1}$ at L_i for the first time. Note that these $i-1$ chargers are fully charged at L_{i+1}, thus the energy transferred to them at L_i is exactly the energy consumed by their travellings from L_{i+1} to L_i; (iv) $\mathcal{C}_i$ transfers some energy to $\mathcal{C}_1, \mathcal{C}_2, \ldots$, and $\mathcal{C}_{i-1}$ at L_i for the second time, which

is just enough for them to travel from L_i to L_{i+1}, and (v) $\mathcal{C}_i$ moves from L_i to L_{i+1}. Combining above together, we have the following equations:

$$\begin{cases} 2icd(L_i - L_{i+1}) + b(L_i - L_{i+1})/d = P \ (1 \le i < M), \\ 2Mcd(L_M - 0) + b(L_M - 0)/d \le P. \end{cases} \quad (2)$$

The second formula is an inequality, since PushWait cannot always use up exactly the total energy of M chargers. We then have:

$$\begin{cases} L_1 = Nd, \\ L_i = Nd - \sum_{j=1}^{i-1} \frac{Pd}{2cd^2j+b} \qquad (2 \le i \le M). \end{cases} \quad (3)$$

The number of chargers M can be determined by: $L_M > 0$, $L_{M+1} \le 0$. We further have $E^{pl} = Nb$ and $E^{oh} = 2cd\sum_{i=1}^{M} L_i$. The duration of a charging round is Nd/v, and the scheduling cycle is τ.

4.2.2 Properties of PushWait

Theorem 1 (Optimality of PushWait). *Given a WSN $(\mathbf{X}, \mathbf{Y}, \mathbf{B}, \mathbf{T})$ and a charging model $(P, c, v, \eta_1, \eta_2)$, which satisfy conditions $K1K2K3$, PushWait is optimal, in the sense that it achieves the maximum EUE.*

Proof. We prove it with the assumption: $\forall 1 \le i \le N$, $b_i = b$ and $x_i = i \cdot d$. The proof without this assumption is similar, and is left to the reader.

Denote by $Distance(alg)$ the sum of travelling distances by all mobile chargers in a scheduling algorithm alg. As we mentioned before, it is sufficient to prove that $Distance(PushWait)$ is the minimum. Suppose that PushWait requires M mobile chargers to replenish the given WSN. We prove the theorem by mathematical induction on M.

$M = 1$. $Distance(PushWait) = 2L_1$, where L_1 is the length of the given WSN. We note that any scheduling algorithm $anyalg$ must have at least one charger to charge the farthest sensor node in the WSN, therefore, $Distance(anyalg) \ge 2L_1 = Distance(PushWait)$.

$M = 2$. (By contradiction) Suppose that PushWait is not optimal, and the optimal scheduling algorithm is OPT_2. Since one charger cannot cover the entire WSN, there are at least two chargers in OPT_2. One of them, say $\mathcal{C}'$, must charge the farthest sensor, thus it moves at least $2L_1$ distance. By definition, we should have $Distance(OPT_2) < Distance(PushWait) = 2L_1 + 2L_2$. So all the other chargers in OPT_2 cannot travel as far as L_2. However, according to our calculation of L_2 in PushWait, a charger with a full battery at L_2 can only charge the sensors between L_2 and L_1 and return to L_2 with 0 energy; then we know $\mathcal{C}'$ in OPT_2 can, by no means, reach L_1: a contradiction! Therefore, no such OPT_2 exists, and PushWait is optimal.

I.H.: PushWait is optimal for any $M < n$.

$M = n$. (By contradiction) Suppose that PushWait is not optimal, and the optimal scheduling algorithm is OPT_n. Imagine that a virtual base station BS' is located at L_n, then, OPT_n and PushWait require Q and $(n-1)P$ energy, respectively, to cover the sensors between L_n and L_1. By the induction hypothesis, $Q > (n-1)P$. Then, the task of OPT_n is to cover the sensors from BS to L_n and to deliver Q energy to L_n. It is then straightforward to see that, OPT_n requires at least n chargers to reach L_n; otherwise, the total residual energy of less than n chargers at L_n is definitely less than $(n-1)P$. Since $Q > (n-1)P$, OPT_n consumes more energy than PushWait: a contradiction! No such OPT_n exists, and PushWait is optimal. □

Theorem 2 (Coverage). *Given a sufficiently large WSN $(\mathbf{X}, \mathbf{Y}, \mathbf{B}, \mathbf{T})$ and a charging model $(P, c, v, \eta_1, \eta_2)$, which satisfy conditions $K1K2K3$, the maximum coverages of EqualShare, SolelyCharge, CLCharge, and PushWait are $P/2c$, $P/2c$, P/c, and infinity, respectively.*

Please refer to the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TC.2013.2297926, for the detailed proofs of Theorems 2, 3, 4 and 5.

5 $K1K2\overline{K3}$: 1D, Same Recharging Cycle, Energy Loss

In this section, we first introduce the observations that give us insights into scheduling for this new scenario, then we present our solution.

5.1 Observations

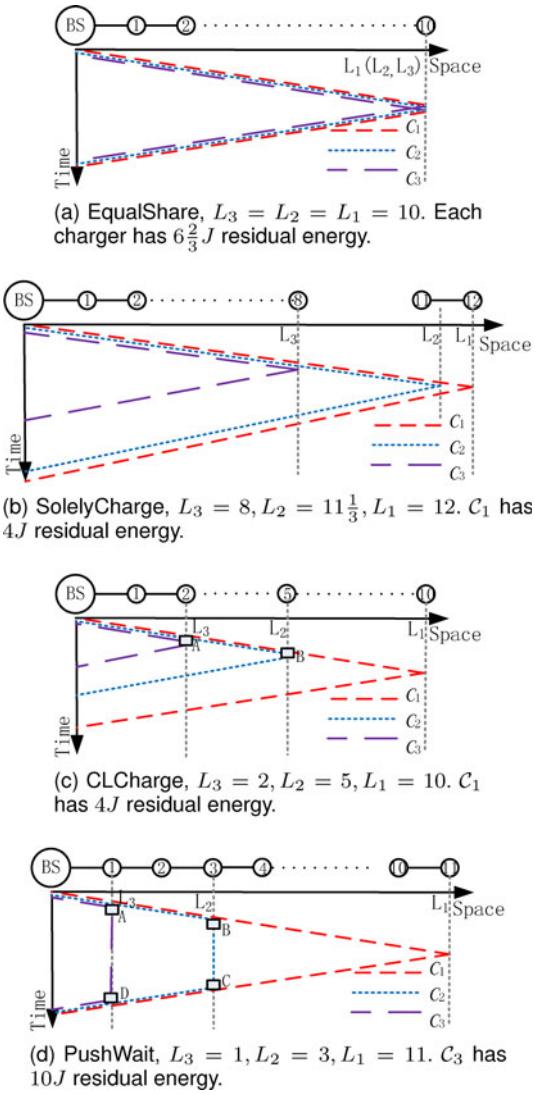
Mobile chargers' collaboration helps PushWait achieve optimality; however, when energy loss during energy transfer is not negligible, collaboration increases E^{oh}, and hence, may impair the EUE of PushWait. We use the following example to illustrate this observation.

We use the same problem settings as in Fig. 1, except that $\eta_1 = 0.5$ and $\eta_2 = 0.25$. Fig. 3 shows the time-space views of four scheduling algorithms. The farthest distance that $\mathcal{C}_i$ moves away from the BS is determined via the same analysis as before. For example, in Fig. 3a, $\mathcal{C}_3$ can cover only eight sensors, because $8c + 8c + 2 \times 8\eta_1 = 80$; $\mathcal{C}_1$ can cover only s_{12}, because it cannot return to the BS if it covers s_{13} as well; when all three chargers return to the BS, only $\mathcal{C}_1$ has $4\ J$ residual energy. The number of sensor nodes that can be covered in four algorithms is 10, 12, 10, and 11, respectively; the EUE of them is $\frac{10\times2}{240-20} \approx 0.091$, $\frac{12\times2}{240-4} \approx 0.102$, $\frac{10\times2}{240-4} \approx 0.085$, and $\frac{11\times2}{240-10} \approx 0.096$, respectively. Recall that PushWait is optimal for scenario $K1K2K3$; however, in scenario $K1K2\overline{K3}$, due to the energy loss between chargers, its EUE is only the second highest, while SolelyCharge achieves the highest EUE.

