Future Generation Computer Systems 140 (2023) 53–66

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

An efficient data collection algorithm for partitioned wireless sensor networks

Gongshun Min, Liang Liu *, Wenbin Zhai, Zijie Wang, Wanying Lu

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
Key Laboratory of Civil Aviation Smart Airport Theory and System, Civil Aviation University of China, Tianjing, China

ARTICLE INFO

Article history:
Received 13 March 2022
Received in revised form 30 August 2022
Accepted 3 September 2022
Available online 9 September 2022

Keywords:
Convex hull
Data collection
GTSP(Generalized Traveling Salesman Problem)
ISP tree(Improved Shortest Path tree)
MA (Mobile Agent)
Partitioned WSNs(partitioned Wireless Sensor Networks)

ABSTRACT

Data collection with mobile agent (MA) can balance the energy consumption of nodes in partitioned wireless sensor networks. However, the existing data collection algorithms for partitioned WSNs do not formalize the problem systematically. These algorithms are not efficient and difficult to meet the timeliness requirement. In order to solve the above shortcomings, we first formalize the Data Collection Problem (DCP). Then, it is transformed into a Generalized Traveling Salesman Problem (GTSP), and we propose a GTSP-Based data collection Algorithm (GBA) to calculate the rendezvous points (RPs) and the moving path of MA. GBA selects RPs by solving the GTSP problem. Furthermore, based on GBA, we take advantage of constructing convex hull to plan a more efficient moving path for MA. In addition, in order to reduce the energy burden on the RPs, we design an Improved Shortest Path tree (ISP tree) to aggregate data from nodes to RPs in partitioned WSNs. Finally, extensive simulations demonstrate the effectiveness and advantages of our algorithm in terms of the length of MA's moving path and the amount of data collection per unit time.

1. Introduction

Wireless Sensor Networks (WSNs) are the core component of the Internet of Things (IoT) [1–3]. They consist of multiple sensor nodes which are deployed in some areas to perform monitoring tasks and collect data. In recent years, WSNs have been used for air quality monitoring, water quality monitoring, smart transportation, smart agriculture, smart living [4–7], etc.

The core function of WSNs is to collect data from the monitored area. Many effective data collection algorithms have been proposed in recent years [8–11]. However, they are much concerned about connected WSNs, in which all nodes are directly or indirectly connected to each other. Due to the influence of the natural environment and topography, many large-scale WSNs are disconnected [12]. For example, the WSN shown in Fig. 1 is not fully connected, but is composed of multiple sub-WSN(Here after, we use s-WSN to denote sub-WSN), and all the nodes in each s-WSN are fully connected. In this paper, we call this kind of wireless sensor networks partitioned WSNs.

When collecting data from partitioned WSNs, a feasible manner is to deploy one static sink in each s-WSN and arrange a base station in the partitioned WSNs. Each node in s-WSN sends the sensing data to the sink in the s-WSN through multi-hop routing, then the sink sends the data to the base station for processing. This data collection method has two main disadvantages: (1) It introduces additional hardware overhead because of arranging a base station in for communication with the sink deployed in each s-WSN [13]. (2) The nodes closer to the sinks have to bear the data forwarding of the nodes farther away from the sinks while forwarding their own data, which results in unbalanced energy consumption of the nodes closer to the sinks. This phenomenon is called "energy hole" effect, which shortens the lifetime of the network [14,15].

To solve the above problems, many researchers introduce mobile agent(MA) to collect data in partitioned WSNs [16,17]. The nodes in each s-WSN send the sensing data to the rendezvous points(RPs) in the s-WSN. Then, MA visits the RPs in each s-WSN to collect data in the RPs. After visiting all RPs in all s-WSN, MA returns to the base station with data.

When using MA to collect data from partitioned WSNs, a key problem is how to collect data as fast as possible. Because the sensing data of nodes is usually time-sensitive [18,19], it is crucial to collect data from the networks timely. The time used by MA for a complete travel is related to the length of travel path. Therefore, how to select RPs and plan an efficient moving path for MA is a critical issue.

The data collection algorithms for connected WSNs cannot be used in partitioned WSNs directly. Moreover, the existing data collection algorithms for partitioned WSNs do not formalize the problem systematically. They usually select RPs firstly, and then use heuristic algorithms to find a moving path for MA to traverse

* Corresponding author.
E-mail address: liangliu@nuaa.edu.cn (L. Liu).

https://doi.org/10.1016/j.future.2022.09.006
0167-739X/

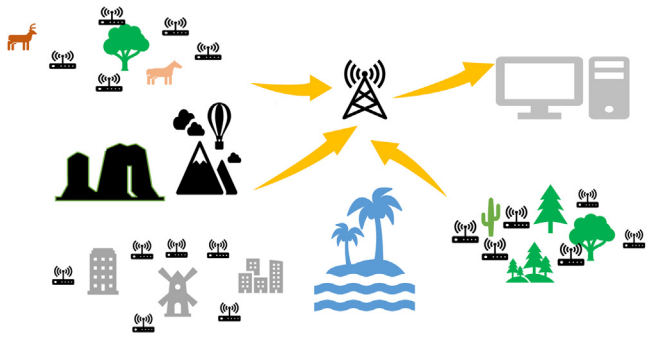

Fig. 1. Partitioned wireless sensor network.

the RPs. These algorithms are not efficient and difficult to meet the timeliness requirement.

In order to solve the above problems, we propose an Efficient Data collection Algorithm(EDA) for partitioned WSNs. First, we formalize the data collection problem as a generalized traveling salesman problem (GTSP [20,21]). Unlike the traditional algorithms which divide the selection of RPs and planning the moving path for MA into two steps, this paper combines the selection of RPs and obtaining the moving path of MA into one step by solving the GTSP problem.

The main contributions of this paper are outlined follows. (1) We formalize the Data Collection Problem(DCP) systematically. (2) DCP is transformed into a GTSP problem, and we propose a GTSP-Based data collection Algorithm(GBA). The RPs and moving path of MA are obtained simultaneously by solving the GTSP problem. (3) Based on GBA, we consider the effect of node's communication range on the moving path of MA and propose an Efficient Path Planning Algorithm(EPPA) to plan an more effective moving path for MA. (4) Extensive simulations demonstrate the effectiveness and superiority of our proposed algorithm in terms of the length of MA's moving path and the amount of data collected per unit time.

We summarize this paper as follows. Section 2 describes the work related to data collection in WSNs. The system model and the DCP problem are illustrated in Section 3. Section 4 describes the transformation and solution of the DCP problem. In Section 5, we conduct experiments and evaluate the proposed algorithm through experimental results. Finally, the paper is summarized in Section 6.

2. Related work

In recent years, many researchers have proposed many algorithms for data collection in WSNs. These works are broadly classified into three categories. (1) how to use the MA to collect data directly from each sensor. (2) how to schedule the MA to travel the RPs.

2.1. Using the MA to access each sensor

Several studies use m mobile ferries to collect data from sensors. The work in [22] divides the entire sensor network into m grids and assigns a ferry to visit the sensors in each grid. In addition, some contact points are chosen at the grid boundaries so that two ferries meet together. In [23], a path planning method is developed for the ferry. The method selects a contact point and cuts the network into m regions that intersect at that point. Each ferry finds a Hamiltonian cycle to visit all sensors and contact points in a region.

A hybrid WSN is considered in [24], which static sensors are responsible for detecting events and generating data, while mobile sensors moving to event locations to collect data. Since mobile sensors have limited energy, the goal of the paper is to extend their lifetime by scheduling them round by round. The event site is divided into clusters, and [24] assigns a mobile sensor to each cluster, which not only considers reducing the travel path of the mobile sensors, but also balances their energy consumption. The mobile sensor then uses the TSP method to access each node in the cluster.

Given a set of sparsely dispersed sensors, some studies arrange the travel paths of mobile agents to collect data from sensor nodes with the goal of minimizing the path length. The study in [25] assumes that the sensors have different communication ranges and once the mobile sink (MS) enters the sensor's communication range, it can obtain data from the sensors. Therefore, [25] uses an evolutionary algorithm to find the contact points on the boundary of each sensor communication range and connects these points through a TSP approach, thus reducing the overall path. The work of [26] is similar to [25] which also finds the contact points from the boundary of the communication range, but it eliminates some redundant points (e.g., visiting the same sensor multiple times). The travel path is further shortened by this approach.

The study in [27] assumes that the speed of the MS is tunable, and the goal is to arrange the travel path of the MS while varying its speed along the path so that the MS can collect data from each sensor in the shortest possible time. [28] used an incremental strategy to compute travel path with delay constraints. They first use the TSP method to find the paths that access all sensors. By modeling the communication range of the sensors as disks, the Welzl method [29] is then used to find the smallest closed disk on the path. In this way, the path is reduced because when the MS is located within the overlapping area of the communication range of multiple sensors, it can collect data from these sensors.

When there are many sensors, accessing each node makes the travel path longer. Therefore, the above schemes may not perform well in a large-scale WSN.

2.2. Scheduling the MA to travel the RPs

Both [30,31] assume that a MS keeps moving along a fixed pre-planning trajectory and sensors are scattered around the path of the MS. When a sensor lies within the communication range of the MS, it becomes a RP and other sensor nodes forward data to this RP via multiple-hop routing.

Several studies use a tree connecting all sensors to find the set of travel paths and RPs. [32] forms a minimal Steiner tree and traverses it in a pre-ordered manner until the MS moves to visit the RPs over the delay threshold. Since the Steiner tree may have virtual vertices (i.e., they do not correspond to sensors), [32] replaces RPs that may be virtual vertices with the closest sensors. However, this approach may result in longer data forwarding paths for some non-RP sensors, therefore, it will force sensors to spend more energy on communication. Xing et al. [33] create a routing tree in a WSN and assign a weight to each tree edge based on the number of sensors that use the tree edge to forward data. Then, the MS selects the edge with a larger weight under a delay constraint. Since the MS will move along the tree edges, some RPs may be visited multiple times. Therefore, [33] iteratively corrects the travel path by the TSP method. However, the TSP method is invoked $n*RP$ times in each iteration, so the scheme incurs a high computational overhead.

The work in [34] proposes a cluster-based (CB) approach to find RPs. It randomly selects some sensors as heads of clusters, and other sensors which are closest to the heads will join the clusters. Then, one RP is selected from each cluster. if the travel path used to access all RPs is shorter than the threshold L_{max}, the

Table 1
Table of related work.

