Optimizing Charging Locations and Charging Time for Energy Depletion Avoidance in Wireless Rechargeable Sensor Networks

Tran Thi Huong[†]*, Huynh Thi Thanh Binh *, Phi Le Nguyen*, Doan Cao Thanh Long *, Vuong Dinh An *, Le Trong Vinh †
*School of Information and Communication Technology, Hanoi University of Science and Technology, Hanoi, Vietnam
†Vietnam National University, University of Science, Hanoi
Email: *{binhht,lenp}@soict.hust.edu.vn,†,{huongtt9451,vinhlt}@vnu.edu.vn
* {an.vd180003, long.dct162513}@sis.hust.edu.vn

***Abstract*—In recent years, Wireless Rechargeable Sensor Networks, which exploit wireless energy transfer technologies to address the energy constraint problem in traditional Wireless Sensor Networks, has emerged as a promising solution. There are two important factors that affect the performance of a charging process: charging path and charging time. In the literature, many studies have been done to propose efficient charging algorithms. However, most of the existing works focus only on optimizing the charging path. In this paper, we are the first one to jointly take into account both the charging path and charging time. Specifically, we aim at determining the optimal charging path and the charging time at each charging location to minimize the number of dead nodes. We first mathematically formulate the problem under mixed integer and linear programming. Then, we propose a periodic charging scheme, which is based on the Greedy and Genetic algorithm approaches. The experiment results show that our proposed the algorithm reduces significantly the number of dead nodes compared to a relevant benchmark.**

***Index Terms*—Wireless rechargeable sensor networks, node failure avoidance, Genetic algorithm, charging algorithm.**

I. Introduction

Nowadays, wireless sensor networks (WSNs) play an important role in the Internet of things (IoT) thanks to the benefits they bring in many areas such as smart traffic applications, patient monitoring in the hospital, battlefield surveillance and so on [1], [2]. A WSN is comprised of sensor nodes deployed over a region of interest to monitor and control the physical environment. In WSNs, every sensor node simultaneously accomplishes two demanding tasks: sensing and communicating. In many applications, the network cannot achieve its objective if all the nodes cannot sense or report the sensed data. In such scenarios, the death of even only one node may cause the network to operate un-functionally [3]. For example, in an earthquake forecasting WSN, the death of a sensor locating at a critical region (e.g., the center of the earthquake) may reduce the forecast accuracy significantly. As we don't know in advance the critical region, all sensors need to operate functionally at all the time. Therefore, one of the most critical issues in WSN is to avoid the energy depletion of the nodes.

In recent years, advancement in wireless power transmission technology has emerged as a promising solution for the energy depletion issue. The technology allows us to replenish the energy of sensors for depletion avoidance. As a result, it has brought about a new generation of wireless sensor networks called wireless rechargeable sensor networks (WRSNs, for short) [4]. WRSNs consist of a mobile charger (MC) that is a robot carrying wireless power charger. Moreover, each sensor node in WRSN is equipped with a wireless energy receiver. The MC travels around the network and charges the sensors.

The performance of a charging algorithm is decided by two important factors. The first factor is the charging path which is a sequence of charging locations the MC will stop; the second factor is the charging time which is the duration the MC stops at each charging location and charges the sensors. In the literature, many efforts have been devoted to optimizing the charging path. The authors in [5] focused on optimizing charging path to maximizing the ratio of the mobile charger' vacation time at the depot over the cycle time. Liguang Xie et. al. in [6] studied how to minimize the number of charging locations while ensuring all sensors are charged.

However, none of the existing works addresses the charging time problem. Most of the charging schemes proposed so far leverage the fully charging approach where sensors are always charged to the maximum battery capacity [7][8], or the partial charging approach where sensors are charged a fixed amount of energy [9]. It is worth to note that the charging time is very importance factor that decide how much the charging algorithm can prolong the network life time. A too long charging time may cause the uncharged sensors to have to wait for too long, thus they may die before charging. In the contrast, a too short charging time may lead to the charged sensors not having enough energy to operate until the next charging round. The fully charging and fixed amount charging approaches may be acceptable with the very small networks where the number of nodes needed to be charged is insignificant. However, when the network grows and the number of nodes need to be charged becomes large, these approaches may suffer from many dead nodes caused by too long waiting time.

In this paper, we are the first one addressing at the same time the optimization problem of both the charging path and

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

the charging time. Specifically, we study how to minimize the number of dead nodes in each charging round when the battery capacity of the MC is limited. To this end, we first formulate the problem under a mixed-integer linear programming model. Then, we propose an approximation algorithm that consists of two phases. The first phase is to determine the optimal charging locations, and the second phase is to determine the charging time the MC spends at each charging location. Our contributions are as follows:

- To the best of our knowledge, this work is the first attempting to propose a charging scheme which jointly considers both the charging path and the charging time.
- We mathematically formulate the problem of minimizing the number of failure nodes as a linear programming model.
- We propose an approximation solution to solve the concerned problem.
- We conduct experiments to show that our proposed algorithm outperforms existing research on similar problems.

The rest of the paper is constructed as follows. Section II introduces the related works. Section III describes the network model and provides the notations and concepts used throughout this paper. Section IV presents our proposed greedy algorithm, and Section V shows our genetic approach based algorithm. Section VI evaluates and compares the performance of our proposed algorithms with the existing research. Finally, Section VII concludes the paper.

II. Related Work

The studies about the charging scheme problem in WRSN can be divided into two main categories: on-demand charging and periodic charging.