This example suggests to us that SolelyCharge may perform better than PushWait for $K1K2\overline{K3}$. The following theorem provides another property of SolelyCharge.

Theorem 3. (Optimality of SolelyCharge). *Given a WSN $(\mathbf{X}, \mathbf{Y}, \mathbf{B}, \mathbf{T})$ and a charging model $(P, c, v, \eta_1, \eta_2)$, satisfying conditions $K1K2$, if collaboration among chargers is not permitted, SolelyCharge is optimal.*

Similarly, we also find that, for scenario $K1K2\overline{K3}$, (i) PushWait remains optimal if $\eta_2 = 1$, and (ii) the coverage of PushWait is infinity, as the number of chargers



(a) EqualShare, $L_3 = L_2 = L_1 = 10$. Each charger has $6\frac{2}{3}J$ residual energy.

(b) SolelyCharge, $L_3 = 8, L_2 = 11\frac{1}{3}, L_1 = 12$. $\mathcal{C}_1$ has $4J$ residual energy.

(c) CLCharge, $L_3 = 2, L_2 = 5, L_1 = 10$. $\mathcal{C}_1$ has $4J$ residual energy.

(d) PushWait, $L_3 = 1, L_2 = 3, L_1 = 11$. $\mathcal{C}_3$ has $10J$ residual energy.

Fig. 3. Time-space views of four scheduling algorithms for $K1K2\overline{K3}$ where $x_i = i$, $b_i = 2\ J$, $P = 80\ J$, $c = 3\ J/m$, $\eta_1 = 0.5$, and $\eta_2 = 0.25$.

approaches infinity. The proofs are easy, and are left to the interested reader.

5.2 Our Solution: ηPushWait

Although SolelyCharge is optimal if collaboration is not allowed, it has limited coverage (see Theorem 2). PushWait is not optimal for scenario $K1K2\overline{K3}$, but it can cover a one-dimensional WSN of infinite length.

We therefore propose to combine SolelyCharge with PushWait to construct our solution $\eta PushWait$, which is better than both of them. Denote by $cg(alg, M)$ the coverage of a scheduling algorithm alg with M mobile chargers. For example, in scenario $K1K2\overline{K3}$, if we assume that, $\forall 1 \le i \le N$, $b_i = b$ and $x_i = i \cdot d$, following a similar analysis as in Section 4.2.1, we have

$$cg(SolelyCharge, M) = \sum_{i=1}^{M} \frac{\eta_1 d b^{i-1} P}{(2\eta_1 c d^2 + b)^i},$$
$$cg(PushWait, M) = \sum_{i=0}^{M-1} \frac{\eta_1 \eta_2 d P}{\eta_2 b + 2\eta_1 c d^2 (\eta_2 + i)}. \quad (4)$$

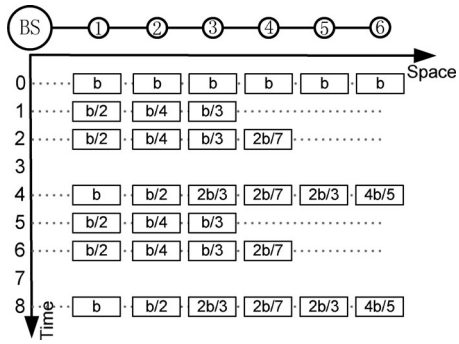


Fig. 4. A scheduling example for $K1\overline{K2}K3$ where $x_i = i$, $b_i = b$, $\tau_1 = 2$, $\tau_2 = 4$, $\tau_3 = 3$, $\tau_4 = 7$, $\tau_5 = 6$, and $\tau_6 = 5$. For instance, at time 2, we use PushWait to deliver $b/2$, $b/4$, $b/3$, and $2b/7$ energy to s_1, s_2, s_3, and s_4, respectively. The scheduling cycle of this example is 4.

Given a WSN and mobile chargers that satisfy $K1K2\overline{K3}$, let M' be the largest value of M that ensures $cg(SolelyCharge, M) \ge cg(PushWait, M)$, *i.e.*, $M' = arg\ max_{(cg(SolelyCharge,M) \ge cg(PushWait,M))} M$. Then, η PushWait can be constructed as follows. If the length of the given WSN is not greater than $cg(SolelyCharge, M')$, we use SolelyCharge; otherwise, we have the following strategy: using SolelyCharge to charge sensors between the BS and $cg(SolelyCharge, m)$, and using PushWait to charge the remaining sensors, where m $(1 \le m \le M')$ is a positive integer that maximizes the EUE of such a strategy.

6 $K1\overline{K2}K3$: 1D, Different Recharging Cycles, No Energy Loss

In this section, we first introduce a necessary condition of optimal solutions, for removing some unnecessary scheduling choices. We then present our solution ClusterCharging (β) and its properties. We also comment on how to deal with scenario $K1\overline{K2K3}$.

6.1 A Necessary Condition

When sensor nodes have different recharging cycles, a scheduling cycle may include multiple charging rounds, which greatly complicates the scheduling problem. Fig. 4 shows a scheduling example in scenario $K1\overline{K2}K3$. There are six sensors in the WSN; for $1 \le i \le 6$, $x_i = i$, $b_i = b$; $\tau_1 = 2$, $\tau_2 = 4$, $\tau_3 = 3$, $\tau_4 = 7$, $\tau_5 = 6$, and $\tau_6 = 5$. All sensor nodes are initialized to their full batteries at time 0. At time 1, we plan to charge s_1, s_2, and s_3. Since $\tau_2 = 4$ and s_2 has a full battery at time 0, s_2 needs only $b/4$ energy at time 1. So we employ PushWait to deliver $b/2$, $b/4$, and $b/3$ energy to s_1, s_2, and s_3, respectively, at time 1. In the scheduling example, there are also charging rounds at time points 2 and 4. From time 5, the three charging rounds between time points 1 and 4 are repeated.

Two important questions can be raised for such a scheduling example. First and foremost, how are we to go about characterizing its long-term EUE? Since the scheduling cycle is 4, we can use the EUE within a scheduling cycle to exactly represent the long-term EUE. Second, how to define the scheduling cycle? As we mentioned before, it is defined as the time interval between two consecutive time points

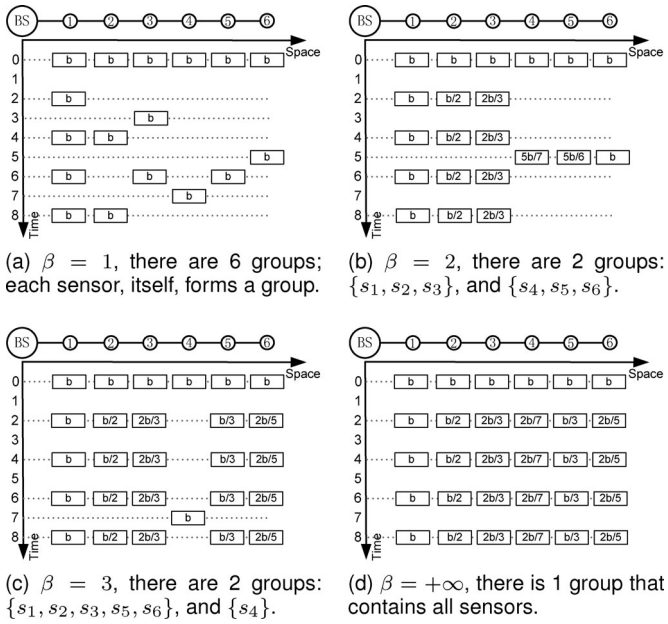


Fig. 5. Illustrations of ClusterCharging(β) for $K1\overline{K2}K3$ with the same settings as in Fig. 4.

when all sensors are fully charged. For example, the WSN in the figure is fully charged at time points 0, 4, 8, and so on, so the scheduling cycle is 4.

This example also suggests that the solution space of scenario $K1\overline{K2}K3$ could be extremely large. This is because, not only the length of the scheduling cycle, but also the set of sensor nodes to be charged in each charging round, has exponential possible choices.