No.	Algorithm	Hol-Opt
21	Ferry route design algorithms	No
22	Centralized ferry relaying	No
23	Centralized and a distributed heuristics	No
24	TSPN	No
25	Substitution heuristic algorithm	No
26	Heuristic algorithm	No
27	MR-CSS	No
28	Seidel's recent linear programming algorithm	No
29,30	Selection of RPs algorithm	No
31	Rendezvous-based data collection algorithm	No
32	RP-CP and RP-UG	No
33	Cluster-based algorithm	No
34	Weighted rendezvous planning algorithm	No

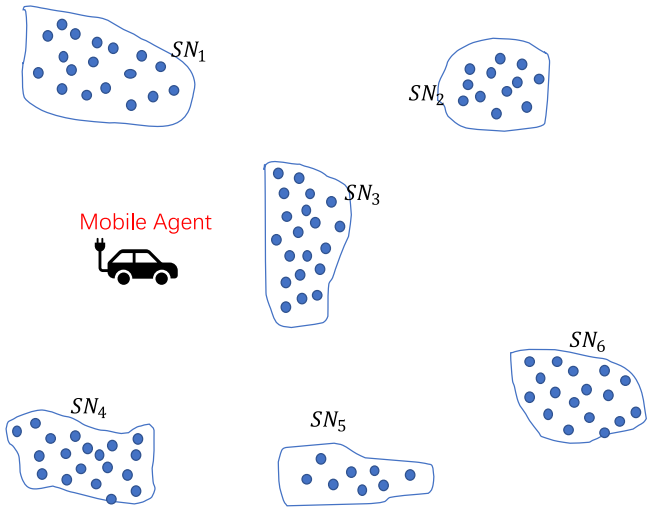

Fig. 2. Network model.

CB adds more cluster heads and repeats the procedure. However, since cluster heads are chosen randomly, some clusters may have more sensors, which makes their sensors spend more energy on communication. In [35], a weighted rendezvous planning (WRP) scheme is proposed by forming a spanning tree in a WSN. Each sensor s_i is assigned a weight $W_i = N_i * H(i, M)$,where N_i is the number of packets sent by s_i to the RP and $H(i, M)$ is the number of hops between s_i and its nearest RP. Since WRP assumes that each sensor generates one sensing packet, N_i is the number of children of s_i plus 1. Then, WRP selects an RP with the maximum weight and uses the TSP method to schedule the paths to visit each RP. The iterations are repeated until the path length exceeds L_{max}. However, since it uses the TSP method to recalculate the paths (including the newly discovered paths) to visit the RPs in each iteration, the time complexity of WRP is $O(n^2\gamma)$, which is higher than [33].

Finally, We have organized the papers involved in the related work into Table 1. In Table 1, we summarize the related algorithms in detail, as well as the limitation of them. The limitation of these algorithms is no holistic optimization. No holistic optimization means the selection of RPs and planning moving path for MA are two separate steps. We use Hol-Opt to represent holistic optimization in Table 1.

3. Network model and problem statement

3.1. Network model

Consider a set of sensor nodes $S = \{s_1, s_2, \ldots, s_n\}$ distributed in an area randomly, these nodes form a partitioned WSNs, namely PN, as shown in Fig. 2. PN consists of K sub-network $SN_i (1 \le i \le K)$. Each SN_i has several nodes. $\forall s_i, s_j \in SN_k$, they can communicate with each other directly or indirectly. While $\forall s_i \in SN_k, \forall s_j \in SN_h$, they cannot communicate with each other. Each sensor node generates a data packet of size l bits during a time interval T.

In a round of data collection, Each node in SN_i sends data to the RP in SN_i. MA visits the RP in SN_i one by one from the starting point, and collects data in the RP. The process of data collection ends after visiting all SN_i, the time used for this process is denoted as T_{tour}.

We adapt the following assumptions proposed in [35]. First, We use only one MA to collect data. Second, MA visits the RPs at a constant speed v. Third, the communication time of data gathering can be neglected compared with the traveling time of MA. Finally, in practical applications, the number of sub-networks should be less than a specific constant C_0, i.e., $K < C_0$ [12].

3.2. Energy consumption model

The typical energy consumption model for wireless transmission in [34] is adopted in our study. For two interconnected nodes s_i and s_j with Euclidean distance $d(s_i, s_j)$, when s_i sends y bits of data to s_j, the energy consumed by s_i sending the data is

$$E_T(s_i, s_j) = \gamma_1 y + \gamma_2 d(s_i, s_j)^{\epsilon} y \tag{1}$$

and the energy consumed by s_j receiving the data is

$$E_R(s_j, s_i) = \gamma_1 y \tag{2}$$

where γ_1 is the energy used by the transmitter electronics, γ_2 is the transmitting amplifier, and ϵ is the propagation loss index satisfying $2 < \epsilon < 4$. The values of amplifier γ_2 and ϵ are determined by the actual situation.

3.3. Problem statement

In this subsection, we formalize the data collection problem(DCP) in partitioned WSNs. Our work is to first find a set of sensor nodes SN_{rp} to become RPs. $SN_{rp} = (m_1, m_2, \ldots, m_k, \ldots, m_K)$, where $m_k \in SN_k$, $1 \le k \le K$. Each node in SN_k sends its data to m_k. Then find an efficient path P_{ma} for MA to visit these RPs. $P_{ma} = (n_{i_1} \to n_{i_2} \to \ldots \to n_{i_K} \to n_{i_1})$, where n_{i_k} is a location near the RP m_{i_k}, $1 \le i_k \le K$. MA moves to n_{i_k} to collect the data in m_{i_k}. The length of P_{ma} is denoted as $L_{P_{ma}}$, the time used for this process is denoted as $T_{P_{ma}}$. Our overall goal is to make the amount of data collected by MA per unit time as much as possible. DCP is formalized as follows:

$$\max \frac{\sum_{i=1}^{K} L_i}{T_{P_{ma}}} \tag{3}$$

$$L_i = |SN_i| l \tag{4}$$

$$T_{P_{ma}} = \frac{L_{P_{ma}}}{v} \tag{5}$$

s.t.

$$L_{P_{ma}} \le T v \tag{6}$$

$$d(m_{i_k}, n_{i_k}) \le r \tag{7}$$

where $|SN_i|$ represents the number of nodes in SN_i, L_i represents the total amount of data collected by MA in SN_i. $T_{P_{ma}}$ represents the time used by MA to complete one round of data collection. T in Eq. (6) is an upper bound on the data delivery delay, and v is the constant travel speed of MA. Eq. (7) guarantees that MA is within the communication range of the RPs to be traversed.

4. Design of the algorithm

4.1. Main challenges of DCP

The total time $T_{P_{ma}}$ for MA to complete a round of data collection task depends on the selection of RPs and the moving path of MA. The shorter the moving path of MA will cause the smaller $T_{P_{ma}}$ and the more data collected per unit time. Therefore, there are two challenges in solving the DCP problem: (1) how to select suitable RPs; (2) how to plan an efficient moving path for MA to traverse the RPs.

To address the above challenges, we convert the selection of RPs and the planning MA's moving path into a GTSP problem, and propose a GTSP-Based data collection Algorithm(GBA). The RPs and moving path of MA are obtained simultaneously by solving the GTSP problem. According to the moving path of MA obtained by solving GTSP, MA moves directly to the location of the RPs to collect data.

However, in fact, MA only needs to move within the communication range of a RP to collect data from the RP. Therefore, after obtaining the RPs by using GBA, we propose an efficient path planning algorithm(EPPA). EPPA uses the convex hull generation algorithm to plan a shorter moving path for MA. And in EPPA, MA does not need to move to the location of the RPs.

Finally, in order to aggregate the data of the nodes in SN_i to the corresponding RP, we construct an Improved Shortest Path tree(ISP tree) with a virtual root for each $SN_i(1 \le i \le K)$. GBA and EPPA aim at reducing the length of the moving path of MA, while the ISP tree reduces the energy consumption during collecting the data from the nodes in SN_i. In the following sections, we will introduce GBA, EPPA and the ISP tree in detail.

4.2. GBA

The selection of RPs and planning moving path for MA are two separate steps in traditional data collection algorithms. They first select RPs by clustering according to node distance, node density, etc [32–34]. Then, the visiting of RPs are transformed into a TSP problem to obtain the moving path of MA. Such algorithms narrow the solution space of suitable RPs and moving paths of MA, which undoubtedly omit many feasible and efficient solutions.

In order to avoid the above shortcomings, we combine selecting RPs and planning moving path for MA into one step. For each sub-network $SN_i(1 \le i \le K)$, MA needs to traverse one node in each SN_i and traverse this node only once. The optimization goal is to minimize the length of MA's traversal path. This is essentially a GTSP problem. The node being traversed in each $SN_i(1 \le i \le K)$ is essentially the RP of SN_i. Therefore, we found that selecting RPs and finding MA's moving path can be translated into a GTSP problem.

The GTSP problem is a typical NP-hard problem [36]. We use multiple heuristic algorithms proposed in [36–39] to find suitable RPs and efficient moving path of MA, including Ant Colony optimization (ACO), Genetic Algorithm (GA), Simulated Annealing (SA) and Tabu search (Tabu). The proposed algorithms based on ACO, GA, SA and Tabu are shown in Algorithm 1, 2, 3, 4 respectively.

Due to space limitations, we take Algorithm 2 as an example to illustrate how to calculate the RPs and the moving path of MA using GA. There are n_i sensor nodes in each $SN_i(1 \le i \le K)$. We use $s_{i_m,j_{i_m}}$ to represent the sensor node numbered j_{i_m} in SN_{i_m}, where $1 \le i_m \le K$, and $1 \le j_{i_m} \le n_{i_m}$. The traversal path to visit RPs is represented as $[s_{i_1,j_{i_1}} \to s_{i_2,j_{i_2}} \to \cdots \to s_{i_K,j_{i_K}} \to s_{i_1,j_{i_1}}]$. As shown in Fig. 3, There are 6 sub-networks in the partitioned WSNs, a feasible traversal path is shaped as $[s_{2,3} \to s_{3,5} \to s_{5,3} \to s_{4,2} \to s_{1,4} \to s_{6,2} \to s_{2,3}]$. To conveniently describe the details of GA, we introduce two definitions: the sub-network sequence and the sensor node sequence. The sub-network sequence is shaped like $(i_1, i_2, \ldots, i_m, \ldots, i_K)$ where $1 \le i_m \le K$. It represents the order MA traverses the sub-networks. The sensor node sequence is shaped like $(j_{i_1}, j_{i_2}, \ldots, j_{i_m}, \ldots, j_{i_K})$