In the periodic charging model, the MC follows a predetermined charging plan to charge the sensors. The authors in [5] addressed the charging path optimization. The authors have demonstrated that their problem is equivalent to the traveling salesman problem. Therefore, the optimal path for MC is a Hamilton cycle with the least cost of visiting all the nodes. In [4], the authors presented an improvement of [5] with finite traveling energy of MC and different average energy consumption rate of each node. The authors investigated three scenarios and applied Genetic Algorithm (GA) in combination with Particle Swarm Optimization (PSO). The authors in [10] presented a periodic charging scheme for all sensor nodes by designing a charging path travels each sensor to minimize the number of dead nodes. They also proposed an approximation algorithm that consists of two-phase. Phase one determines a charging path that prioritize the sensors with the lowest lifetime first. Phase two finds the charging time duration for each sensor of the charging path given in phase one. The authors in [11] proposed a model to maximize the network lifetime by determining the optimal travel path and the optimal charging time for each sensor. Specifically, they considered both the remaining power of sensors and the distance to the MC to choose the optimal travel path by a greedy algorithm. After that, the charging time for each sensor is calculated based on linear programming. In [12], the authors suggested a model to optimize the MC's travel distance by determining the charging locations based on dividing network into grids. The MC departs from the charging station in the center of the network, travels through the charging points one-by-one, from the innermost rings to the outermost rings. During the travel, the MC determines the next charging location by applying the nearest-neighbor algorithm.

Regarding the on-demand charging model, the work in [13] suggested a model based on linear programming aiming to minimize the total charging delay of sensors. Moreover, the authors applied a gravitational search algorithm in which an agent and its weight depicted each sensor and the waiting time, respectively. In [14], the authors proposed a strategy to find the charging path for MC and charging time for each sensor to maximize the network lifetime. They applied a greedy algorithm depend on the lifetime of each node and conducted experiments on a real system to evaluate its effectiveness. The authors in [8] proposed an online algorithm called Invalid Node Minimized Algorithm (INMA) to optimize the waiting queue. The algorithm introduced a strategy to select the next charged nodes in the MC's travel path that minimize the number of depleted sensor nodes in the queue. Most of the studies on designing on-demand charging path for MC assumed that the energy of MC is extremely large or infinite, so in each charge, the MC will fully charges the sensors. This assumption reveals limitations when either the distances between the sensors or the energy consumption rate of sensors are large (as shown in Section VI).

Unfortunately, none of the existing works jointly consider both the charging path and the charging time.

III. Network model and problem formulation

A. Network model

We consider a WRSN deployed over a two-dimensional area of interest. The network consists of three components. The first component is a set of wireless sensors each of which is equipped with a wireless energy receiver. The second one is a mobile charger, which is a robot carrying a wireless charger. The last component is a base station responsible for gathering data from the sensors. The sensors periodically send packets containing the sensed data and information about the residual energy to the base station. From the information obtained, the base station can estimate the average energy consumption of the sensors using some methods such as [8]. The MC moves to the charging locations and charges the sensors periodically. After each charging round, the MC returns to the depot and recharges itself. In this paper, we assume that the MC always fully recharges at the depot and charges to the sensors as much as possible in each charging round. Our objective is to minimize the number of failure nodes (i.e. the nodes that depleted energy) after each charging round.

B. Problem formulation

Suppose the MC has finished charging $k-1$ ($k \geq 1$) rounds and comes back to the depot. Now, we are going to determine the charging schedule at round k. A charging schedule is

TABLE I
LIST OF NOTATIONS.

S_0	The depot
$\{S_1, S_2, ..., S_n\}$	The set of all sensors
$d(A, B)$	Euclidean distance between two locations or objects A and B
E_{max}	Battery capacity of sensors
E_{min}	The minimum energy that the sensor need to operate functionally
E_{MC}	Battery capacity of the MC
P_M	Per-second energy consumption of the MC when traveling
v	Velocity of the MC
$\{D_1, ..., D_m\}$	charging locations at round k
a_u	Time for the MC travels from D_{u-1} to D_u
τ_u	Charging time at D_u
p_j	Energy consumption rate of S_j at round k
$E_{remain,j}$	The residual energy of sensor S_j after finishing the charging round $k-1$
$E_{depot,j}$	The residual energy of sensor S_j after finishing the charging round k
$E_{i,j}$	The residual energy of s_j when the MC arrives at charging location D_i

composed of a charging path (which is a sequence of charging locations), and the charging time at each charging location. Obviously, the energy of a sensor during a charging round may attain the minimum value either when the MC arrives at a charging location and starts to charge, or when the MC finishes the charging round. Therefore, to determine whether a node dies during charging round k, we only need to consider its residual energy at the two timings mentioned above.

We suppose $D_0 \rightarrow D_1 \rightarrow D_2 \rightarrow ... \rightarrow D_m \rightarrow D_{m+1}$ as the charging path at round k, where $D_1, ..., D_m$ are the charging locations and $D_0 \equiv D_{m+1}$ represent the depot. According to [12], the per-second energy that the MC staying at a charging location D_i charges to a sensor S_j, denoted as $P_{i,j}$, is determined by the Friis formula as follows.