When should we plan to make a charging round? The next theorem tells us that, we only need to start a charging round when there is at least one dying sensor node. For example, the charging round at time 1 in Fig. 4 is redundant, since no sensor nodes will run out of energy if the charging round is cancelled.

Theorem 4 (Necessary condition). *Given a node s that is x_s distance away from the BS, the battery capacity of s is b; using PushWait to deliver b energy to s one time achieves a higher EUE than using PushWait twice.*

6.2 Our Solution: ClusterCharging(β)

The last scheduling example motivates us to think about using PushWait as the basic routine to construct a solution to scenario $K1\overline{K2}K3$. Keeping Theorem 4 in mind, we have two intuitive ideas.

At one extreme, when we plan to recharge a sensor node, we want to transfer as much energy as possible to it, so as to increase the payload energy. Based on this intuition, we start a charging round only when there is at least one dying sensor node, and in this charging round, we only charge the dying sensor nodes. With the same settings as in Fig. 4, Fig. 5a shows such an example: a sensor node is charged only when it is dying.

At another extreme, when there is a charging round, we want to charge as many sensor nodes as possible, so as to increase the payload energy. Fig. 5d demonstrates this extreme case.

In fact, these two design options compete with each other; thus, we strive to strike a balance between them, and propose our solution *ClusterCharging(β)*.

In ClusterCharging(β), we first sort sensor nodes in decreasing order of their recharging cycles, then we divide them into groups in a first-fit manner, such that the ratio of the maximum recharging cycle to the minimum recharging cycle in each group is not greater than a given threshold, say β. Here, by "first-fit" we mean that, for each sensor node in the already-sorted order, we attempt to put it in the first group that can accommodate it; if this is not possible, this sensor forms a new group.

Then, we start a charging round only when there is at least one dying sensor node, and in this charging round, we employ PushWait to charge all sensor nodes in a group on the condition that this group contains at least one dying sensor node. Note that, in each charging round, different sensor nodes may need different amounts of energy, e.g., s_1, s_2, and s_3 require $b/2$, $b/4$, and $b/3$ energy, respectively, at time 1 in Fig. 4. Remember that PushWait can still achieve its optimality in each round, due to Theorem 1.

Let us take Fig. 5 for example. "$\beta = 1$" represents the extreme case in Fig. 5a, where each sensor node, itself, forms a group. Additionally, "$\beta = +\infty$" represents the other extreme case in Fig. 5d, where all sensor nodes form a single group. In Fig. 5b, we consider sensor nodes in the decreasing order of their recharging cycles, i.e., s_1, s_3, s_2, s_6, s_5, and s_4. Firstly, s_1 forms a group $\{s_1\}$; we then attempt to put s_3 into $\{s_1\}$, since $\tau_3/\tau_1 = 1.5 < \beta = 2$, it is feasible for them to be in the same group; the group $\{s_1, s_3\}$ can also accommodate s_2; when we want to put s_6 into $\{s_1, s_3, s_2\}$, as $\tau_6/\tau_1 > \beta$, s_6 forms a new group, and so on. In Fig. 5c, β is set to 3, resulting in two groups, i.e., $\{s_1, s_2, s_3, s_5, s_6\}$, and $\{s_4\}$.

Different values of β lead to different EUEs of ClusterCharging(β), and the optimal value of β varies with the parameters of a given MCS problem. Therefore, for a given MCS problem that satisfies $K1\overline{K2}K3$, we maximize the EUE of ClusterCharging(β) by searching the optimal β in range $[1, \frac{\tau_{max}}{\tau_{min}} + 1]$, where $\tau_{min} = min_{1 \le i \le N} \tau_i$, and $\tau_{max} = max_{1 \le i \le N} \tau_i$.

6.2.1 Properties of ClusterCharging(β)

The scheduling cycle of ClusterCharging(β) can be estimated as follows. Here, we assume that the recharging cycles of sensor nodes are integers, for ease of representation. In fact, τ_i is typically large enough for us to let $\tau_i' = \lfloor \tau_i \rfloor$, without incurring large estimation error.

When $\beta = 1$, our algorithm lazily charges each sensor node just before it runs out of energy. The scheduling cycle in this case is the *least common multiple* of $\tau_1, \tau_2, \ldots,$ and τ_N. Denote it as lcm. Then, the payload energy in a scheduling cycle is $E^{pl} = \sum_{i=1}^{N} \frac{lcm}{\tau_i} b_i$.

When $\beta = +\infty$, ClusterCharging(∞) charges all sensor nodes until their batteries are full every τ_{min} time, thus, the scheduling cycle is τ_{min}. In this case, the payload energy in a scheduling cycle is $E^{pl} = \sum_{i=1}^{N} \frac{\tau_{min}}{\tau_i} b_i$.

For $2 \le \beta < +\infty$, let the number of groups be g. As the ratio of the maximum recharging cycle to the minimum

recharging cycle in a group is not greater than β, we have $\tau_{min}\beta^g \leq \tau_{max}$. Thus, we have $g \leq \lfloor log_\beta(\tau_{max}/\tau_{min})\rfloor$, and the scheduling cycle of ClusterCharging(β) is roughly $\tau_{min}\beta^{\lfloor log_\beta(\frac{\tau_{max}}{\tau_{min}})\rfloor}$.

For example, the scheduling cycles in Figs. 5a, 5b, 5c, and 5d are 420, 10, 14, and 2, respectively.

Theorem 5 (Approx. ratio of ClusterCharging(β)). *Given a WSN $(\mathbf{X}, \mathbf{Y}, \mathbf{B}, \mathbf{T})$ and a charging model $(P, c, v, \eta_1, \eta_2)$, satisfying conditions $K1\overline{K2}K3$, the approximation ratio of ClusterCharging(β) is*

$$\frac{b_{min}(2cx_N + \sum_{i=1}^{N} b_i)}{P\tau_{max}k\sum_{i=1}^{N} b_i},$$

where $b_{min} = min_{i=1}^{N} b_i$, and

$$k = argmin_{\left(\sum_{i=1}^{k}\frac{1}{i} \geq \frac{2cx_N\tau_{max}+b_{min}}{P\tau_{max}}\right)} k.$$

6.3 Our Solution to $K1\overline{K2K3}$

For this scenario, we design a scheduling algorithm called *ηClusterCharging(β)*: sensor nodes are divided into groups in a similar way as ClusterCharging(β), but in each charging round, we employ ηPushWait instead of PushWait in ClusterCharging(β) to replenish sensor nodes. We omit the details due to space limitations.

7 $\overline{K1K2K3}$: The General 2D WSN

In this section, we propose a scheduling algorithm called *HηClusterCharging(β)* for chargers' scheduling in general 2D WSNs. HηClusterCharging(β) is a Hamiltonian cycle-based extension of ηClusterCharging(β). In the following sections, before explaining our algorithm in detail, we first introduce some necessary notations. We also present a concrete example for the reader to better understand the details.

7.1 Preliminaries

Finally, we come to the general 2D scenario. As we mentioned before, the condition $\overline{K2}$ forces us to divide sensor nodes into groups; the condition $\overline{K3}$ impairs the advantages of collaborative mobile charging, so we have to combine SolelyCharge with PushWait. But, what challenges does the condition $\overline{K1}$, i.e., 2D, pose to us?

The answer is that, given a set of sensor nodes to be replenished in a charging round, we must discover how to determine the charging sequence, so as to minimize the overhead energy.

In a charging round, denote by S the set of sensor nodes that are going to be charged. We construct a complete graph with vertices being $S \bigcup \{BS\}$, and edge weights being the euclidean distance between two corresponding vertices. Formally, we denote such a complete graph by $G[S]$, where the edge weight between two sensor nodes is defined as:

$$d(s_i, s_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}.$$

The edge weight between a sensor node and the BS is defined in a similar manner.