Algorithm 1 Calculating RPs and MA's traversal path based on ACO

Input:
(x, y),the coordinate of each sensor node in all SN_i,$1 \le i \le K$;
Output:
SN_{rp}, the set of RP; $[(x_1, y_1) \to (x_2, y_2) \to \ldots \to (x_K, y_K)]$, the traversal path to visit these RPs;

for $ietr \leftarrow 0$ to $iter_{num}$ **do**
  Initialize pheromone matrix M_{phe} and path matrix M_{path};
  Randomly assign node in SN_i as initial location of ants;
  Calculate pheromone probability $prob_{phe}$;
  Select node in next SN_i to be visited according to $prob_{phe}$;
  Update SN_{rp}:add the selected node to SN_{rp};
  Update traversal path:add the selected node to traversal path;
  if ants not traverse every SN_i **then**
    Jump to 2
  end if
  Record SN_{rp} and traversal path;
  update $prob_{phe}$;
end for
return SN_{rp} and traversal path;

Algorithm 2 Calculating RPs and MA's traversal path based on GA

Input:
(x, y), the coordinate of each sensor node in all $SN_i(1 \le i \le K)$;
Output:
SN_{rp}, the set of RP; $[(x_1, y_1) \to (x_2, y_2) \to \ldots \to (x_K, y_K)]$,the traversal path to visit these RPs;

Randomly generated P^0;
for $iter \leftarrow 0$ to $iter_{num}$ **do**
  for $i \leftarrow 1$ to N **do**
    Calculate $f(h_i^{iter}, b_i^{iter})$;
    Add (h_i^{iter}, b_i^{iter}) to P^{iter+1};
  end for
  for $i \leftarrow 1$ to $\lfloor N/2 \rfloor$ **do**
    Select p_1 and p_2 in P^{iter} by using roulette wheel;
    Randomly generate $p \in [0, 1]$;
    if $p \ge P_c$ **then**
      Crossover operation on p_1 and p_2;
      Add ch_1 and ch_2 to P^{iter+1};
    end if
  end for
  for $i \leftarrow 1$ to N **do**
    Randomly generate $p \in [0, 1]$;
    if $p \ge P_m$ **then**
      Mutation operation on h_i^{iter} or b_i^{iter};
      Add mutated individual to P^{iter+1};
    end if
  end for
  for individual in P^{iter+1} **do**
    Select N individuals with best f;
  end for
end for
Record the best (h_i^{iter}, b_i^{iter}) in $P^{iter_{num}}$;
return SN_{rp} and traversal path;

Algorithm 3 Calculating RPs and MA's traversal path based on SA

Input:
(x, y), the coordinate of each sensor node in all $SN_i(1 \leq i \leq K)$;
Output:
SN_{rp}, the set of RP; $[(x_1, y_1) \rightarrow (x_2, y_2) \rightarrow \ldots \rightarrow (x_K, y_K)]$,the traversal path to visit these RPs;

Generate a random initial traversal path;
while $T_{start} > T_{end}$ **do**
  for $iter \leftarrow 0$ to $iter_{num}$ **do**
    Randomly replace (x_i, y_i) in traversal path with the node in SN_i;
    Randomly swap two nodes in traversal path;
    Calculate the path distance difference $\triangle$;
    if $\triangle \leqslant 0$ **then**
      Accept new traversal path;
    end if
    if $\triangle > 0$ **then**
      Probably accepte new traversal path;
    end if
    record SN_{rp} and traversal path;
  end for
  Update T_{start};
end while
return SN_{rp} and traversal path;

Algorithm 4 Calculating RPs and MA's traversal path based on Tabu

Input:
(x, y),the coordinate of each sensor node in all $SN_i(1 \leq i \leq K)$;
Output:
SN_{rp}, the set of RP; $[(x_1, y_1) \rightarrow (x_2, y_2) \rightarrow \ldots \rightarrow (x_K, y_K)]$, the traversal path to visit these RPs;

Setting up contraindication table and contraindication length;
Randomly generate the initial traversal path;
for $iter \leftarrow 0$ to $iter_{num}$ **do**
  Generating candidate solutions from neighborhood search;
  Comparison of the optimal traversal path among the candidate paths with the current optimal path;
  The traversal path with better effect is added to the taboo table;
  Update the taboo length;
end for
return SN_{rp} and traversal path;

where $1 \leq j_{i_m} \leq n_{i_m}$. It represents the order MA traverses the RPs in sub-networks. For example, in Fig. 3, the sub-network sequence is (2, 3, 5, 4, 1, 6, 2) and the sensor node sequence is (3, 5, 3, 2, 4, 2, 3). The specific steps for calculating RPs and MA's moving path using GA are as follows.

(1) Initialization. First, generate the $0-th$ generation population P^0 which contains N individuals. Each individual represents a feasible traversal path. We use a tuple (h_i^j, b_i^j) to represent an individual, where $1 \leq i \leq N$, $1 \leq j \leq iter_{num}$ ($iter_{num}$ is a constant). h_i^j is the sub-network sequence of the ith individual in the jth population, b_i^j is the corresponding sensor node sequence. In the initialization phase, $P^0 = \{(h_1^0, b_1^0), (h_2^0, b_2^0), \ldots, (h_N^0, b_N^0)\}$, the sub-network sequence and the sensor node sequence are generated randomly.

(2) Calculation of the population fitness. In each round of evolution, the fitness of each individual in the population is first calculated. The fitness of individual (h_i^j, b_i^j) is defined as $f(h_i^j, b_i^j) = \frac{1}{L(h_i^j, b_i^j)}$, where $L(h_i^j, b_i^j)$ is the length of the traversal path.

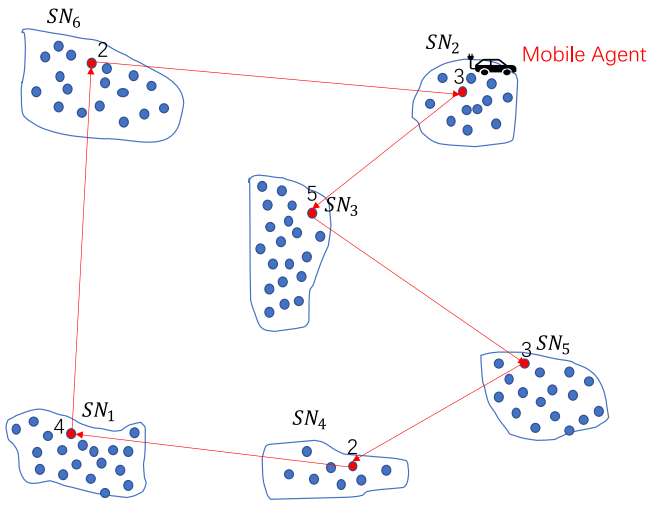

Fig. 3. A feasible traversal path.

(3) Selection operation. According to roulette wheel selection, two individuals in the current population are randomly selected for the crossover operation, and these two individuals are called two parents. The two new individuals resulting from the crossover operation between the two parents are called offsprings. This process is performed $\lfloor N/2 \rfloor$ times to produce a total of N or $N-1$ offsprings.

(4) Crossover operation. First, a random probability p in the range [0, 1] is generated. When p is greater the crossover probability P_c (P_c is a constant), the crossover operation is performed. Second, using partial crossover operators, swapping partial gene fragments of the sub-network sequence of the two parents p_1 and p_2. Take Fig. 4(a) as an example, the sub-network sequence in p_1 is (1, 2, 3, 4, 5, 6, 7, 8), and the sub-network sequence in p_2 is (3, 5, 8, 1, 7, 4, 2, 6), swapping the two fragments (2, 3, 4, 5) in p_1 with (5, 8, 1, 7) in p_2 is shown in Fig. 4(b). After gene fragments are swapped, two offsprings ch_1 and ch_2 are generated. However, we can see that there is a conflict in the sub-network sequence of ch_1 and ch_2. Then, we establish a mapping relationship $2 \longleftrightarrow 5, 3 \longleftrightarrow 8, 4 \longleftrightarrow 1, 5 \longleftrightarrow 7$ based on the two sets of genes exchanged. Finally, the conflict in ch_1 and ch_2 are eliminated by the mapping relationship, and the new sub-network sequence is obtained as shown in Fig. 4(c).

(5) Mutation operation. First, a random probability p is generated in the range [0,1]. When p is greater the mutation probability P_m(P_m is a constant), the mutation operation is performed. We design two mutation operators to mutate the sub-network sequence and the sensor node sequence of individuals in the current generation respectively. For example, the sub-network sequence of an individual is (1, 2, 3, 4, 5, 6), and the gene fragments from position 2 to position 4 of the sub-network sequence are intercepted to obtain (2, 3, 4). The sub-network sequence is intercepted and rejoined to get (1, 5, 6). The mutated sub-network sequence (1, 5, 2, 3, 4, 6) is obtained after inserting the fragment (2, 3, 4) into the 2th fragment of (1, 5, 6). The sub-network sequence is (1, 2, 3, 4, 5, 6), and the corresponding sensor node sequence is (2, 3, 3, 1, 4, 5). We use $node_1$ in SN_2 instead of $node_3$ and $node_2$ in SN_4 instead of $node_1$, the mutated sensor node sequence becomes (2, 1, 3, 2, 4, 5).

(6) Generation of new population. The individuals of the current population, the offspring produced by the crossover operation and the new individuals produced by the mutation operation together constitute the new population. The size of the

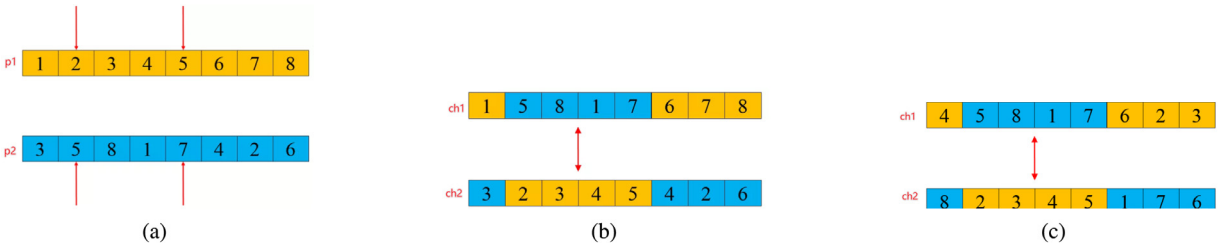

Fig. 4. Crossover operation. (a) Swap fragments. (b) There is a conflict in the traversal sequence. (c) The new traversal sequence obtained after deconfliction.

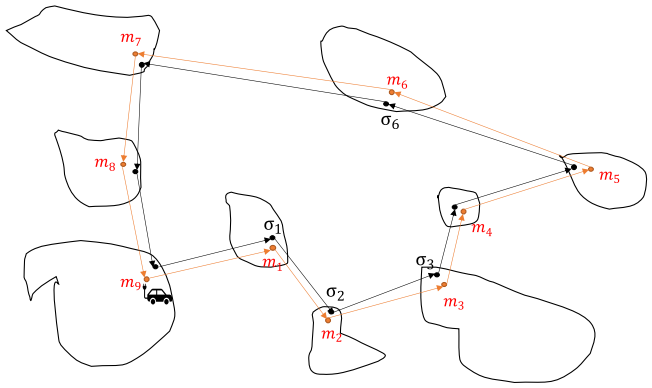

Fig. 5. MA can receive data as long as it moves within the communication range.

new population will reach $[2N, 3N]$, and N optimal individuals are selected from the new population to represent the next population.