$$P_{i,j} = \frac{G_s G_r \mu}{L_p}\left(\frac{\lambda}{4\pi(d(S_j, D_i) + \beta)}\right)^2$$

which can be simplified as

$$P_{i,j} = \frac{\alpha}{(d(S_j, D_i) + \beta)^2} \tag{1}$$

where, α and β are constants indicated by the hardware of the charger and the received devices. Let p_j be the average energy consumption rate of sensor S_j after charging round $(k-1)$ (i.e., the charging consumption rates of the sensors are estimated by the base station); $E_{remain,j}$ and $E_{depot,j}$ be the residual energy of sensor S_j after finishing the charging round $k-1$ and k; τ_u be the charging time at D_u. We denote $E_{i,j}$ is the residual energy of s_j when the MC arrives at charging location D_i. Then, $E_{i,j}$ and $E_{depot,j}$ can be determined as follows:

$$E_{i,j} = E_{remain,j} + \sum_{u=1}^{i-1} P_{u,j}\tau_u - \sum_{u=1}^{i} a_u p_j - \sum_{u=1}^{i-1} \tau_u p_j$$

$$E_{depot,j} = E_{depot,j} + \sum_{u=1}^{m} P_{u,j}\tau_u - \sum_{u=1}^{m+1} a_u p_j - \sum_{u=1}^{m} \tau_u p_j$$

where $a_u = \dfrac{d(D_{u-1}, D_u)}{v}$ is the time for the MC travels from D_{u-1} to D_u. We define a binary variable f_j indicating whether S_j dies at round k as follows.

$$f_j = \begin{cases} 1 \text{ if } \exists\ i \in \{1, .., m\} \text{ satisfying that } & E_{i,j} < E_{min} \text{ or } \\ & E_{depot,j} < E_{min}, \\ 0 \text{ otherwise} \end{cases}$$

This conditions can be represented as follows.

$$f_j = \max\{x_{1,j}, x_{2,j}, ..., x_{m,j}, y_{depot,j}\} \tag{2}$$

where $x_{1,j}$,...,$x_{m,j}$ and $y_{depot,j}$ are binary variables defined by

$$\text{If } E_{i,j} < E_{min} \text{ then } x_{i,j} = 1, \text{ else } x_{i,j} = 0,\ i = \overline{1, m} \tag{3}$$

$$\text{If } E_{depot,j} < E_{min} \text{ then } y_{depot,j} = 1, \text{ else } y_{depot,j} = 0 \tag{4}$$

Consequently, our problem can be mathematically formulated as follows.

Objective

$$\text{Minimize} \quad F = \sum_{i=1}^{n} f_j \tag{5}$$

Subject to

$$\sum_{j=1}^{n}\sum_{i=1}^{m} P_{i,j}\tau_i + \sum_{u=1}^{m} a_i P_M = E_{MC} \tag{6}$$

$$\begin{aligned} & E_{min} - E_{remain,j} \\ & \quad + \Big(\sum_{u=1}^{i} a_u + \sum_{u=1}^{i-1} P_{u,j}\tau_u\Big)p_j \geq M(x_{i,j} - 1) \\ & \qquad \forall i = \overline{1, m}\ , \forall j = \overline{1, n} \end{aligned} \tag{7}$$

$$\begin{aligned} & E_{min} - E_{remain,j} + \Big(\sum_{u=1}^{i} a_u + \sum_{u=1}^{i-1} P_{u,j}\tau_u\Big)p_j \leq M x_{i,j} \\ & \qquad \forall i = \overline{1, m}\ , \forall j = \overline{1, n} \end{aligned} \tag{8}$$

$$\begin{aligned} & E_{min} - E_{depot,j} - \sum_{u=1}^{m} P_{u,j}\tau_u \\ & \quad + \Big(\sum_{u=1}^{m+1} a_u + \sum_{u=1}^{m} \tau_u\Big) \times p_j \geq M(y_{depot,j} - 1) \\ & \qquad \forall j = \overline{1, n} \end{aligned} \tag{9}$$

$$\begin{aligned} & E_{min} - E_{depot,j} - \sum_{u=1}^{m} P_{u,j}\tau_u \\ & \quad + \Big(\sum_{u=1}^{m+1} a_u + \sum_{u=1}^{m} \tau_{u,j}\Big)p_j \leq M y_{depot,j} \\ & \qquad \forall j = \overline{1, n} \end{aligned} \tag{10}$$

$$P_{i,j}\tau_i \leq E_{max} - E_{depot,j} + (\sum_{u=1}^{i} a_u - \sum_{u=1}^{i-1} \tau_u)p_i \quad (11)$$

$$f_j \geq x_{i,j}, \forall i = \overline{1,m}, \forall j = \overline{1,n} \quad (12)$$

$$f_j \geq y_{depot,j}, \forall j = \overline{1,n} \quad (13)$$

$$f_j \leq x_{1,j} + ... + x_{m,j} + y_{depot,j}, \forall j = \overline{1,n} \quad (14)$$

$$x_{1,j}, x_{2,j}, ..., x_{m,j}, y_{depot,j} \in \{0,1\}, \forall j = \overline{1,n} \quad (15)$$

$$f_j \in \{0,1\}, \forall j = \overline{1,n} \quad (16)$$

where M is a significantly large constant. Constraint (6) depicts that the total energy that the MC consumes for traveling and received charging energy of all sensors in one round does not exceed its battery capacity. Constraints (7) and (8) represent condition (3), while constraints (9) and (10) depict condition (4). Constraint (11) represents that the total energy of a sensor after charging cannot exceed its battery capacity. Constraints (12)-(14) describe condition (2). We summarize all the notations in Table I.

As the charging locations are undetermined, the formulation described above cannot be solved using LP-solvers. To this end, we propose an approximation algorithm that consists of two phases. The first phase is to identify the optimal charging locations, and the second phase is to determine the charging time the MC spends at each charging location.

IV. Greedy charging location determination algorithm

A. Basic ideas

Our algorithm is designed based on the following observations: Firstly, to increase the lifetime of the sensors, every sensor should be charged as much as possible. Moreover, as the battery capacity of the MC is limited, we should minimize the traveling time of the MC to save the energy consumed for traveling, thereby increasing the energy that the MC charges to the sensors. This can be done by minimizing the number of charging locations and the distance between the charging locations.