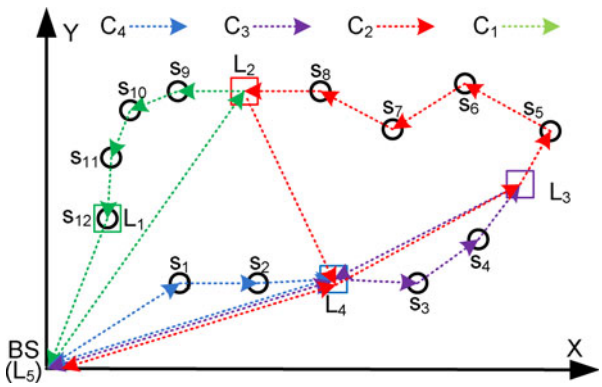


Fig. 6. Improving the scheduling results in a charging round through shortcutting: mobile chargers are no longer restricted to moving along a Hamiltonian cycle; they can take shortcuts when necessary.

A Hamiltonian cycle [28] is a cycle in an undirected graph that visits each vertex exactly once. If each edge in an undirected graph is associated with a weight, and the weight of a cycle is defined as the sum of edge weights belonging to the cycle, then, the minimum weight Hamiltonian cycle refers to the cycle with the minimum weight. Finding the minimum weight Hamiltonian cycle (aka. the Travelling Salesman Problem [28]) is proven to be NP-hard. We denote the minimum weight Hamiltonian cycle of $G[S]$ by H. Denote by $d_H(P_1, P_2)$ the sum of euclidean distances of line segments between two positions P_1 and P_2 on H. For example, $d_H(L_4, L_3) = d(L_4, s_3) + d(s_3, s_4) + d(s_4, L_3)$ in Fig. 6.

7.2 Our Solution: HηClusterCharging(β)

Generally speaking, in HηClusterCharging(β), we first divide sensor nodes into groups, and plan a charging round when there is at least one dying sensor node; in each charging round, we then try to find the minimum weight Hamiltonian cycle in the complete graph on the corresponding set of sensors and the BS; finally, we apply ηPushWait to the Hamiltonian cycle, and further improve the results through shortcutting.

Putting it formally, given a MCS problem instance, i.e., a WSN $(\mathbf{X}, \mathbf{Y}, \mathbf{B}, \mathbf{T})$ and a charging model $(P, c, v, \eta_1, \eta_2)$, the H$\eta$ClusterCharging($\beta$) works as follows.

Step 1. Sort sensor nodes in decreasing order of their recharging cycles, then divide sensor nodes into groups with respect to a threshold β, as in ClusterCharging(β).

Step 2. Decide the charging round plan, that is, decide the set of sensor nodes S that should be replenished in each charging round. We apply the following steps, i.e., 3-5, to each charging round.

Step 3. Construct a complete graph $G[S]$ and use the minimum spanning tree-based heuristic [29] to generate a Hamilton cycle H in $G[S]$.

Step 4. Randomly choose a direction for H. Suppose that we start from the BS and visit sensor nodes following the chosen direction along H. Without loss of generality, denote the sequence of sensor nodes we visit by s_1, $s_2, \ldots$, and $s_{|S|}$. We apply ηPushWait to H, which can be seen as a one-dimensional manifold [30], thus, we can obtain the number of required chargers M and the farthest positions that chargers will reach, i.e., L_1, $L_2, \ldots$,

and L_M. Again, without loss of generality, we let $L_{M+1} = (0, 0)$. (Note that, since we apply ηPushWait to this 1D manifold, each L_i will be located on an edge between two consecutive sensor nodes or at the location of a sensor node; particularly, L_1 is located at the same location with the farthest sensor node, i.e., $s_{|S|}$.)

Step 5. We improve ηPushWait through shortcutting. In this step, we only present how chargers take shortcuts and do not elaborate on the energy transfers between mobile chargers, for the sake of presentation brevity.

Step 5.1. For charger $\mathcal{C}_M$, it charges the sensor nodes between the BS and L_M, transfers energy to the other chargers at L_M, and waits at L_M for the other chargers' return. When $\mathcal{C}_M$ finishes its charging task, it can take a shortcut: it directly returns to the BS.

Step 5.2. For charger $\mathcal{C}_i$ $(1 \leq i \leq M-1)$, denote the current position of $\mathcal{C}_i$ as L_g. Before it finishes charging the sensor nodes between L_{i+1} and L_i:

- (Case 5.2.1) when $i+2 \leq g \leq M+1$, it directly takes a shortcut to $L_{j_{min}}$, where j_{min} satisfies:

$$j_{min} = argmin_{(d(L_g,L_j) \leq d_H(L_{j+1},L_j), i+1 \leq j \leq g-1)} j.$$

- (Case 5.2.2) When $g = i+1$, it begins to charge the sensor nodes between L_{i+1} and L_i.

After this, on its way back to the BS:

- (Case 5.2.3) For $i \leq g \leq M$, it directly takes a shortcut to $L_{j_{max}}$, where j_{max} satisfies:

$$j_{max} = argmax_{(d(L_g,L_j) \leq d_H(L_g,L_{g+1}), g+1 \leq j \leq M+1)} j.$$

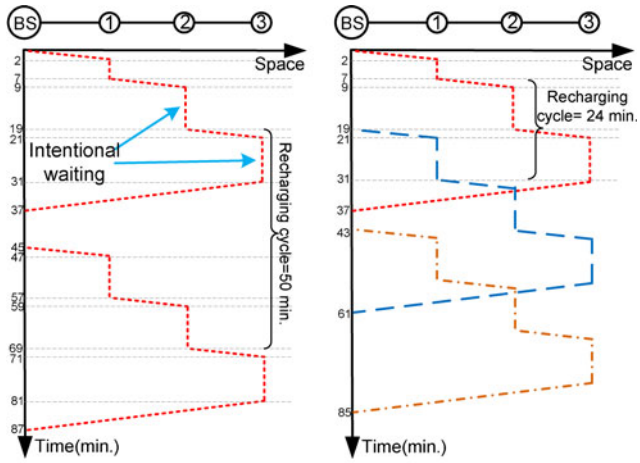
7.3 A Concrete Example

We provide an example in Fig. 6 as to better show the intuition behind our algorithm. Suppose that we have to replenish 12 sensor nodes in a charging round. Without loss of generality, the Hamiltonian cycle H we find is $BS - s_1 - s_2 - \cdots - s_{12} - BS$. After applying ηPushWait to this cycle, we know that, this round requires four chargers, and the farthest position of each charger is L_i $(1 \leq i \leq 4)$. In the following, we show when and how a charger can take a shortcut.

$\mathcal{C}_4$ is responsible for replenishing sensor nodes between L_5 (BS) and L_4. It can only take a shortcut after it completes its task, thus, its trajectory is $BS \rightarrow s_1 \rightarrow s_2 \rightarrow L_4 \Rightarrow BS$, where "$\rightarrow$" denotes a path segment along H, and "$\Rightarrow$" denotes a shortcut.

When $\mathcal{C}_3$ starts from the BS, since $g = 5$, the situation satisfies case 5.2.1 in HηClusterCharging(β), and we can determine $j_{min} = 4$, so it directly moves to L_4. Then, the situation satisfies case 5.2.2, thus, $\mathcal{C}_3$ charges sensor nodes between L_4 and L_3. The situation begins to satisfy case 5.2.3 when it arrives at L_3, as $d(L_3, L_5) > d_H(L_3, L_4)$, it does not have enough energy to move directly to the BS. (Please keep in mind that, according to ηPushWait, if $\mathcal{C}_3$ moves along H to L_4, it would have 0 energy at L_4.) Thus, the trajectory of $\mathcal{C}_3$ is $BS \Rightarrow L_4 \rightarrow s_3 \rightarrow s_4 \rightarrow L_3 \Rightarrow L_4 \Rightarrow BS$.

Following a similar argument, we can have the trajectory of $\mathcal{C}_2$: $BS \Rightarrow L_4 \Rightarrow L_3 \rightarrow s_5 \rightarrow s_6 \rightarrow s_7 \rightarrow s_8 \rightarrow L_2 \Rightarrow L_4 \Rightarrow BS$.



(a) Without SD. The first charging round (0 to 37 min.) synchronizes the battery levels through intentionally waiting.

(b) Without LC. Pipeline-like periodic charging rounds are scheduled to cope with short recharging cycle (24min $<$ 42min).