4.3. EPPA

In GBA, MA moves directly to the locations of RPs to collect data. However, in practice, MA can receive data as long as it moves within the communication range of RPs. This leaves room for further optimization of MA's moving path. For example, as shown in Fig. 5, $m_i(1 \le i \le 9)$ is the RP of SN_i, the red route is the traversal path of RPs obtained by GBA in Section 4.2. $\sigma_i(1 \le i \le 9)$ is the point within the communication range of m_i, if MA moves along the black route, MA can also collect data in SN_i. It is obvious that this moving path of MA is shorter than the traversal path of RPs obtained by GBA. Therefore, after obtaining the RPs by GBA, we propose an efficient path planning algorithm(EPPA).

EPPA constructs convex hull to plan a more effective moving path for MA to collect data from each RP. A convex hull of a set of points S is defined as the smallest polygon containing S, where for any two points $s_i, s_j \in S$, the line segment from s_i to s_j does not intersect the boundary of the polygon [40]. A shortest loop path containing part of the RPs can be found firstly by constructing convex hull. Then, other RPs join the current path under the rule of length increment minimization. Therefore, We can quickly get a traversal path of RPs as short as possible. Finally, we construct a moving path of MA which is contained by the traversal path.

Algorithm 5 shows the details of EPPA, which includes the following steps.

(1) Construct a complete convex hull. First construct a convex hull for RPs, we use CH_{rp} to represent the set of the vertices of the convex hull. The construction process consists of ascending phase and descending phase, as shown in Figs. 6(a), 6(b) respectively.

Ascending phase: The coordinate of the node m_i in SN_{rp} is denoted by (x_i, y_i), where $1 \le i \le K$. In order to construct a convex hull, we first select the node with smallest vertical coordinate and the node with largest vertical coordinate. These two nodes are denoted as m_l and m_u. Second, select all nodes with vertical coordinates greater than y_l. Third, calculate the angle θ between m_l and these nodes. Fourth, select the node m_i that makes the angle $\theta_{l,i}$ smallest, then, m_i will become a vertex of the convex hull. Repeat above processes until a vertices set $(m_l, \ldots, m_i, m_j, \ldots, m_u)$ is found.

Descending phase: First, select all nodes whose vertical coordinates are smaller than y_u. Second, calculate the angle θ between m_u and these nodes. Third, select the node m_j that makes the angle $\theta_{u,j}$ smallest, m_j will become a vertex of the convex hull. These processes are repeated until another vertices set

Algorithm 5 Plan moving path for MA

Input:
SN_{rp},the set of RPs $m_1, m_2, \ldots m_k, \ldots m_K$ obtained by GBA,where $m_i = (x_i, y_i)$.
Output:
MA's moving path $[\sigma_1 \to \sigma_2 \ldots \to \sigma_K]$.

for $i \leftarrow 1$ to K **do**
  Select point m_l with the smallest y_i;
  Select point m_u with the largest y_i;
end for
Add m_l to CH_{rp};
while m_l does not encounter m_u **do**
  Compute $\theta_{i,j}$ of $\overrightarrow{m_i m_j}$;
  Update point m_j with the smallest $\theta_{i,j}$;
  Aaa m_j to CH_{rp};
end while
Add m_u to CH_{rp};
while m_u does not encounter m_l **do**
  Compute $\theta_{i,j}$ of $\overrightarrow{m_i m_j}$;
  Update point m_j with the smallest $\theta_{i,j}$;
  Add m_j to CH_{rp};
end while
if $CH_{rp} = SN_{rp}$ **then**
  for $i \leftarrow 1$ to K **do**
    Find m_{left} and m_{right} of m_i in the convex hull;
    Find σ_i such that $\overrightarrow{m_i \sigma_i} \perp \overrightarrow{m_{left} m_{right}}$ and $d(m_i, \sigma_i) \le r$;
    Add σ_i to the moving path of MA;
  end for
end if
if $CH_{rp} \subseteq SN_{rp}$ **then**
  for m_t not in CH_{rp} **do**
    Connect m_t with (m_i, m_j) such that $m_t = \arg\min[d(m_i, m_t) + d(m_t, m_j) - d(m_i, m_j)]$;
  end for
  for m_i not in CH_{rp} **do**
    Find σ_i such that $\overrightarrow{m_i \sigma_i} \perp \overrightarrow{m_l m_u}$ and $d(m_i, \sigma_i) \le r$;
    Add σ_i to the moving path of MA;
  end for
end if
return MA's moving path

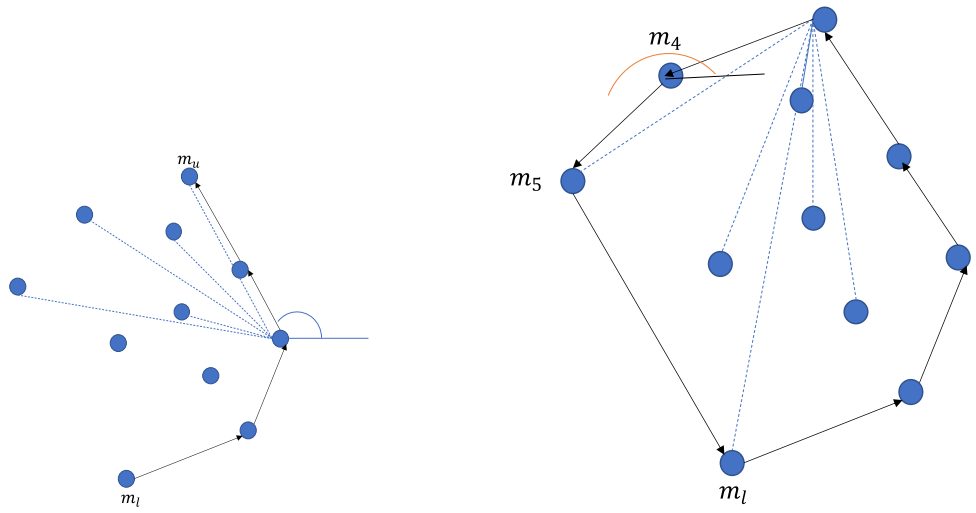

Fig. 6. The construction process of convex hull. (a) Ascending phase. (b) Descending phase.

$(m_u, \ldots, m_j, m_i, \ldots, m_l)$ is found. Where $\theta_{i,j}$ is calculated by the following equation.

$$\theta i, j = \begin{cases} \arccos \frac{x_j - x_i}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}} & y_i \leqslant y_j \\ +\infty & else \end{cases} \tag{8}$$

and

$$\theta i, j = \begin{cases} 2\pi - \arccos \frac{x_j - x_i}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}} & y_i \geqslant y_j \\ +\infty & else \end{cases} \tag{9}$$

After combining the set of vertices obtained from the ascending and descending phases, a complete convex hull can be generated. For example, in Fig. 6(a), m_l is the node with smallest vertical coordinate and m_u is the node with largest vertical coordinate. These two nodes first become the vertices of the convex hull. Then by calculating the angle θ, m_1, m_2 and m_3 are added to the convex hull. In Fig. 6(b), m_4, m_5 are chosen as the vertices of the convex hull. Thus, we get a complete convex hull $[m_l \rightarrow m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_u \rightarrow m_4 \rightarrow m_5 \rightarrow m_l]$. We use CH_{rp} to represent the set of vertices in the convex hull, and $CH_{rp} = \{m_l, m_1, m_2, m_3, m_u, m_4, m_5\}$.

(2) Plan moving path for MA. There are two cases: $CH_{rp} = SN_{rp}$ or $CH_{rp} \subseteq SN_{rp}$.

Case 1($CH_{rp} = SN_{rp}$): All RPs are the vertices of the convex hull, as shown in Fig. 7(a). This convex hull can be seen as the traversal path of RPs. Fig. 7(b) is used as an example to illustrate the process of planning a moving path for MA. First, select a vertex of the convex hull, such as m_1 in Fig. 7(c). Second, connect two vertices adjacent to m_1 to form a triangle, make a vertical line through m_1, and the point σ_1 where the vertical line intersects with the communication circle is the point of MA's moving path. Repeat this process to find all points of MA's moving path, and the moving path of MA is obtained by traversing these points with the order of traversing RPs. The moving path obtained after the above processes is shown in Fig. 7(c). MA's moving path is $[\sigma_1 \rightarrow \sigma_2 \rightarrow \sigma_3 \rightarrow \sigma_4 \rightarrow \sigma_5 \rightarrow \sigma_1]$.

Case 2($CH_{rp} \subseteq SN_{rp}$): There exists RPs which are not the vertices of convex hull,as shown in Fig. 8(a). $[m_1 \rightarrow m_2 \rightarrow m_4 \rightarrow m_5 \rightarrow m_6 \rightarrow m_7 \rightarrow m_8 \rightarrow m_1]$ is a convex hull, σ_i is calculated in case 1. The vertices of the convex hull can form several pairs, such as (m_1, m_2), (m_2, m_4), …, (m_8, m_9). We choose the non-vertex m_t to connect the vertex pair (m_i, m_j), where $m_t = \arg\min[d(m_i, m_t) + d(m_t, m_j) - d(m_i, m_j)]$. As shown in Fig. 8(b), m_9 connects the vertex pair (m_8, m_1) and m_3 connects (m_2, m_4), forming a traversal path $[m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4 \rightarrow m_5 \rightarrow m_6 \rightarrow m_7 \rightarrow m_8 \rightarrow m_9 \rightarrow m_1]$ to visit the RPs.

Then we calculate the path point in the communication range of each non-vertex. First, choose the non-vertex m_t, the vertices m_l and m_u with smallest vertical coordinate and largest vertical coordinate respectively. These three points form a triangle $\triangle m_t m_l m_u$. Second, make a vertical line through m_t to the side $\overrightarrow{m_l m_u}$, and the point σ_t where the vertical line intersects with the communication circle is the point of MA's moving path. As shown in Fig. 9(a), m_1, m_5 and m_9 can form a triangle. In this triangle, a vertical line is made through m_9, the point σ_9 where the vertical line intersects the communication circle is the path point. Repeat this process to get all path points. Then follow the traversal path of RPs to get a new moving path of MA. As shown in Fig. 9(b), MA's moving path $[\sigma_1 \rightarrow \sigma_2 \rightarrow \cdots \rightarrow \sigma_8 \rightarrow \sigma_9 \rightarrow \sigma_1]$ calculated by EPPA is included in the traversal path of RPs, so the length of moving path can be reduced.