Secondly, to avoid node failure, sensors with low residual energy and high energy consumption rate should be charged more. Accordingly, charging locations should be located close to these sensors. Specifically, let $\delta(D_i, S_j)$ denote the energy gain of sensor S_j when the MC stays at charging location D_i, which is defined by the difference of the per-second energy that S_j is charged to the energy that S_j consumes, then to prevent S_j from running out of energy, there must exist at least one charging location at which the energy gain of S_j is positive.

Based on the above observations, our idea to determine the charging locations is as follows. For each sensor, we first determine a location namely *preferred charging region* which is a disc centered at the sensor, and whose radius is the maximum distance at which the energy gain of the sensor is positive. Then, we try to locate the minimal number of charging locations such that each sensor has at least one charging location staying inside its preferred charging region.

Algorithm 1: Greedy charging location determination.

Input:
Network area $A(W \times H)$
$\{S_1, S_2, ..., S_n\}$: List of sensors
$p = \{p_1, p_2, ..., p_n\}$: Set of average energy consumption rate of sensors
Charging parameters α, β

Output: Set of charging locations $D \subset A$

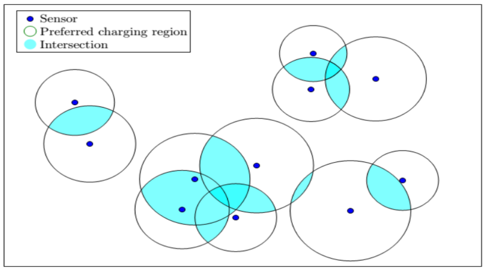
$\mathcal{S} \leftarrow S$
$R \leftarrow \varnothing$, $\mathcal{I} \leftarrow \varnothing$
$D \leftarrow \varnothing$
for $j = 1$ **to** n **do**
  $R_j \leftarrow \max\{0, \sqrt{\frac{\alpha}{p_j}} - \beta\}$
  Construct disk C_j at S_j with radius R_j
end for
while $\mathcal{S} \neq \varnothing$ **do**
  $\mathcal{I} = \cap C_j$
  Chose the region $e \subset \mathcal{I}$ belongs to the most circle
  Select a location $P \in e$ to minimize the mean distance to its covered sensors
  $D \leftarrow D \cup \{P\}$
  Remove C_j that contains at least one charging location from domain A
  Remove sensors have at least one charging location nearby from $\mathcal{S}$
end while
return D

B. Greedy Charging Location determination algorithm

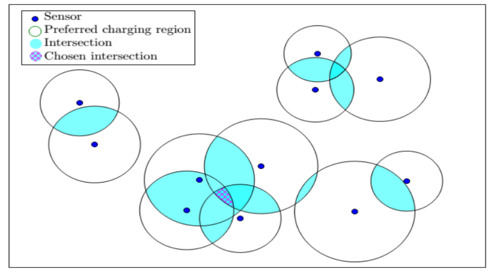
Our algorithm consists of two steps is described as follows.

1) *Step 1: Determining the preferred charging regions of the sensors.* For each sensor S_j, the per-second energy that S_j will be charged when the MC stays at D_i is defined by $P_{i,j} = \frac{\alpha}{(d(D_i,S_j)+\beta)^2}$. Therefore, the energy gain of sensor S_j is positive if and only if $d(D_i, S_j) \leq \sqrt{\frac{\alpha}{p_j}} - \beta$, where p_j is the energy consumption per second of S_j. Therefore, we define the preferred charging region of S_j as the disc with the radius of $\max\{0, \sqrt{\frac{\alpha}{p_j}} - \beta\}$ centered at S_j.
2) *Step 2: Choosing the optimal charging location.* We denote $\mathcal{S}$ as the set of all sensors whose preferred charging regions have not covered any charging location determined so far. We determine the intersections of the preferred charging regions of the sensors belonging to $\mathcal{S}$, and denote the set of these intersections as $\mathcal{I}$. Then, we determine intersection region $e \in \mathcal{I}$ which belongs to the most charging regions of sensors in $\mathcal{S}$. Finally, a new charging location is placed inside e so that its distance to the covered sensors (i.e., the sensors whose preferred charging regions cover e) is minimum.
Step 2 is repeated until the preferred charging region of every sensor covers at least one charging location (i.e., $\mathcal{S} = \varnothing$).

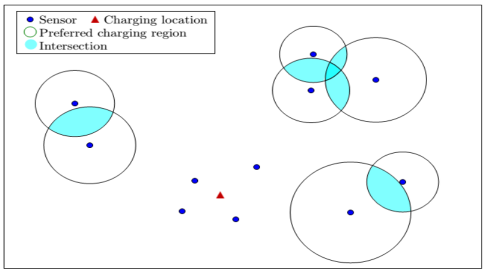
Algorithm 1 shows the pseudo-code of our greedy algorithm. Fig. 1 gives an illustrated example of our algorithm.



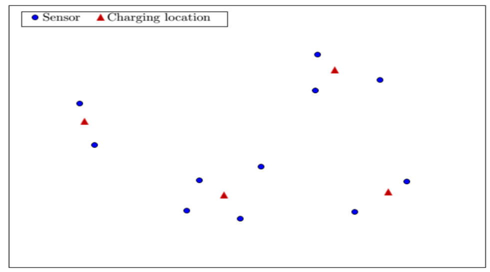
(a) Determining the preferred charging locations of the sensors.



(b) Choosing the intersection region belonging to the most preferred charging locations.



(c) Determining the charging location inside the chosen region.