Fig. 7. Examples in a WSN satisfying $K1K2K3$, where $x_i = 10i$, $b_i = 2J$, $P = 200\ J$, $c = 3\ J/m$, $v = 5\ m/min.$, and the charger can transfer 0.2J to a node in 1 min.

When $\mathcal{C}_1$ is at L_5, since $d(L_5, L_2) < d_H(L_3, L_2)$, we have $j_{min} = 2$, and it takes a shortcut to L_2. After charging sensor nodes between L_2 and L_1, it arrives at L_1. As $d(L_1, L_5) < d_H(L_1, L_2)$, we have $j_{max} = 5$, thus, it directly returns to the BS. The trajectory of $\mathcal{C}_1$ is $BS \Rightarrow L_2 \rightarrow s_9 \rightarrow s_{10} \rightarrow s_{11} \rightarrow s_{12} \Rightarrow BS$.

8 Extensions and Remarks

In this section, we show how to apply our solutions to contexts without the *short duration* and *long cycle* assumptions. We also provide a few remarks on the relations between the proposed algorithms.

Without SD. When the duration of a charging is not negligible, we use the first charging round to synchronize the batter levels of all sensor nodes. The purpose of synchronization is to make sure that, each sensor node s_i would require exactly b_i amount of energy when the mobile charger approaches it in the following charging rounds. Since the energy transfer rate is fixed, we can modify our scheduling algorithms by adding a fixed charging duration at each sensor node.

We use Fig. 7a for illustration. In this example, the WSN satisfies conditions $K1K2K3$; three nodes are 10 meters apart. The battery capacity of each node is 2 J, the recharging cycle of each node is 50 min. Thus, the energy consumption rate is $0.04\ J/min$. The charger can transfer 0.2 J energy to a node in 1 min. Therefore, it takes 10 min to transfer 2 J to a node.

Suppose that when the charger arrives at s_1 for the first time, it has 1.2J residual energy; since $(2 - 1.2)/(0.2 - 0.04) = 5$, it takes 5 min for the charger to replenish s_1 to its full battery. When the charger arrives at s_2, it also has 1.2 J residual energy. Although the charger could finish charging s_2 in 5 min, *the charger should intentionally wait another 5 min before heading for* s_3, for the purpose of synchronizing energy levels among sensor nodes, as shown in Fig. 7a. In doing so, in the following charging rounds, each

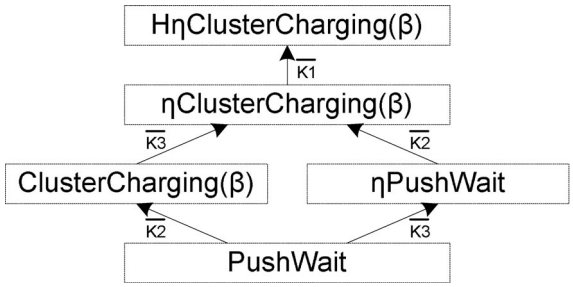


Fig. 8. The relations between our proposed algorithms.

sensor node will have exactly 0.4 J energy when the charger begins to recharge it, implying that it would take the same amount of time (i.e., 10 min) for the charger to recharging each node to its full battery. For instance, s_1, s_2, and s_3 are of full battery at the seventh, 19th, and 31st min, respectively; in the second charging round, they become fully charged at the 47th, 59th, and 71st min, respectively. We see that, the corresponding time interval for every sensor is 50 min, which is the recharging cycle of each sensor node.

Without LC. Generally speaking, when the recharging cycle of a sensor node is not longer than a charging round, we can adopt a pipeline-like solution. Fig. 7b shows an example. Since the recharging cycle is 24 min, the second (resp. third) charging round has to start at the 19th (resp. 43rd) min. It is not hard to see that, pipeline-like PushWait can still achieve optimality. An additional requirement of such a solution is that, it needs more chargers, e.g., two chargers are required in Fig. 7b.

Remarks on relations. As shown in Fig. 8, serving as the foundation of the entire set of algorithms, PushWait is proposed as an optimal algorithm for scenario $K1K2K3$. To cope with energy loss ($\overline{K3}$), we let PushWait and SolelyCharge compete with each other to generate ηPushWait; to overcome nonuniform recharging cycles, we divide sensor nodes into groups and propose ClusterCharging (β). When we are confronted with both of them ($\overline{K1K2}$), it is natural for us to put ClusterCharging(β) and ηPushWait together. For a general 2D scenario, we incorporate finding the minimum weight Hamiltonian cycle into ηClusterCharging(β), as to decide the visiting sequence of sensor nodes, and we also improve the scheduling results through shortcutting. It is worth mentioning that, HηClusterCharging(β) is generic: when we apply it to any other scenario (e.g., $K1K2K3$), it is equivalent to the corresponding algorithm (e.g., PushWait).

9 Performance Evaluation

Extensive simulations have been conducted to evaluate the performance of the proposed algorithms under different a variety of network settings.

9.1 Simulation Setup

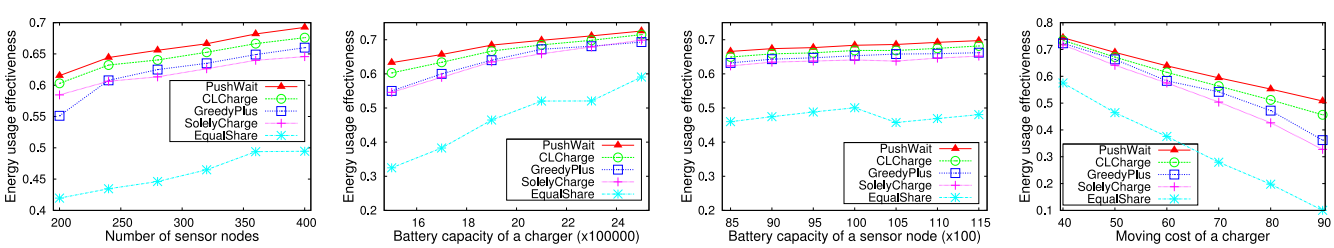
We assume that wireless sensor nodes are uniformly deployed over a 10 km×10 km two-dimensional square area. The base station BS is located at the bottom-left vertex of the square, and the coordinates of BS is $(0, 0)$. By default, the number of sensor nodes (N) is 400. Following similar settings in [21], we assume that sensor nodes are powered by a 1.5 V 2,000 mAh Alkaline rechargeable battery, then the battery capacity (b) is $1.5\text{V} \times 2\text{A} \times 3,600 \text{ sec} = 10.8KJ$. The battery capacity of a mobile charger (P) is 2,000 KJ; the moving speed of a charger (v) is 1 m/s; the charger's moving power consumption rate is 50 W, thus, the moving cost of a charger (c) is 50 J/m. The wireless charging efficiency (η_1) is by default 1.5 percent. The charging efficiency between mobile chargers (η_2) is 30 percent. (Since the energy transfer between chargers could be wired, the charging efficiency between chargers is much higher than wireless charging efficiency.)

We compare the proposed solutions with EqualShare, SolelyCharge, CLCharge, and $GreedyPlus$ [21]. The original version of GreedyPlus does not consider multiple chargers, and we tailored it to our scenarios: multiple mobile chargers are seen as one large charger, which adopts binary search to find a suitable target network lifetime. Note that, a sensor may be recharged several times in a charging round in GreedyPlus.

9.2 Results in the Scenario $K1K2K3$

Fig. 9 shows the performance comparisons of five algorithms. In this scenario, we randomly place sensors nodes along the x-axis ($K1$); recharging cycles are the same ($K2$); $\eta_1 = \eta_2 = 1$ ($K3$). Remember that only PushWait has an unlimited coverage, to make these algorithms comparable, the WSN length should be less than $P/2c$. This constraint is always respected in our parameter settings. We want to evaluate the impact of one parameter through varying this parameter while keeping all of the other parameters unchanged.