4.4. How to aggregate data from nodes to RP

In order to aggregate data from nodes in $SN_i (1 \leq i \leq K)$ to m_i(the RP in SN_i), a original method is to construct a Shortest Path tree(SP tree) with RP as the root node. There are three types of nodes in SP tree: the root node, the internal node and the leaf node. RP acts as the root node. Data always flows from the leaf to RP through the internal node. The internal node must aggregate its own data with the data received from its children before it can forward to the next internal node. However, aggregating all data into m_i imposes huge energy overhead on m_i.

To cope with the above shortcomings,we propose an Improved SP tree(ISP tree). Considering that when MA moves to the communication range of RP, MA is able to collect data in the RP. Meanwhile, MA can also collect data from those nodes which are able to communicate with MA. As shown in Fig. 10, the sensor nodes within the blue outline can communicate directly with MA for data transmission. MA moves to σ to collect data in RP, at this time, MA can collect the data in nodes s_2 and s_3.

Based on the above observations, for each SN_i, we introduce σ_i which obtained by EPPA in part C as the virtual root node to construct a better SP tree.

In SN_i, we first take σ_i as the root, and the nodes which can directly communicate with σ_i are selected as RPs. Second, find the shortest paths from root to non-RP nodes based on the

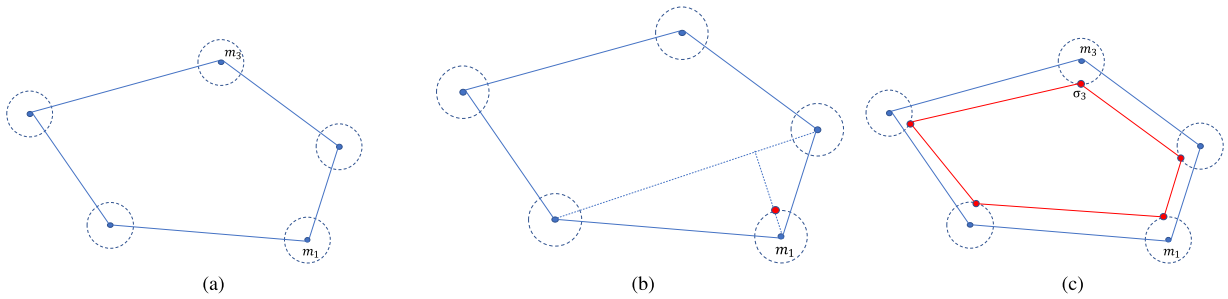

Fig. 7. Plan a moving path for MA. (a) All RPs are the vertices of the convex hull. (b) Find path points on the communication circle of the RPs. (c) Complete path for MA.

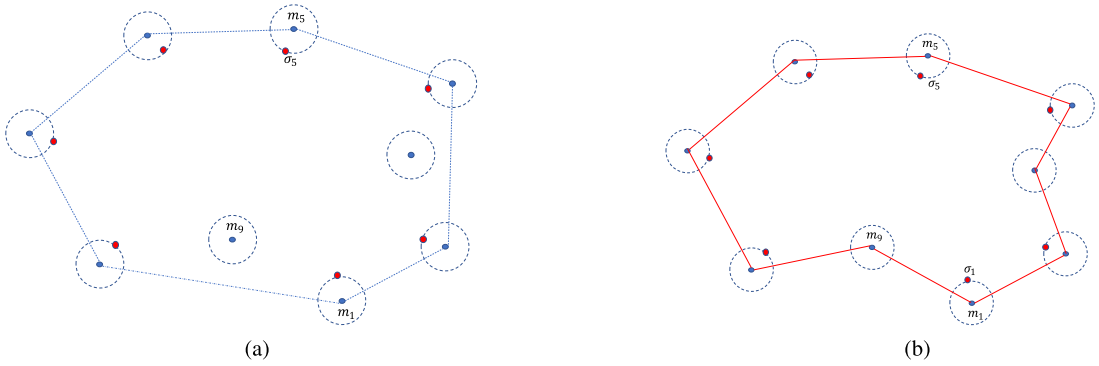

Fig. 8. Choose non-vertex to connect vertex pair. (a) Some RPs are not the vertices of the convex hull. (b) Selection of the nearest vertex pair.

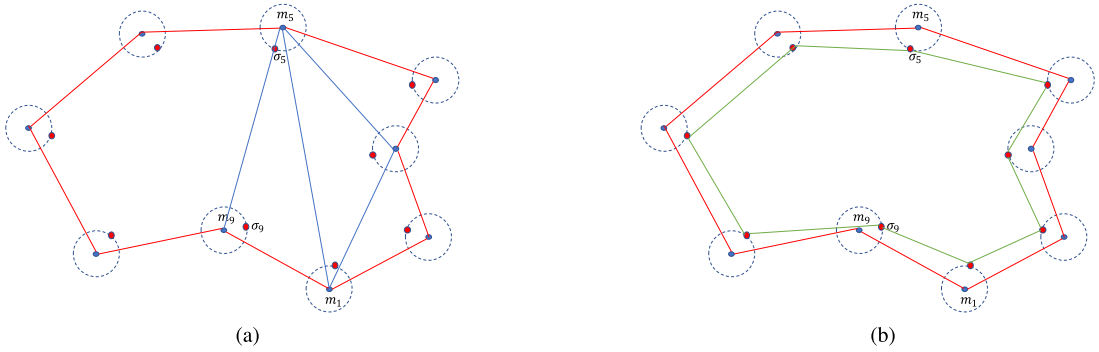

Fig. 9. Plan a moving path for MA. (a) Find path points on the communication circle of the RPs. (b) Complete path for MA.

Dijkstra algorithm. The ISP tree ensures that the sum of hop distances between leaf nodes and RPs is small, thus achieving the goal of saving energy in SN_i. Algorithm 6 shows the details of constructing ISP tree with virtual root.

The energy overhead of data transmission with two different routing trees is analyzed as follows. As shown in Fig. 11(a) each node generates l bits of data, and the energy consumption of SN_i comes from two main sources: (1) the energy consumed by sending data; (2) the energy consumed by receiving data. While transmitting data along the routing $s_5 \rightarrow s_4 \rightarrow s_3 \rightarrow s_2 \rightarrow s_1$, the energy consumed in the process of sending data E_t is

$$E_t = \gamma_1 l \sum_{i=1}^{4} i + \gamma_2 l \sum_{i=1}^{4} (5-i) d(s_{i+1}, s_i)^{\epsilon} \tag{10}$$

The energy consumed in the process of receiving data E_r is

$$E_r = 4\gamma_1 l \tag{11}$$

From the above equations, it is easy to see that the total energy consumption of SN_i is related to the sum of hop distances between nodes. Therefore, reducing the sum of hop distances is the key to reduce the energy consumption. As shown in Fig. 11(b), there are four routings for data transmission, $s_5 \rightarrow s_4 \rightarrow s_3 \rightarrow s_2$, $s_9 \rightarrow s_8 \rightarrow s_7 \rightarrow s_6$, $s_{10} \rightarrow s_7 \rightarrow s_6$, and $s_{11} \rightarrow s_7 \rightarrow s_6$. Obviously, the sum of hop distances decrease and the energy consumption of SN_i is reduced. In addition, RP s_1 receives less data than before, so the energy burden of s_1 becomes smaller.

4.5. Complexity analysis

The complexity of our proposed algorithm is analyzed in this subsection.

Taking Algorithm 2 as an example, the complexity of constructing initial population which contains N individuals is $O(Nn)$. The complexity of calculating the fitness of population is $O(NK)$. The complexity of selection operation for two individuals is $O(N)$, and the complexity of crossover operation for two individuals is $O(K)$, so the complexity of crossover operation for population is $O(N^2K)$. The complexity of mutation operation for individual is

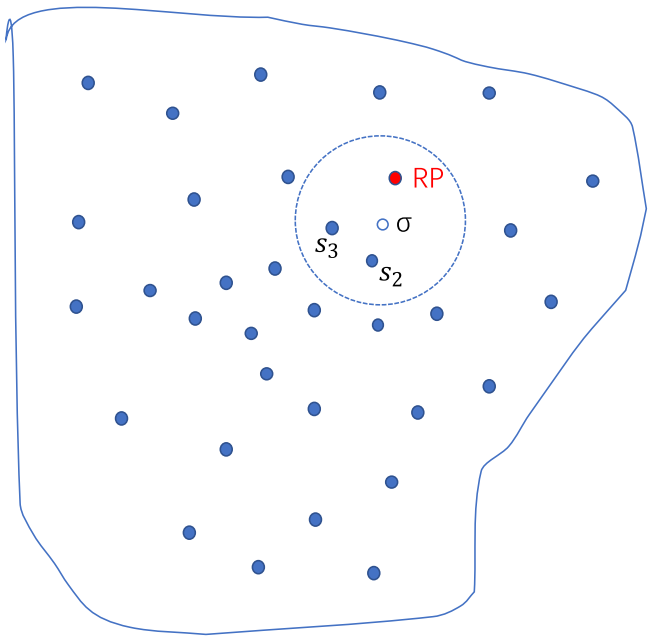

Fig. 10. MA can communicate directly with multiple nodes.

Algorithm 6 Aggregating data from nodes to RPs

Input:
(x, y),the coordinate of each sensor node in all SN_i,$1 \leq i \leq K$; SN_{rp},the set of RPs $m_1, m_2, \ldots m_k, \ldots m_K$ obtained by GBA,where $m_i = (x_i, y_i)$; σ,the set of path points $\sigma_1, \sigma_2, \ldots \sigma_k, \ldots \sigma_K$ in MA's moving path obtained by EPPA, where $\sigma_j = (x_j, y_j)$.

Output:
An ISP tree with a virtual root.

for $k \leftarrow 1$ to K **do**
  Add σ_k to S;
  for s_i in SN_k **do**
    if $d(s_i, m_k) \leq r$ **then**
      Add s_i to S;
    end if
    if $d(s_i, m_k) > r$ **then**
      Add s_i to U;
    end if
  end for
  while U is not empty **do**
    Select s_i from U such that min $d(s_i, S)$;
    Add s_i to S;
    Remove s_i from U;
    Update the shortest path from s_j to σ_k,where $s_j \in U$;
  end while
end for
return The ISP tree;

$O(K)$, so the complexity of mutation operation for population is $O(NK)$. Therefore, the complexity of Algorithm 2 meets

$$\begin{aligned} O(\text{Alg. 2}) &= O(Nn + NK + N^2K + NK) \\ &= O(Nn + (N+2)NK) \\ &= O(N^2 n) \end{aligned} \tag{12}$$

The complexity of computing m_l and m_u is $O(K)$. The complexity of constructing a complete convex hull is $O(K^2)$. The complexity of finding a moving path for MA is $O(K^2)$. Thus the complexity of EPPA is $O(K^2)$.