(d) Repeating step 2 until the preferred charging location of each sensor covers at least one charging location.

Fig. 1. Illustration of charging location determination algorithm.

V. GA-based Charging Time Optimization Algorithm

Let $\{D_1, ..., D_m\}$ be the charging locations determined by our algorithm presented in Section IV. In this section, we propose a meta-heuristic algorithm to determine the charging schedule of the MC. The charging schedule is comprised of the travel path through the charging locations, and the charging time that the MC spends at each charging location.

A. Individual encoding and fitness function

Since we need to optimize both the travel path and the charging time, we propose a method to encode both these two elements into one individual. Each individual is encoded by a $2 \times n$ matrix whose first row represents the order of the charging locations visited by the MC, and the second row represents the charging time at each corresponding charging location. Specifically, an individual I is encoded as follows.

$$I = \begin{pmatrix} \pi_1 & \pi_2 & ... & \pi_m \\ \tau_1 & \tau_2 & ... & \tau_m \end{pmatrix}$$

where $\{\pi_1, \pi_2, ..., \pi_m\}$ is a permutation of $\{1, 2, ..., m\}$ and $\tau_1, \tau_2, ..., \tau_m$ are the charging time at $D_{\pi_1}, D_{\pi_2}, ..., D_{\pi_m}$, respectively. Fig. 2 illustrates our individual encoding method. As the MC has limited battery capacity, each individual has to satisfy the following energy constraints from (6) to (11). The total energy consumption of the MC in a charging round is comprised of two components. The first one, denoted as E_T, is the energy consumed for traveling through all the charging locations. The second one, denoted as E_C, is the energy that the MC charges all sensors at all charging locations. The sum of E_T and E_C must not exceed the battery capacity of the MC. This constraint can be mathematically represented as follows.

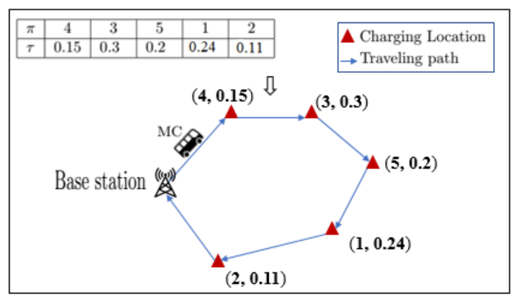


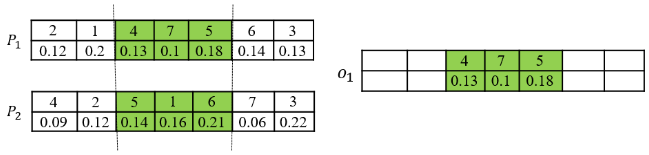
Fig. 2. Individual encoding example.

$$\begin{aligned} E_T &= \sum_{i=0}^{m} \frac{d(D_{\pi_i}, D_{\pi_{i+1}})}{v} P_M \\ E_C &= \sum_{i=1}^{m} \sum_{j=1}^{n} P_{ij} \tau_i \\ \Rightarrow \sum_{i=1}^{m} \sum_{j=1}^{n} P_{ij} &\leq E_{MC} - \sum_{i=0}^{m} \frac{d(D_{\pi_i}, D_{\pi_{i+1}})}{v} P_M \end{aligned} \quad (17)$$

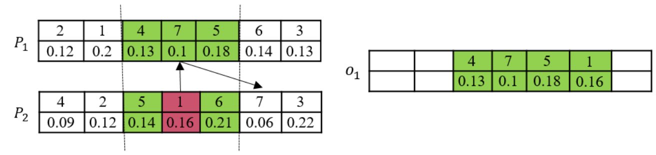
Let $w = \min_{i \in [1,m]} \{\sum_{j=1}^{n} P_{ij}\}$, we have:

$$\sum_{i=1}^{m} \tau_i \leq \frac{E_{MC} - \sum_{i=0}^{m} \frac{d(D_{\pi_i}, D_{\pi_{i+1}})}{v} P_M}{w} \quad (18)$$

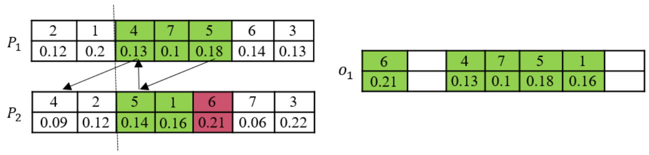
The fitness value of an individual I is the total number of dead nodes when the MC follows the charging schedule defined by



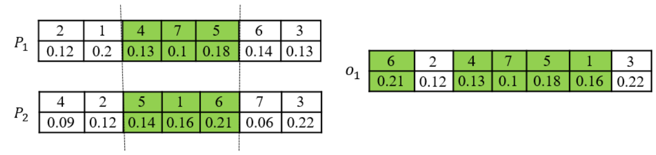
(a) O_1 inherits the segment between two crossover points of P_1 (the green segment).



(b) Skips "5" as it already appears in O_1. Inserts "1" into the sixth position of O_1 which corresponds to the position of "7" in P_2.



(c) Inserts "6" into O_1. As the fifth position in O_1 is not available, "6" is inserted into the first position.



(d) Fills the remaining genes of P_2 into O_1 according to their positions in P_2.

Fig. 3. An example of EPMX crossover operator.

I:

$$F = \sum_{i=1}^{n} f_i \tag{19}$$

where f_i ($i = \overline{1, n}$) are binary parameters satisfying constraints described in Section III-B.