Generally speaking, as we theoretically demonstrated earlier, PushWait achieves the highest EUE among the five algorithms. CLCharge takes advantage of collaboration



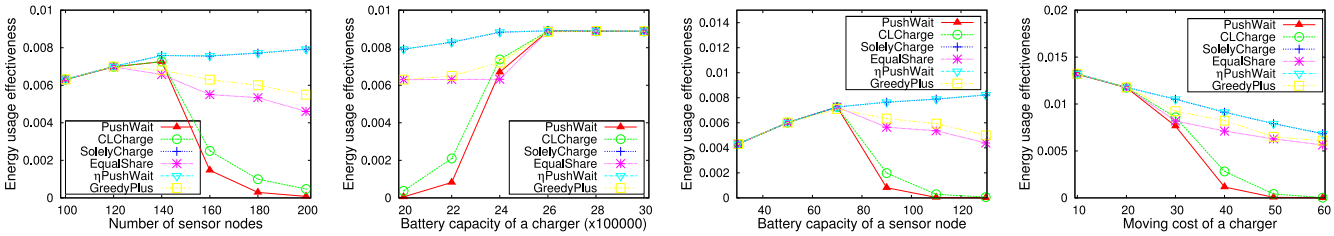
(a) Varying the number of sensor nodes

(b) Varying the battery capacity of a charger

(c) Varying the battery capacity of a sensor node

(d) Varying the moving cost of a charger

Fig. 9. Performance comparisons in scenario $K1K2K3$. PushWait is the optimal algorithm.



(a) Varying the number of sensor nodes (b) Varying the battery capacity of a charger (c) Varying the battery capacity of a sensor node (d) Varying the moving cost of a charger

Fig. 10. Performance comparisons in scenario $K1K2\overline{K3}$. ηPushWait is the optimal algorithm.

between chargers, so it has the second highest EUE. GreedyPlus greedily selects the next charging target, outperforming SolelyCharge and EqualShare. EqualShare has the worst performance.

In Fig. 9a, when the number of sensor nodes increases, since they are restricted to the 10 km $\times$ 10 km square, the density of sensor nodes also increases, so mobile chargers can transfer energy to more sensor nodes without incurring much moving cost. According to this, all of the five algorithms perform better when the number of sensor nodes goes up. The main reason for the relatively low EUE of EqualShare is that, every charger in EqualShare has to move to the farthest sensor node, and thus, the increase in the number of sensor nodes also leads to an increase in the amount of overhead energy.

In Fig. 9b, as the battery capacity of a charger increases, the EUEs of all five algorithms also increase. This is reasonable, since chargers can deliver energy from one position to another position with less moving cost. Similar trends are observed in Figs. 9c and 9d, where the EUEs of the five algorithms get larger with the increase in the battery capacity of a sensor node, and become smaller with the increase in the moving cost of a mobile charger, respectively.

9.3 Results in the Scenario $K1K2\overline{K3}$

Fig. 10 shows the performance comparisons of six algorithms. In this scenario, we randomly deploy sensor nodes along the x-axis ($K1$); recharging cycles are the same ($K2$); by default, $\eta_1 = 1.5\%$, and $\eta 2 = 30\%$ ($\overline{K3}$). The number of sensor nodes is, by default, 200.

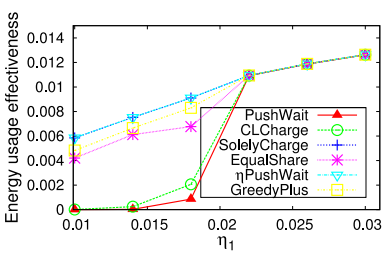
We make two general observations. First, when energy transfer is not perfect, collaboration between chargers brings about damage as well as benefit. In Fig. 10, PushWait has the worst performance among the six algorithms. Second, SolelyCharge always performs better than EqualShare, PushWait, GreedyPlus, and CLCharge. Remember that, if no collaboration is allowed, SolelyCharge is the optimal algorithm; therefore, SolelyCharge can always beat EqualShare. Why can SolelyCharge outperform PushWait and CLCharge? The collaboration between chargers wastes too much energy. It is then natural to think about combining SolelyCharge and PushWait. As it is expected, ηPushWait obtains a higher EUE than the other five algorithms in all settings.

We first look at Figs. 10b and 10d. All algorithms perform better when the battery capacity of a charger increases, because that, chargers could deliver more energy to sensors at the same cost than before. All algorithms perform worse when the moving cost of a charger increases. This is because, an increase in the charger's battery capacity or a decrease in the charger's moving cost always has the positive effect: reducing the amount of overhead energy.

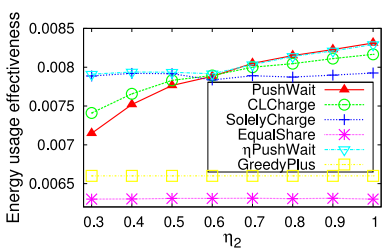
However, in Figs. 10a and 10c, although the trends of ηPushWait are clear, i.e., the EUE of ηPushWait increases when the number of sensor nodes or the battery capacity of a sensor node increases, the other algorithms do not have such a trend. For example, in Fig. 10a, the EUE of PushWait goes up when the number of sensor nodes increases from 100 to 140, and then goes down when the number of sensor nodes increases from 140 to 200. The main reason behind this phenomenon is that, the impact of the number of sensor nodes and the sensor node's battery capacity is not clear. Because both of them have dual influences: increasing the amounts of not only payload energy but also overhead energy. Therefore, the trends in Figs. 9a, 9c, 10a, and 10c are different.

9.4 Impact of Charging Efficiencies

We are also interested in the impact of charging efficiencies. Fig. 11a shows the case where η_2 is fixed, i.e., the charging efficiency between chargers is fixed. When we increase η_1,



(a) Varying η_1 while keeping $\eta_2 = 0.3$.

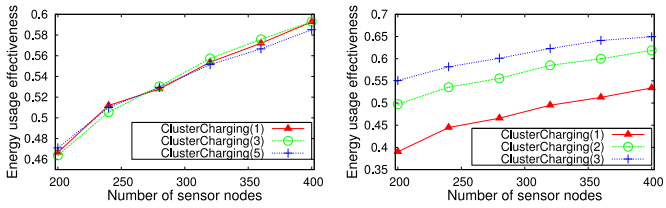


(b) Varying η_2 while keeping $\eta_1 = 0.015$.

η_2 \ η_1	0.010	0.014	0.018	0.022	0.026	0.030
0.3	107.5	69.9	32.4	140.3	180.1	180.1
0.4	107.2	69.8	32.5	140.5	160.2	200.0
0.5	107.1	69.0	28.6	160.4	200.0	200.0
0.6	107.1	28.4	26.0	180.2	200.0	200.0
0.7	11.9	8.1	4.2	200.0	200.0	200.0
0.8	1.0	1.4	1.0	200.0	200.0	200.0
0.9	1.0	1.5	1.0	200.0	200.0	200.0
1.0	1.0	1.6	1.0	200.0	200.0	200.0

(c) The optimal value of m in ηPushWait.

Fig. 11. Impact of charging efficiencies and the optimal values of m in ηPushWait.



(a) The case when all τ_i's are uniformly generated between 1 and 6.

(b) The case when all τ_i's are uniformly generated between 3 and 8.

Fig. 12. Performance of ClusterCharging(β) in scenario $K1\overline{K2}K3$. The optimal β in (b) is 3.

the energy loss during wireless charging becomes less, so the EUE of each algorithm gets larger. Note that, it is different from the cases in Figs. 9c and 10c. Here, although the energy transferred to each sensor node decreases, the overall payload energy is unchanged; while in the previous cases, when the battery capacity of a sensor node decreases (taking the reverse direction of b-axis in Figs. 9c and 10c, the energy delivered to each sensor node also decreases, but meanwhile, the total payload energy also decreases.

Let us take a close look at Fig. 11b. There are three interesting observations. First, since there is no energy transfer between chargers in SolelyCharge, GreedyPlus and EqualShare, the EUEs of them remain unchanged when η_2 increases. Second, CLCharge has a higher and lower EUE than PushWait when $\eta_2 \leq 0.6$ and $\eta_2 \geq 0.6$, respectively. The rationale behind this phenomenon is that, the total energy exchanged between chargers in CLCharge is less than that in PushWait, thus, CLCharge may perform better than PushWait if η_2 is small. Third, ηPushWait always has the best performance, because it takes advantage of SolelyCharge when $\eta_2 \leq 0.6$ and takes advantage of PushWait when $\eta_2 > 0.6$, as demonstrated in the figure.