Table 2
Parameter and value.

Parameter	Value
The number of sub-networks K	5 to 10
The total number of nodes n	100 to 500
The communication range of nodes r	5 to 20
The time interval T for sensory data to remain valid	300 s
The sensory data size l	100 bits
The speed of MA v	10 m/s
The finite time T_f	3600 s

The complexity of constructing an ISP tree in a sub-network is $O(|SN_i|^2)$, where $|SN_i|$ denotes the number of sensor nodes in SN_i. Therefore, the complexity of constructing ISP tree for the entire partitioned WSNs is $O(K|SN_i|^2)$.

5. Experiments

In this section, we evaluate the performance of our algorithm in terms of the length of MA's moving path and the amount of data collected. It is compared with the typical data collection algorithm Objective-Variable Tour Planning (OVTP) [12]. These two algorithms aim at finding a moving path for MA to collect data in partitioned WSNs. In addition, we also evaluated the performance of the ISP tree and the computational cost of our algorithm. We implement our algorithm using the well-known simulator Cooja in [41]. The isolated $SN_i(1 \leq i \leq K)$ are generated randomly, the number of sub-networks K varies from 5 to 10, and the total number of nodes n varies from 100 to 500. It is assumed that each node generates 100 bits of packets at each time interval T. The maximum communication range r of each node varies between from 5 to 20. The speed of MA is $v = 10$ m/s. We first calculate the RPs and the moving path of MA using GBA proposed in Section 4.2. We then use EPPA to optimize the moving path of MA obtained by GBA. Therefore, we only show the final results obtained by EPPA. We propose four algorithms to calculate RPs and MA's moving path in GBA, in this section, we give their experimental results and compare them.

The relevant parameters required for the experiments are summarized in Table 2.

5.1. The length of MA's moving path

In this section, We generate several different sub-networks distributions. We perform 1000 experiments respectively when the values of K, n, r are different, and the average value is reported as the final result. Take Fig. 12 as an example, GBA indicates the best result among Algorithm 1, 2, 3, 4. GA-EPPA indicates that the base solution is first solved by GA, and then the base solution is optimized by EPPA.

(1) The impact of K

Fig. 12 shows the impact of K on the length of MA's moving path. It can be seen that the length of MA's moving path increases as K increases. When the number of sub-networks K increases, MA needs to move to a wider geographical area to access these sub-networks, and accordingly the length of moving path increases.

(2) The impact of n

Fig. 13 shows the impact of n on the length of MA's moving path. It can be seen that the length of MA's moving path decreases as n increases. This is because the length of MA's moving path is influenced by the positions of the nodes. The increase of n means that some nodes are located at better locations, thus the moving path of MA becomes shorter.

(3) The impact of r

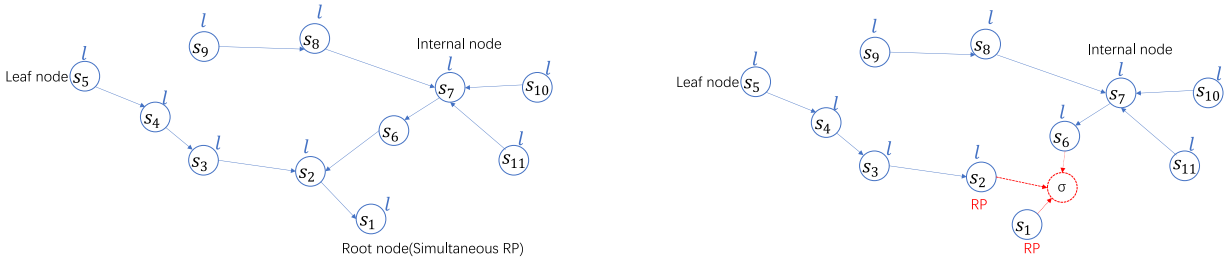

Fig. 11. Routing tree. (a) SP tree with RP as root. (b) SP tree with virtual root.

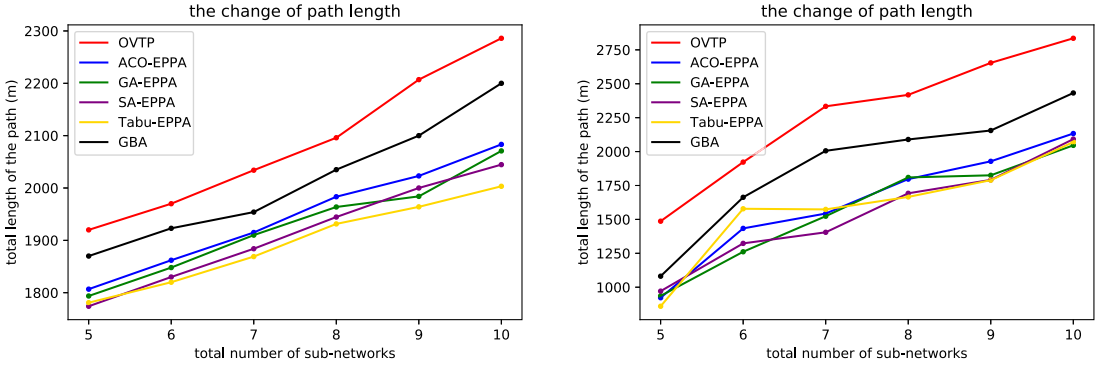

Fig. 12. The total length of the moving path of MA. (a) $T = 300$ s, $r = 15$ m, and the total number of nodes n is 200, respectively. (b) $T = 300$ s, $r = 15$ m, and the total number of nodes n is 400, respectively.

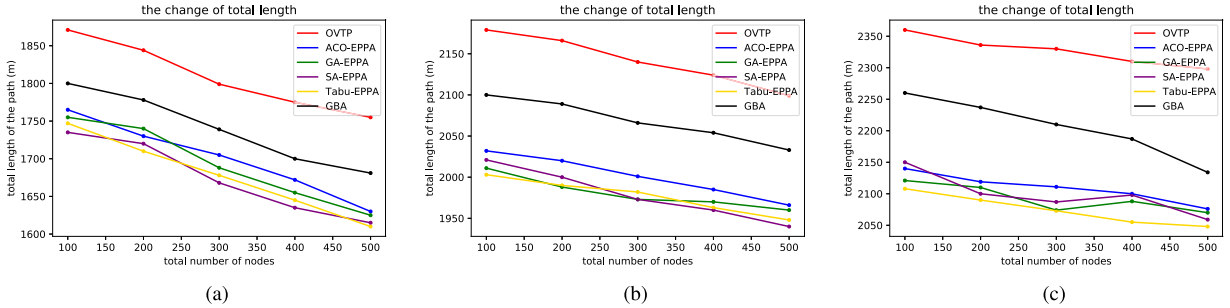

Fig. 13. The total length of the moving path of MA. (a) $T = 300$ s, $r = 15$ m, $K = 5$, respectively. (b) $T = 300$ s, $r = 15$ m, $K = 8$, respectively. (c) $T = 300$ s, $r = 15$ m, $K = 10$, respectively.

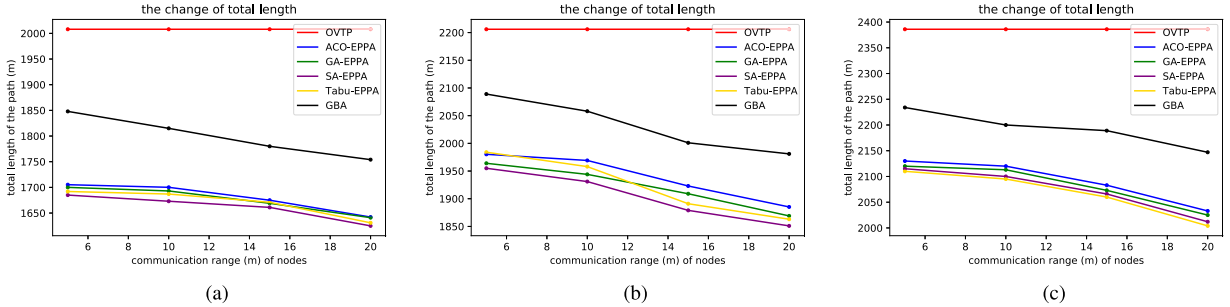

Fig. 14. The total length of the moving path of MA. (a) $T = 300$ s, $K = 5$, the total number of nodes n is 200, respectively. (b) $T = 300$ s, $K = 8$, the total number of nodes n is 300, respectively. (c) $T = 300$ s, $K = 10$, the total number of nodes n is 400, respectively.

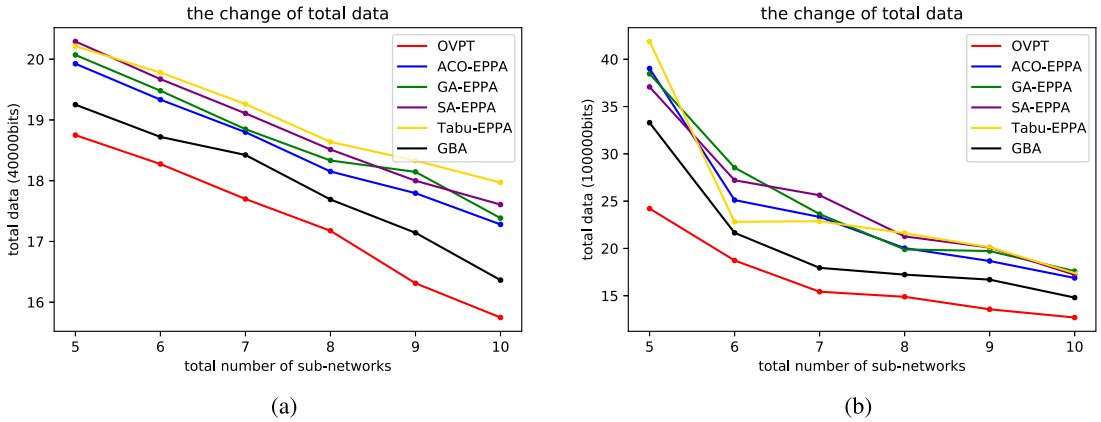

Fig. 15. The amount of data collected in T_f. (a) $T = 300$ s, $r = 15$ m, and the total number of nodes n is 200, respectively. (b) $T = 300$ s, $r = 15$ m, and the total number of nodes n is 400, respectively.