B. Population initialization

The travel paths (i.e., the first row) of the individuals in the initial population are generated randomly, where each travel path is a permutation of $\{1, 2, ..., m\}$. The charging time (i.e., the second row) of the individuals is initialized as follows. Let $\tau_i^* = \frac{\tau_i}{\frac{E_{MC}-E_T}{w}}$, then constraint (18) is equivalent to:

$$\sum_{i=1}^{m} \tau_i^* \leq 1$$

We first randomly generate m numbers $\rho_1, \rho_2, ..., \rho_m$ following the uniform distribution in interval $[0, 1]$. After that, we sort these m generated numbers in the increasing order. Without loss of generality, assuming that $0 \leq \rho_1 \leq \rho_2 \leq ... \leq \rho_m \leq 1$. Let $\tau_1^* = \rho_1$, $\tau_2^* = \rho_2 - \rho_1$,..., $\tau_m^* = \rho_m - \rho_{m-1}$, then it is obvious that $\sum_{i=1}^{m} \tau_i^* = \rho_m \leq 1$, hence satisfying the constraint constraint (18). From τ_i^*, we can calculate τ_i by multiplying with $\frac{E_{MC}-E_T}{w}$.

C. Crossover operation

Let P_1 and P_2 be two parents, we generate two offsprings O_1 and O_2 by using our proposed crossover and mutation operators described in the following.

Our crossover algorithm combines two operators: Enhanced Partially Mapped crossover (EPMX) and Enhanced Simulated Binary crossover (ESBX). Specifically, EPMX and ESBX are our proposed algorithms which are based on the Partially Mappep Crossover (PMX) [15], and the Simulated Binary crossover(SBX) [16], respectively . To ease the presentation, hereafter, we present individual I as $\begin{pmatrix} \pi_1^I & \pi_2^I & ... & \pi_m^I \\ \tau_1^I & \tau_2^I & ... & \tau_m^I \end{pmatrix}$.

Originally, PMX is used only for crossing over two individuals which are just one-dimension arrays. Therefore, PMX cannot be applied directly to our problems whose individuals are encoded by two-dimension arrays. To this end, we propose EPMX which customizes PMX as follows. We choose two random positions $i, j \in \{1, 2, ..., m\}$, $i < j$. Then, the first offspring (i.e., O_1) inherits all the genes from position i to j of P_1. It means that $\pi_u^{O_1} = \pi_u^{P_1}$ and $\tau_u^{O_1} = \tau_u^{P_1}$ for all $u = i, i+1, ..., j$. The other genes of O_1 are determined as follows.

- Firstly, we check all the genes at positions from i to j of parent P_2. At each position $k \in [i, j]$, if $\pi_k^{P_2}$ has not appear in O_1, then we find a position $x*$ in O_1 to assign the value of the k-th gene of P_2 as follows. We find x_1 in P_2 such that $\pi_{x_1}^{P_2} = \pi_k^{P_1}$. If $x_1 \in [i, j]$, we continue to find x_2 such that: $\pi_{x_2}^{P_2} = \pi_{x_1}^{P_1}$. Repeating this process until finding x_* such that $x_* \notin [i, j]$. Then, $\pi_{x_*}^{O_1} = \pi_k^{P_2}$ and $\tau_{x_*}^{O_1} = \tau_k^{P_2}$.
- Finally, we assign the remaining genes of P_2 (i.e., in both two rows) to the corresponding positions in O_1.

The second offspring, i.e., O_2 is generated similarly by swapping roles of P_1 and P_2. Fig. 3 illustrates the EPMX crossover operator.

Our proposed ESBX is based on the Simulated Binary crossover (SBX) algorithm. We let O_1 and O_2 inherit exactly same the first rows as P_1 and P_2, respectively. Note that, by doing this, the traveling paths of O_1 and O_2 are the same as that of P_1 and P_2. Accordingly, the energy consumption for traveling in O_1 and O_2 are identical to that in P_1 and P_2, respectively. Therefore, to guarantee the constraint (18), the charging time of O_1 and O_2 should be made to not too different from that of P_1 and P_2, respectively. Consequently, we generate the charging time for O_1 and O_2 as follows.

$$\begin{cases} \tau_k^{O_1} = \dfrac{\tau_k^{P_1} + \tau_k^{P_2}}{2} - \frac{1}{2}\beta(\tau_k^{P_2} - \tau_k^{P_1}) \\ \tau_k^{O_2} = \dfrac{\tau_k^{P_1} + \tau_k^{P_2}}{2} + \frac{1}{2}\beta(\tau_k^{P_2} - \tau_k^{P_1}) \end{cases} \forall k \in \{1, 2, ..., m\}$$

where β is a random number whose probability density function is defined as:

$$f(\beta) = \begin{cases} 0.5(h+1)\beta^h \text{ if } \beta < 1 \\ 0.5(h+1)\frac{1}{\beta^{h+2}} \text{ if } \beta \geq 1 \end{cases}$$

h is a positive integer.

D. Mutation operation

We leverage the Swap mutation and the Polynomial mutation to mutate the individuals. Let I be an individual being mutated, then our mutation process are conducted as follows. In the Swap mutation, we first copy all the genes of the parent I to the offspring O. Then, we choose two random points $i, j \in \{1, 2, ..., m\}$, and swap the i-th and the j-th genes of O, i.e., swap (π_i, τ_i) with (π_j, τ_j).

Concerning the Polynomial mutation operator, we customize the original algorithm proposed by [17]. Specifically, we first copy the first row (i.e., the travel path of the MC) of the parent I to the offspring I. Then, the original Polynomial mutation operator is applied to the second row of I to achieve the charging time for offspring O.