We further show the average optimal value of m in ηPushWait in Fig. 11c. Remember that there are 200 sensor nodes in our example. As a whole, if $\eta_1 \leq 0.018$, less sensor nodes are replenished by SolelyCharge due to the increase of η_2; otherwise, more sensor nodes are covered by SolelyCharge due to the increase of η_2. However, when η_1 increases, the trend is not clear. The reason is that, as shown in Fig. 11a, when η_1 increases, both of PushWait and SolelyCharge get a higher EUE, therefore, we cannot find a clear trend in the change of m when η_1 increases.

9.5 Results in the Scenario $K1\overline{K2}K3$

Figs. 12a and 12b show the performance of ClusterCharging (β) under two different settings, respectively. In this scenario, we randomly deploy sensor nodes along the x-axis ($K1$); the recharging cycles of sensor nodes are uniformly generated from two ranges: $[1, 6]$ and $[3, 8]$ ($\overline{K2}$); $\eta_1 = \eta 2 = 1$ ($K3$). (The result in the scenario $K1\overline{K2}K3$ is similar to the results in this scenario, and it is omitted due to space limitations.)

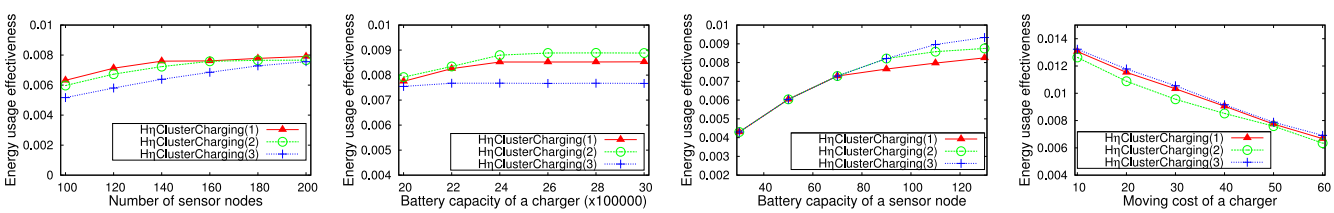
Since τ_{max}/τ_{min} is 6 in Fig. 12a and $6/3$ in Fig. 12b, we show the performance of ClusterCharging(β) with $\beta = 1$ to 5 in Fig. 12a and $\beta = 1$ to 3 in Fig. 12b. We notice that, ClusterCharging(β) with three different β's perform almost the same in Fig. 12a, while ClusterCharging(3) outperforms the other two algorithms in Fig. 12b. The main reason is that, the relative gap between recharging cycles in Fig. 12a is large, while the relative gap in Fig. 12b is small. For example, if we use ClusterCharging(5) to replenish the WSN in Fig. 12a, then the energy we have to transfer to each sensor node varies from $1.8KJ$ ($b/6$) to $10.8KJ$ (b). We can see that, some sensor nodes just need a small amount of energy. However, if we use ClusterCharging(5) to replenish the WSN in Fig. 12b (note that, ClusterCharging(5) is equivalent to ClusterCharging(3) for the setting in Fig. 12b, then the energy we have to transfer to each sensor node varies only from $4.05KJ$ ($3b/8$) to $10.8KJ$ (b). Therefore, ClusterCharging(5) can have the best performance in Fig. 12b.

9.6 Results in General 2D WSNs

Fig. 13 shows the performance of HηClusterCharging(β) in a general 2D WSN, where we randomly place sensor nodes within the square field ($\overline{K1}$); the recharging cycles of sensor nodes are uniformly generated from $[1, 6]$ ($\overline{K2}$); and, by default, $\eta_1 = 1.5\%$, and $\eta 2 = 30\%$ ($\overline{K3}$). In Hη ClusterCharging(β), we adopt the minimum spanning tree-based method to generate the minimum weight Hamiltonian cycle.

On average, the performance of HηClusterCharging(β) becomes better when the respective parameter varies in Figs. 13a, 13b,13c, and becomes worse when the moving cost of a charger increases in Fig. 13d. In each figure, we show the EUEs of HηClusterCharging(β) with three different β's, i.e., 1, 2, and 3. We notice that, in different settings, different β achieves the best performance. For example, HηClusterCharging(1), HηClusterCharging(2), HηClusterCharging(3), and HηClusterCharging(3) have the best performance in Figs. 13a, 13b, 13c, 13d, respectively.

In Fig. 13a, the performance gap shrinks as the number of sensor nodes increases. This is because, the difference between HηClusterCharging(β) with different β's is reduced when the density of sensor nodes increases. We



(a) Varying the number of sensor nodes

(b) Varying the battery capacity of a charger

(c) Varying the battery capacity of a sensor node

(d) Varying the moving cost of a charger

Fig. 13. Performance of HηClusterCharging(β) in general 2D WSNs.

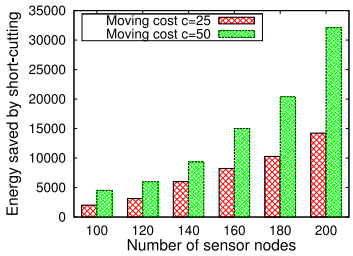


Fig. 14. The benefit of shortcutting.

also find that, the impact of a charger's battery capacity on the performance gap is greater than that of a sensor node's battery capacity or a charger's moving cost, as shown in Figs. 13b, 13c, 13d.

We are also interested in investigating how much benefit shortcutting brings about. The results are shown in Fig. 14. When the moving cost of a mobile charger (c) is fixed, if the number of sensor nodes increases, the energy saved by shortcutting also increases. This is reasonable, since an increase in the number of sensor nodes results in another increase in the number of mobile chargers required, thus, more chargers may take shortcuts when necessary. When the moving cost doubles, we find the energy saved also doubles or even trebles. This is because, when moving cost increases, the number of chargers required also increases.

In summary, numerical results confirm the advantages of the collaborative mobile charging paradigm, and the proposed set of scheduling algorithms can improve the energy usage effectiveness over the non-collaboration schemes significantly.

10 Conclusions

In this paper, we study the mobile charging scheduling problem in WSNs. To improve the energy usage effectiveness and charging coverage, we introduce the collaborative mobile charging paradigm. We design a set of scheduling algorithms for five different scenarios that cover the problem space, i.e., PushWait for $K1K2K3$, ηPushWait for $K1K2\overline{K3}$, ClusterCharging(β) for $K1\overline{K2}K3$, ηClusterCharging(β) for $K1\overline{K2K3}$, and HηClusterCharging(β) for $\overline{K1K2K3}$. To highlight the effectiveness of our algorithms, as well as the benefit of collaborative mobile charging, we conduct extensive performance evaluations, the results of which validate the advantages of our algorithms.

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful suggestions. This work was supported in part by NSFC Grants (No. 61073028, No. 61202113, and No. 61321491, 91218302), Key Project of Jiangsu Research Program Grant (No. BE2013116), Jiangsu NSF Grant (No. BK2011510), College graduate research and innovation project of Jiangsu Grant (No. CXZZ12_0055), Program A for outstanding PhD candidate of Nanjing University (No. 201301A08), and US National Science Foundation (NSF) grants (ECCS 1231461, ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167), EU FP7 IRSES MobileCloud Project Grant (No. 612212).

References

[1] J. Panchard, S. Rao, M.S. Sheshshayee, P. Papadimitratos, S. Kumar, and J.-P. Hubaux, "Wireless Sensor Networking for Rain-Fed Farming Decision Support," *Proc. ACM SIGCOMM Workshop Networked Systems for Developing Regions (NSDR)*, pp. 31-36, 2008.

[2] L. Yu, N. Wang, and X. Meng, "Real-Time Forest Fire Detection with Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Wireless Comm., Networking and Mobile Computing (WiCOM)*, pp. 1214-1217, 2005.

[3] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Health Monitoring of Civil Infrastructures Using Wireless Sensor Networks," *Proc. ACM/IEEE Sixth Int'l Conf. Information Processing in Sensor Networks (IPSN '07)*, pp. 254-263, 2007.