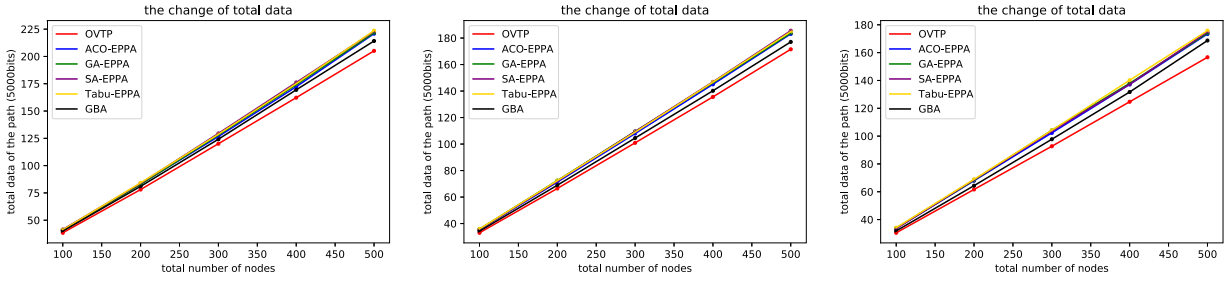

Fig. 16. The amount of data collected in T_f. (a) $T = 300$ s, $r = 15$ m, $K = 5$, respectively. (b) $T = 300$ s, $r = 15$ m, $K = 8$, respectively. (c) $T = 300$ s, $r = 15$ m, $K = 10$, respectively.

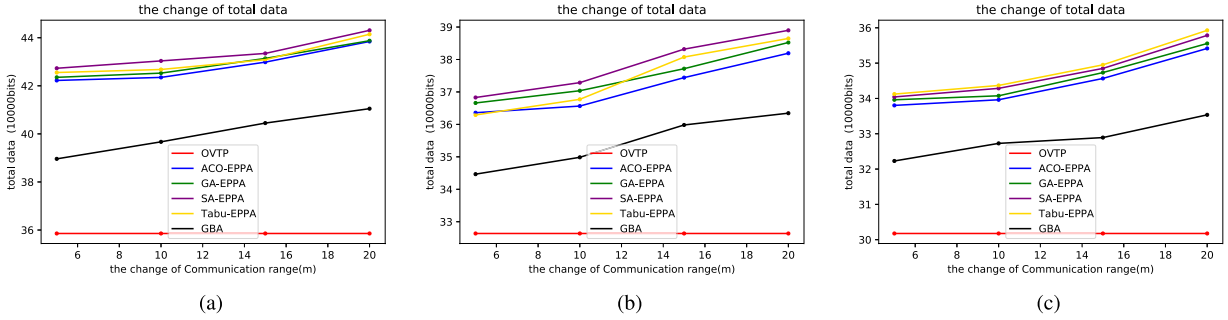

Fig. 17. The amount of data collected in T_f. (a) $T = 300$ s, $K = 5$, the total number of nodes n is 200, respectively. (b) $T = 300$ s, $K = 8$, the total number of nodes n is 300, respectively. (c) $T = 300$ s, $K = 10$, the total number of nodes n is 400, respectively.

Fig. 14 shows the impact of r on the length of MA's moving path. It can be seen that as r becomes larger, the moving path of MA obtained by our algorithm becomes longer. This is because MA moves directly to the location of RPs to collect data in OVTP, while in EPPA MA moves to a point within the communication range of RPs. As r becomes larger, EPPA is able to plan shorter moving path for MA by constructing convex hull. Likewise, in order to reduce the energy overhead on single RP, OVTP selects multiple RPs in each SN_i, and MA must move to the location of all RPs in each SN_i to collect data which results in longer moving path of MA. Under the consideration of reducing energy overhead on single RP, the ISP tree also selects multiple RPs, but MA always move to a point σ_i computed by the EPPA in each SN_i to collect data from RPs in SN_i, which results in shorter moving path of MA.

As shown in Figs. 12, 13, 14, our proposed GBA can improve nearly 16%–20% of the length of MA's moving path compared with OVTP on average. On the basis of the GBA, EPPA improve nearly 25%–35% of the length of MA's moving path compared with OVTP on average. Comparing Figs. 12, 13, 14, we can find that the length of MA's moving path calculated by our algorithm is always shorter than the length of moving path calculated by OVTP, regardless of the variation of K, n, and r. There are three main reasons. (1) The selection of RPs and planning moving path for MA are two separate steps in OVTP, while our GBA combines selecting RPs and planning moving path for MA into one step, the RPs and

the moving path of MA are obtained simultaneously by solving the GTSP problem. Therefore, GBA expands the solution space of suitable RPs and moving paths of MA, which obtains many feasible solutions omitted by OVTP. (2) EPPA further optimizes the moving path of MA obtained by using GBA, making the length of MA's moving path further reduced. And considering the effect of communication range, MA no longer moves directly to RP in EPPA, but chooses a location on the communication range of RP, which further reduces the length of MA's moving path. (3) Both OVTP and our algorithm select multiple RPs in one sub-networks for reducing energy consumption. However, in our algorithm, MA only needs to move to one path point in SN_i to collect data, while in OVTP MA moves to multiple RPs in SN_i, no doubt we plan a shorter moving path for MA.

5.2. The amount of data collected in a given time

Given a finite time T_f of 3600 s, we compare the amount of data collected by EPPA and OVTP within T_f. Each node generates 100 bits of data in time T, and it is assumed that MA collects data without data loss and the time used to collect data is negligible. We perform 1000 experiments respectively when the values of K, n, r are different, and the average value is reported as the final result.

(1) The impact of K

Fig. 15 shows the impact of K on the amount of data collected in T_f. It can be seen that as K increases, the amount of data collected by MA becomes less. This is because when n is constant, the total amount of data collected by the MA is related to the length of the moving path of MA. The smaller the K, the shorter the moving path of MA. The Shorter moving path of MA means that MA can perform more data collection tasks within T_f, thus MA can collect more data.

(2) The impact of n

Fig. 16 shows the impact of n on the amount of data collected in T_f. It can be seen that as n increases, the amount of data collected by MA becomes more. This because as n increases, the length of MA's moving path becomes shorter, MA can perform more data collection tasks in T_f. And an increasement in n means that MA is able to collect more data in one data collection task, thus MA can collect more data.

(3) The impact of r

Fig. 17 shows the impact of r on the amount of data collected in T_f. It can be seen that as r increases, the amount of data collected by MA using EPPA becomes more. This is because as r increases, the moving path of MA obtained by EPPA becomes shorter. The Shorter moving path of MA means that MA can perform more data collection tasks within T_f, thus MA can collect more data.

As shown in Figs. 15, 16, 17, our proposed GBA can improve nearly 8% − 10% on the amount of data collected with OVTP on average. On the basis of the GBA, EPPA improve nearly 15% on the amount of data collected compared with OVTP on average. Comparing Figs. 15, 16, 17, we can find that EPPA always performs better than OVTP in the amount of data collected, regardless of the variation of K, n, and r. The fundamental reason is that EPPA plans a more efficient moving path for MA comparing with OVTP.

5.3. The energy consumption

In this section, we evaluate the performance of the ISP tree. We generate several different nodes distributions on the simulator and perform 1000 experiments for $n = 100$ to $n = 500$ respectively, and the average value is reported as the final result.

As shown in Fig. 18, our proposed ISP tree can save nearly 20% of energy consumption compared with the SP tree on average.

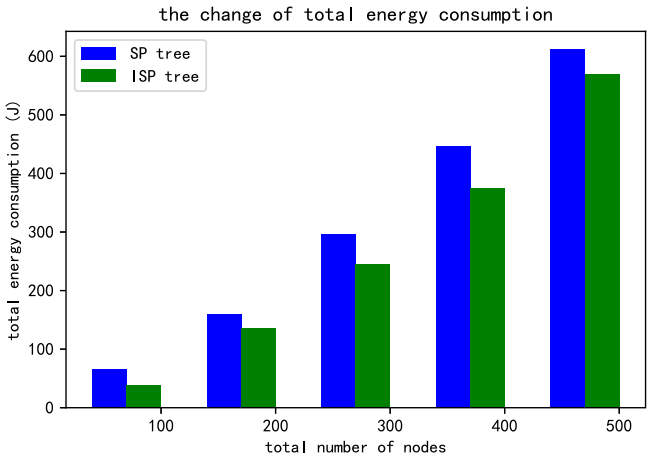

Fig. 18. The total energy consumption.

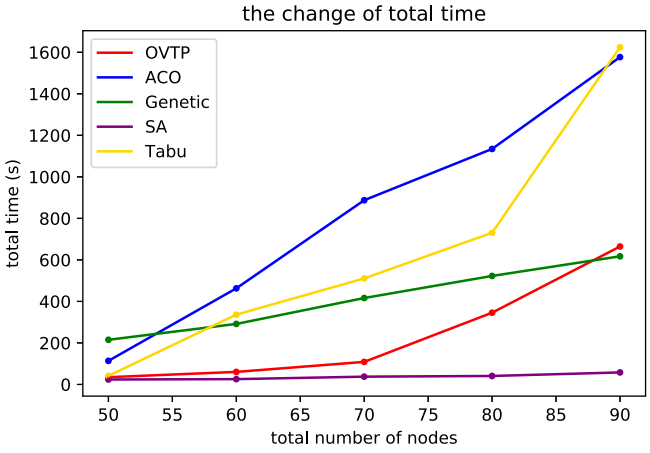

Fig. 19. The computational cost.

The energy consumption is further reduced when the node distribution is very dense and more nodes are selected as RPs. This is because the total energy consumption of partitioned WSNs is related to the sum of hop distances between nodes. Therefore, reducing the sum of hop distances is the key to reduce the energy consumption. ISP tree avoids multi-hop routing by selecting multiple RPs, therefore, the sum of hop distances decrease and the energy consumption is reduced.

5.4. The calculation cost of different algorithms

In this section, we evaluate the calculation cost of different algorithms. We generate several different nodes distributions on the simulator and perform 10 000 experiments for $n = 50$ to $n = 90$ respectively, and the average value is reported as the final result.

As shown in Fig. 19, the computational cost of all the algorithms increases with the number of sensor nodes. The computational cost of Genetic algorithm is comparable to that of OVTP, while the computational cost of SA is always lower than that of OVTP. However, GBA is superior to OVTP in terms of the length of MA's moving path and the data collection volume.

6. Conclusion

This paper studies the Data Collection Problem (DCP) in partitioned WSNs. To solve DCP, we first formalize DCP as an optimization problem. Then, DCP is transformed into a GTSP problem, and we propose a GTSP-Based data collection Algorithm (GBA). The RPs and the moving path of MA are calculated simultaneously by solving it. Based on the RPs and moving path of MA obtained by GBA, we further consider the effect of node's communication range on the moving path of MA and propose an Efficient Path Planning Algorithm(EPPA) to plan a more efficient moving path for MA. Finally, simulation experiments are conducted to evaluate our algorithm. The experiments show that our algorithm outperforms the existing algorithms in terms of the length of MA's moving path and the amount of data collection per unit time.