E. Individual repair

Note that the individual generated by the crossover and mutation operations may not satisfy constraint (18). Therefore, we need a repair algorithm to adjust the generated individuals. Let I be an individual that need to be repaired, and let $\sum_{i=1}^{m}\sum_{j=1}^{n} P_{i,j}\tau_i = E_C$ be the total energy that the MC will charge to all sensors according to schedule defined by I. Then, we adjust the second row of I by multiplying all of its items with $\dfrac{E_{MC} - E_T}{E_C}$.

VI. Performance Evaluation

To simplify the presentation, we name our proposed algorithm as GR-GACS (**GR**reedy **G**enetic **A**lgorithm - based **C**harging **S**chedule). We compare our proposal with most relevant work (named as INMA [8]), whose objective is avoid node failure. In INMA, when the energy of a sensor is below a predefined threshold, it will send a charging request to the MC. The MC chose the next charged nodes in the MC's travel path base on the residual energy of the sensors and the distance from sensors to the MC. Specifically, the next charged sensors are selected to minimize the number of other requesting nodes that may suffer from energy depletion.

A. Simulation settings

All experiments are implemented in JMetalPy framework and conducted on a computer with an Intel Core i7-6700HQ CPU and 8GB of RAM. For the MC, we use parameters which proposed by [5], [4] as follow: $E_{max} = 10800J$, $E_{min} = 540J$, $U = 5J/s$, $E_{MC} = 108000J$, $P_M = 1J/s$ and $V = 5m/s$. The network size is $1000 \times 1000\ m^2$. We placed the sensors in the network area by the following two distribution. In the first distribution, namely grid distribution, the sensors are randomly scattered on 50 $\times$ 50 square grids. In the second distribution, namely normal distribution, the sensors are placed randomly over the network. As mentioned in Section III, the base station collect the information related to the energy of the sensors and estimated the average energy consumption of each node by using some methods such as [8].

We conducted experiments to evaluate the impact of the important factors on the number of dead nodes caused by the algorithms. However, due to the space limit, we present the results concerning only two factors having the greatest influence, which are the number of sensors and the average energy consumption rate of the sensors. For each factor, we calculate the node failure ratio, which is the number of dead nodes over the total number of nodes [8]. Each value plotted on the curves represents the average gathered by 30 runs along with 95% confidence intervals. For determining the charging locations (i.e. Greedy algorithm in Section IV), we set $\alpha = 3600$, $\beta = 30$. For determining the charging time (i.e. GA algorithm in Section V), we conducted experiments and chose the optimal parameters manually. We set the crossover ratio r_c to 0.9, mutation ratio r_m to $\frac{1}{m}$ (with m is the number of charging locations), EPMX crossover has the same chosen probability rate as ESBX crossover. We chose $h = 2$ in probability density function. The number of population is set to 100, the maximum number of generations is 500, and we also terminate the algorithm if the fitness value does not improve after 200 generations. We simulated the algorithms in $32000s$ and measured the total number of dead nodes.

B. Impact of the number of sensors

In this experiment, we vary the number of sensors from 25 to 200. The experiment result is shown in Fig.4, where the x-axis represents the number of sensor nodes, the y-axis represents the ratio of the total dead nodes to the total number of sensors. As shown, both algorithm's slopes have an increasing trend, with GR-GACS outperforms INMA in all cases. For example, at networks with 200 sensors, GR-GACS achieves the average node failure ratios of 57.63% and 38.95% concerning grid and normal distribution, while that of INMA is 72.68% and 59.01%, respectively.

However, as the total number of sensors increase, the performance gap between the algorithms becomes smaller. Specifically, when the number of sensors is 25, the performance gaps are 28.89% and 25.59% concerning the grid and normal distributions, respectively; and when the number of sensors is 200 nodes, the performance gaps are 15.05% and 20.06%, respectively. This can be explained as follows. INMA follows the online charging approach which only charges to sensors that sent the request to MC. On the other hand, GR-GACS follows the periodic charging approach which tries to charge all sensors at every charging round. Consequently, when the number of sensors increases, the energy of the MC becomes insufficient to charge all sensors.

Additionally, as shown in Fig.4, GR-GACS performs significantly better in the normal distribution than the grid distribution. It is because the locations of sensors deployed by normal distribution are concentrated, thus, the number of charging locations is reduced significantly. As a result, MC spends less energy on traveling, and more energy for charging sensors. Concerning the grid distribution, as the locations of sensors are spread over the network, more charging locations are required. This results in high traveling energy consumption of MC and thus reduces the energy the sensors are charged.

C. Impact of the average consumption rate of sensors

In this experiment, the number of sensors is set to 100 and their location is decided by the grid distribution. We

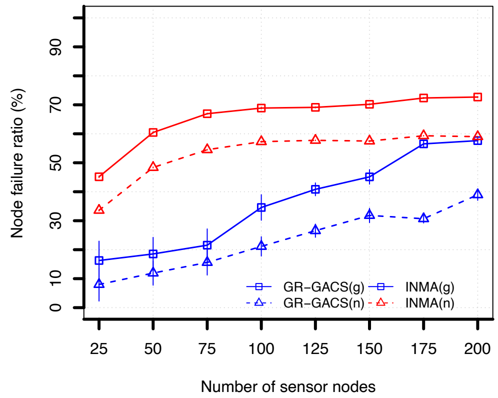


Fig. 4. Impact of number of sensors with respect to two distributions: normal(n) and grid(g).