[4] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks*, vol. 38, no. 4, pp. 393-422, 2002.

[5] X. Jiang, J. Polastre, and D. Culler, "Perpetual Environmentally Powered Sensor Networks," *Proc. ACM/IEEE Fourth Int'l Symp. Information Processing in Sensor Networks (IPSN'05)*, pp. 463-468, 2005.

[6] A. Cammarano, C. Petrioli, and D. Spenza, "Pro-Energy: A Novel Energy Prediction Model for Solar and Wind Energy-Harvesting Wireless Sensor Networks," *Proc. IEEE Ninth Int'l Conf. Mobile Adhoc and Sensor Systems (MASS'12)*, pp. 75-83, 2012.

[7] A. Kansal and M.B. Srivastava, "An Environmental Energy Harvesting Framework for Sensor Networks," *Proc. IEEE Int'l Symp. Low Power Electronics and Design (ISLPED '03)*, pp. 481-486, 2003.

[8] T. Voigt, H. Ritter, and J. Schiller, "Utilizing Solar Power in Wireless Sensor Networks," *Proc. IEEE 28th Ann. Int'l Conf. Local Computer Networks (LCN '03)*, pp. 416-422, 2003.

[9] A. Kansal, J. Hsu, S. Zahedi, and M.B. Srivastava, "Power Management in Energy Harvesting Sensor Networks," *ACM Trans. Embedded Computing Systems*, vol. 6, article 32, Sept. 2007.

[10] Y. Shi, L. Xie, Y. Hou, and H. Sherali, "On Renewable Sensor Networks with Wireless Energy Transfer," *Proc. IEEE INFOCOM*, pp. 1350-1358, 2011.

[11] A. Kurs, A. Karalis, R. Moffatt, J.D. Joannopoulos, P. Fisher, and M. Soljačić, "Wireless Power Transfer via Strongly Coupled Magnetic Resonances," *Science*, vol. 317, no. 5834, pp. 83-86, 2007.

[12] K. Kang, Y.S. Meng, J. Bréger, C.P. Grey, and G. Ceder, "Electrodes with High Power and High Capacity for Rechargeable Lithium Batteries," *Science*, vol. 311, no. 5763, pp. 977-980, 2006.

[13] L. Xie, Y. Shi, Y.T. Hou, W. Lou, H.D. Sherali, and S.F. Midkiff, "On Renewable Sensor Networks with Wireless Energy Transfer: The Multi-Node Case," *Proc. IEEE Ninth Ann. Comm. Soc. Conf. Sensor, Mesh, and Ad Hoc Comm. and Networks (SECON)*, pp. 10-18, 2012.

[14] L. Fu, P. Cheng, Y. Gu, J. Chen, and T. He, "Minimizing Charging Delay in Wireless Rechargeable Sensor Networks," *Proc. IEEE INFOCOM*, pp. 2922-2930, 2013.

[15] T.C. Chen, T.S. Chen, and P.W. Wu, "On Data Collection Using Mobile Robot in Wireless Sensor Networks," *IEEE Trans. Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 41, no. 6, pp. 1213-1224, Nov. 2011.

[16] S. Zhang, J. Wu, and S. Lu, "Collaborative Mobile Charging for Sensor Networks," *Proc. IEEE Ninth Int'l Conf. Mobile Ad-Hoc and Sensor Systems (MASS)*, pp. 84-92, 2012.

[17] S. Bhattacharya, H. Kim, S. Prabh, and T. Abdelzaher, "Energy-Conserving Data Placement and Asynchronous Multicast in Wireless Sensor Networks," *Proc. ACM First Int'l Conf. Mobile Systems, Applications, and Services (MobiSys '03)*, pp. 173-185, 2003.

[18] A. Dunkels, F. Österlind, and Z. He, "An Adaptive Communication Architecture for Wireless Sensor Networks," *Proc. ACM Fifth Int'l Conf. Embedded Networked Sensor Systems (SenSys '07)*, pp. 335-349, 2007.

[19] W. Wang, V. Srinivasan, and K.-C. Chua, "Using Mobile Relays to Prolong the Lifetime of Wireless Sensor Networks," *Proc. ACM MobiCom '05*, pp. 270-283, 2005.

[20] "Wireless Power Consortium," http://www.wirelesspowerconsortium.com/, 2014..

[21] Y. Peng, Z. Li, W. Zhang, and D. Qiao, "Prolonging Sensor Network Lifetime through Wireless Charging," *Proc. IEEE 31st Real-Time Systems Symp. (RTSS '10)*, pp. 129-139, 2010.

[22] Z. Li, Y. Peng, W. Zhang, and D. Qiao, "Study of Joint Routing and Wireless Charging Strategies in Sensor Networks," *Proc. Fifth Int'l Conf. Wireless Algorithms, Systems, and Applications (WASA '10)*, pp. 125-135, 2010.

[23] B. Tong, Z. Li, G. Wang, and W. Zhang, "How Wireless Power Charging Technology Affects Sensor Network Deployment and Routing," *Proc. IEEE 30th Int'l Conf. Distributed Computing Systems (ICDCS)*, pp. 438-447, 2010.
[24] S. He, J. Chen, F. Jiang, D.K. Yau, G. Xing, and Y. Sun, "Energy Provisioning in Wireless Rechargeable Sensor Networks," *Proc. IEEE INFOCOM*, pp. 2006-2014, 2011.
[25] I. Jawhar, N. Mohamed, and K. Shuaib, "A Framework for Pipeline Infrastructure Monitoring Using Wireless Sensor Networks," *Proc. Wireless Telecomm. Systems (WTS '07)*, pp. 1-7, 2007.
[26] I. Jawhar and N. Mohamed, "A Hierarchical and Topological Classification of Linear Sensor Networks," *Proc. Wireless Telecomm. Symp. (WTS '09)*, pp. 1-8, 2009.
[27] A. D'Costa, V. Ramachandran, and A.M. Sayeed, "Distributed Classification of Gaussian Space-Time Sources in Wireless Sensor Networks," *IEEE J. Selected Areas in Comm.*, vol. 22, no. 6, pp. 1026-1036, Aug. 2004.
[28] T. H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press, 2001.
[29] V.V. Vazirani, *Approximation Algorithms*. Springer, 2003.
[30] J.M. Lee, *Introduction to Topological Manifolds*. Springer-Verlag, 2000.

Sheng Zhang received the BS degree from Nanjing University in 2008. He is currently working toward the PhD degree in the Department of Computer Science and Technology, Nanjing University. He is also a member of the State Key Laboratory for Novel Software Technology. His research interests include network virtualization, cloud/service computing, and mobile networks. To date, he has published more than 15 papers, including those appeared in *IEEE Transactions on Parallel and Distributed Systems*, ACM MobiHoc, and IEEE INFOCOM. He received the Best Paper Runner-Up Award from IEEE MASS 2012. He is a student member of the IEEE.

Jie Wu (F'09) is the chair and a Laura H. Carnell professor in the Department of Computer and Information Sciences at Temple University. Prior to joining Temple University, he was a program director at the US National Science Foundation and a Distinguished Professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. He regularly published in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including *IEEE Transactions on Computers*, *IEEE Transactions on Service Computing*, and *Journal of Parallel and Distributed Computing*. He was a general co-chair/chair for IEEE MASS 2006 and IEEE IPDPS 2008 and a program co-chair for IEEE INFOCOM 2011. Currently, he is serving as a general chair for IEEE ICDCS 2013 and ACM MobiHoc 2014, and a program chair for CCF CNCC 2013. He was an IEEE Computer Society distinguished visitor, ACM distinguished speaker, and a chair for the IEEE Technical Committee on Distributed Processing (TCDP). He received the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award. He is a CCF distinguished speaker and a fellow of the IEEE.

Sanglu Lu received the BS, MS, and PhD degrees from Nanjing University in 1992, 1995, and 1997, respectively, all in computer science. She is currently a professor in the Department of Computer Science and Technology and the State Key Laboratory for Novel Software Technology. Her research interests include distributed computing, wireless networks, and pervasive computing. She has published more than 80 papers in refereed journals and conferences in the above areas. She is a member of the IEEE.