As future work, we aim to explore rendezvous planning in partitioned WSNs using multi-MA, and the impact of caching capacity of each node will be considered on the selection of RPs.

CRediT authorship contribution statement

Gongshun Min: Conceptualization, Methodology, Validation, Writing – original draft, Writing – review & editing, Formal analysis, Visualization. **Liang Liu:** Conceptualization, Supervision, Writing – review & editing. **Wenbin Zhai:** Writing – review & editing. **Zijie Wang:** Writing – review & editing. **Wanying Lu:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work is supported by the Open Fund of Key Laboratory of Civil Aviation Smart Airport Theory and System, Civil Aviation University of China under No. SATS202206, the National Natural Science Foundation of China under No. U20B2050, Public Service Platform for Basic Software and Hardware Supply Chain Guarantee under No. TC210804A, the ''National Key R&D Program of China' under No. 2021YFB2700500 and 2021YFB2700502.

References

[1] L. Atzori, A. Iera, G. Morabito, The internet of things: A survey, Comput. Netw. 54 (15) (2010) 2787–2805.
[2] P. Bellavista, G. Cardone, A. Corradi, L. Foschini, Convergence of MANET and WSN in IoT urban scenarios, IEEE Sens. J. 13 (10) (2013) 3558–3567.
[3] A. Ghasempour, Internet of things in smart grid: Architecture, applications, services, key technologies, and challenges, Invent. J. 4 (1) (2019) 1–12.
[4] M. Janidarmian, A.R. Fekr, K. Radecka, Z. Zilic, Multiobjective hierarchical classification using wearable sensors in a health application, IEEE Sens. J. 17 (5) (2017) 1421–1433.
[5] Y.C. Wang, Mobile sensor networks: system hardware and dispatch software, ACM Comput. Surv. 47 (1) (2014) 12:1–12:36.
[6] P. Tokekar, J.V. Hook, D. Mulla, V. Isler, Sensor planning for a symbiotic UAV and UGV system for precision agriculture, IEEE Trans. Robot. 32 (6) (2016) 1498–1511.
[7] Y.C. Wang, C.C. Yang, 3S-cart: a lightweight, interactive sensor-based cart for smart shopping in supermarkets, IEEE Sens. J. 16 (17) (2016) 6774–6781.
[8] Zijie Wang, et al., A UAV path and action planning algorithm for indoor localization information collection, in: 2020 IEEE International Symposium on Parallel and Distributed Processing with Applications, IEEE, 2020.
[9] Wen Weimin, et al., EAPC: Energy-aware path construction for data collection using mobile sink in wireless sensor networks, IEEE Sens. J. 18 (2) (2017) 890–901.
[10] Zhou Zhenyu, et al., When mobile crowd sensing meets UAV: Energy-efficient task assignment and route planning, IEEE Trans. Commun. 66 (11) (2018) 5526–5538.
[11] Yongyong Wei, Rong Zheng, Informative path planning for mobile sensing with reinforcement learning, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications, IEEE, 2020.
[12] Liu Xuxun, et al., Objective-variable tour planning for mobile data collection in partitioned sensor networks, IEEE Trans. Mob. Comput. 21 (1) (2020) 239–251.
[13] Wang Cong, et al., A novel framework of multi-hop wireless charging for sensor networks using resonant repeaters, IEEE Trans. Mob. Comput. 16 (3) (2016) 617–633.
[14] N.A. Pantazis, S.A. Nikolidakis, D.D. Vergados, Energyefficient routing protocols in wireless sensor networks: A survey, IEEE Commun. Surveys Tuts. 15 (2) (2013) 551–591.
[15] J. Ren, Y. Zhang, K. Zhang, A. Liu, J. Chen, X.S. Shen, Lifetime and energy hole evolution analysis in data-gathering wireless sensor networks, IEEE Trans. Ind. Inform. 12 (2) (2016) 788–800.
[16] Y.C. Wang, F.J. Wu, Y.C. Tseng, Mobility management algorithms and applications for mobile sensor networks, Wirel. Commun. Mobile Comput. 12 (1) (2012) 7–21.
[17] Y. Gu, F. Ren, Y. Ji, J. Li, The evolution of sink mobility management in wireless sensor networks: A survey, IEEE Commun. Surveys Tuts. 18 (1) (2016) 507–524.
[18] M. Zhao, M. Ma, Y. Yang, Efficient data gathering with mobile collectors and space-division multiple access technique in wireless sensor networks, IEEE Trans. Comput. 60 (3) (2011) 400–417.
[19] J. Tang, H. Huang, S. Guo, Y. Yangd, Dellat: delivery latency minimization in wireless sensor networks with mobile sink, J. Parallel Distrib. Comput. 83 (2015) 133–142.
[20] A.L. Henry-Labordere, The record balancing problem: A dynamic programming solution of a generalized traveling salesman problem, Revue Francaise D Inf. DeRecherche Oper. 3 (2) (1969) 43–49.
[21] Kevin Vicencio, Brian Davis, Iacopo Gentilini, Multi-goal path planning based on the generalized traveling salesman problem with neighborhoods, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014.
[22] W. Zhao, M. Ammar, E. Zegura, Controlling the mobility of multiple data transport ferries in a delay-tolerant network. in: Proc. IEEE 24th Annu. Joint Conf. IEEE Comput. Commun. Societies, 2005, pp. 1407–1418..
[23] R. Moazzez-Estanjini, J. Wang, I.C. Paschalidis, Scheduling mobile nodes for cooperative data transport in sensor networks, IEEE/ACM Trans. Netw. 21 (3) (2013) 974–989.
[24] Y.C. Wang, W.C. Peng, Y.C. Tseng, Energy-balanced dispatch of mobile sensors in a hybrid wireless sensor network, IEEE Trans. Parallel Distrib. Syst. 21 (12) (2010) 1836–1850.
[25] B. Yuan, M. Orlowska, S. Sadiq, On the optimal robot routing problem in wireless sensor networks, IEEE Trans. Knowl. Data Eng. 19 (9) (2007) 1252–1261.
[26] J. Tang, H. Huang, S. Guo, Y. Yang, Dellat: Delivery latency minimization in wireless sensor networks with mobile sink, J. Parallel Distrib. Comput. 83 (2015) 133–142.
[27] R. Sugihara, R.K. Gupta, Optimal speed control of mobile node for data collection in sensor networks, IEEE Trans. Mob. Comput. 9 (1) (2010) 127–139.
[28] L. He, J. Pan, J. Xu, A progressive approach to reducing data collection latency in wireless sensor networks with mobile elements, IEEE Trans. Mob. Comput. 12 (7) (2013) 1308–1320.
[29] E. Welzl, Smallest enclosing disks (balls and ellipsoids), New Results New Trends Comput. Sci. 555 (1991) 359–370.
[30] A. Chakrabarti, A. Sabharwal, B. Aazhang, Communication power optimization in a sensor network with a path-constrained mobile observer, ACM Trans. Sensor Netw. 2 (3) (2006) 297–324.
[31] S. Gao, H. Zhang, S.K. Das, Efficient data collection in wireless sensor networks with path-constrained mobile sinks, IEEE Trans. Mob. Comput. 10 (4) (2011) 592–608.

[32] G. Xing, T. Wang, W. Jia, M. Li, Rendezvous design algorithms for wireless sensor networks with a mobile base station. in: Proc. ACM Int. Symp. Mobile Ad Hoc Netw. Comput. 2008, pp. 231–240..
[33] G. Xing, T. Wang, Z. Xie, W. Jia, Rendezvous planning in wireless sensor networks with mobile elements, IEEE Trans. Mob. Comput. 7 (12) (2008) 1430–1443.
[34] K. Almiani, A. Viglas, L. Libman, Energy-efficient data gathering with tour length-constrained mobile elements in wireless sensor networks. in: Proc. IEEE Local Comput. Netw. Conf. 2010, pp. 582–589..
[35] H. Salarian, K.W. Chin, F. Naghdy, An energy-efficient mobile-sink path selection strategy for wireless sensor networks, IEEE Trans. Veh. Technol. 63 (5) (2014) 2407–2419.
[36] Oliviu Matei, Petrică Pop, An efficient genetic algorithm for solving the generalized traveling salesman problem, in: Proceedings of the 2010 IEEE 6th International Conference on Intelligent Computer Communication and Processing, IEEE, 2010.
[37] Kan Jun-man, Zhang Yi, Application of an improved ant colony optimization on generalized traveling salesman problem, Energy Procedia 17 (2012) 319–325.
[38] F. Vincent, Yu, et al., Adaptive neighborhood simulated annealing for the heterogeneous fleet vehicle routing problem with multiple cross-docks, Comput. Oper. Res. 129 (2021) 105205.
[39] Ílker Küçükoğlu, Reginald Dewil, Dirk Cattrysse, Hybrid simulated annealing and tabu search method for the electric travelling salesman problem with time windows and mixed charging rates, Expert Syst. Appl. 134 (2019) 279–303.
[40] David. Avis, David Bremner, Raimund Seidel, How good are convex hull algorithms? Comput. Geom. 7 (5–6) (1997) 265–301.
[41] B.A. Bagula, Zenville Erasmus, Iot Emulation with Cooja, ICTP-IoT workshop, 2015.

Gongshun Min received his Bachelor's degree in 2019, from the Nanjing University of Aeronautics and Astronautics, China. He is currently a master student in the college of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics and Key Laboratory of Civil Aviation Smart Airport Theory and System, Civil Aviation University of China,China. His research interests include distributed database and data collection in partitioned WSNs.

Liang Liu is currently an associate professor in the college of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics and Key Laboratory of Civil Aviation Smart Airport Theory and System, Civil Aviation University of China. His research interests include distributed computing, big data and system security. He received the B.S. degree in computer science from Northwestern Polytechnical University, Xi'an, Shanxi Province, China in 2005, and the Ph.D. degree in computer science from Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu Province, China in 2012.

Wenbin Zhai received the B.S. degree from Nanjing University Of Chinese Medicine, Nanjing, Jiangsu Province, China in 2020. He is currently working toward the M.S. degree in Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu Province, China and Key Laboratory of Civil Aviation Smart Airport Theory and System, Civil Aviation University of China. His research interest includes wireless sensor networks.

Zijie Wang received the B.S. degree from Wuhan University of Science and Technology, Wuhan, Hubei Province, China in 2019. He is currently working toward the M.S. degree in Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu Province, China. His research interest includes indoor UAV path planning technology.

Wanying Lu graduated from Henan Polytechnic University with a bachelor's degree in 2021. At present, she is studying for a master's degree in the collage of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics and Key Laboratory of Civil Aviation Smart Airport Theory and System, Civil Aviation University of China. Her main research direction is Time series big data storage and data mining.