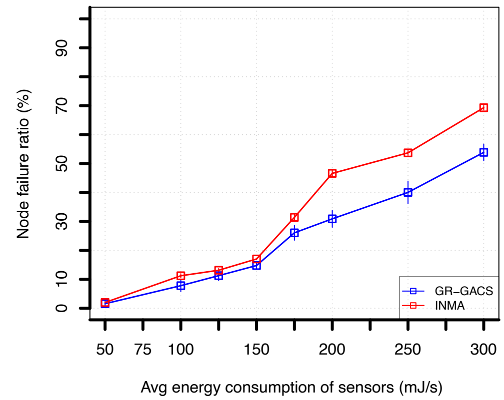


Fig. 5. Impact of the average energy consumption rate of sensors.

vary the average consumption rate of the sensors in the range from $50mJ$ to $300mJ$. It can be observed that the curves of both two algorithms are linear growth. This is straightforward because higher energy consumption means a lower lifetime for each sensor node. As shown, our GR-GACS outperforms INMA, especially with the average energy consumption rate of more than $200mJ$. When the energy consumption rate is small (i.e., from $50mJ$ to $150mJ$), the performance gap between the algorithms is significant which is about 3.43%. The increase in the consumption rate causes an increase in the performance gap. Specifically, when the average consumption is $175mJ$, the gap is only 5.33%, but when the average consumption is $200mJ$, the gap increases to 15.7%.

In summary, our algorithm outperforms INMA concerning all the experiment scenarios.

VII. Conclusion

In this paper, we jointly studied how to optimize the charging path and charging time to minimize the number of dead nodes over a charging round. We provided a mathematical formulation of the concerned problem. We then proposed an approximation which is comprised of two phases: charging path identification and charging time determination. We extensively conducted experiments in different scenarios and compared with the state-of-the-art. The experimental results showed that our proposed algorithm reduces significantly the number of dead nodes compared to the benchmark.

Acknowledgment

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2019.304.

References

[1] P. Le Nguyen, Y. Ji, K. Le, and T. Nguyen. Load balanced and constant stretch routing in the vicinity of holes in wsns. In *2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pages 1–6, 2018.

[2] D. Kandris, C. Nakas, D. Vomvas, and G. Koulouras. Applications of wireless sensor networks: An up-to-date survey. *Applied System innovation*, 14, 2020.

[3] Phi Le Nguyen, Kien Nguyen, Huy Vu, and Yusheng Ji. Telpac: A time and energy efficient protocol for locating and patching coverage holes in wsns. *Journal of Network and Computer Applications*, 147:102439, 2019.

[4] Zengwei Lyu, Zhenchun Wei, Jie Pan, Hua Chen, Chengkai Xia, Jianghong Han, and Lei Shi. Periodic charging planning for a mobile wce in wireless rechargeable sensor networks based on hybrid pso and ga algorithm. *Applied Soft Computing*, 75:388–403, 2019.

[5] Yi Shi, Liguang Xie, Y Thomas Hou, and Hanif D Sherali. On renewable sensor networks with wireless energy transfer. In *2011 Proceedings IEEE INFOCOM*, pages 1350–1358. IEEE, 2011.

[6] Liguang Xie, Yi Shi, Y Thomas Hou, Wenjing Lou, Hanif D Sherali, and Scott F Midkiff. Multi-node wireless energy charging in sensor networks. *IEEE/ACM Transactions on Networking*, 23(2):437–450, 2014.

[7] Yong Feng, Nianbo Liu, Feng Wang, Qian Qian, and Xiuqi Li. Starvation avoidance mobile energy replenishment for wireless rechargeable sensor networks. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2016.

[8] Jinqi Zhu, Yong Feng, Ming Liu, Guihai Chen, and Yongxin Huang. Adaptive online mobile charging for node failure avoidance in wireless rechargeable sensor networks. *Computer Communications*, 126:28–37, 2018.

[9] Wenzheng Xu, Weifa Liang, Xiaohua Jia, and Zichuan Xu. Maximizing sensor lifetime in a rechargeable sensor network via partial energy charging on sensors. In *2016 13th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 1–9. IEEE, 2016.

[10] Tran Thi Huong, Phi Le Nguyen, Huynh Thi Thanh Binh, Kien Nguyen, Ngo Minh Hai, and Le Trong Vinh. Genetic algorithm based - periodic charging scheme for energy depletion avoidance in wrsns. In *Wireless Communication and Network Conference (WCNC)*, 2020 (accepted).

[11] Najeeb Najeeb and Carrick Detweiler. Extending wireless rechargeable sensor network life without full knowledge. *Sensors*, 17(7):1642, 2017.

[12] Xuan Yang, Guangjie Han, Li Liu, Aihua Qian, and Wenbo Zhang. Igrc: An improved grid-based joint routing and charging algorithm for wireless rechargeable sensor networks. *Future Generation Computer Systems*, 92:837–845, 2019.

[13] Amar Kaswan, Abhinav Tomar, and Prasanta K Jana. An efficient scheduling scheme for mobile charger in on-demand wireless rechargeable sensor networks. *Journal of Network and Computer Applications*, 114:123–134, 2018.

[14] Yang Peng, Zi Li, Wensheng Zhang, and Daji Qiao. Prolonging sensor network lifetime through wireless charging. In *2010 31st IEEE Real-Time Systems Symposium*, pages 129–139. IEEE, 2010.

[15] David E Goldberg, Robert Lingle, et al. Alleles, loci, and the traveling salesman problem. In *Proceedings of an international conference on genetic algorithms and their applications*, volume 154, pages 154–159. Lawrence Erlbaum, Hillsdale, NJ, 1985.

[16] Kalyanmoy Deb, Ram Bhushan Agrawal, et al. Simulated binary crossover for continuous search space. *Complex systems*, 9(2):115–148, 1995.

[17] Kalyanmoy Deb and Mayank Goyal. A combined genetic adaptive search (geneas) for engineering design. *Computer Science and informatics*, 26:30–45, 1996.