Journal of Ambient Intelligence and Humanized Computing (2023) 14:12329–12344
https://doi.org/10.1007/s12652-023-04621-w

ORIGINAL RESEARCH

An energy-efficient scheme for maximizing data aggregation tree lifetime in wireless sensor network

Tran Cong Dao[1] · Nguyen Thi Tam[2] · Nguyen Quang Quy[1] · Huynh Thi Thanh Binh[1]

Received: 30 October 2020 / Accepted: 2 May 2023 / Published online: 26 June 2023

Abstract
In Wireless Sensor Networks (WSNs), the increasing network lifetime is an essential requirement. Energy consumption mainly takes place in data sensing, data transmissions, and data processing. In data transmission, the consumption is much higher as compared to the others. One of the approaches to reducing energy consumption is to construct a data aggregation tree. This paper investigates the problem of maximizing network lifetime by minimizing the total transmission and reception energy consumption by all sensor nodes. We also propose a heuristic algorithm, called Reduce Redundant Packet Tree (RRPT), to reduce the redundant packets in the network. This algorithm is based on the observation that the fewer the number of packets, the less energy consumption is. Following RRPT, we propose an Evolutionary Algorithm (EA), namely PGA, which adopts RRPT as a heuristic initialization for further performance improvement. We conduct several comprehensive experiments to verify the performance of our proposed algorithms. The experimental results demonstrate that our PGA can perform significantly better than the previous algorithms with 99% reliability through statistical tests in terms of solution quality.

Keywords Network lifetime · Data aggregation tree · Wireless sensor network · Evolutionary algorithm

1 Introduction

Wireless Sensor Networks (WSNs) are constituted by hundreds to thousands of sensor nodes whose main task is to sense data such as light, temperature, and humidity from the environment or the interesting regions. These data, then, will be reported to the base station or sink via wireless connections. These days, in the era of the Internet of Things (IoT), there is a plethora of applications using WSNs in not only modern smart cities but in other smart environments as well, covering virtually every field ranging from industry, military, smart farming in agriculture, and so on. The energy of WSNs is mainly consumed for data transmission tasks instead of computation tasks. Therefore, if the energy of the sensor node is exhausted, the data will not be transmitted to the sink or base station. The direct transmission of each data packet from sensor nodes to the base station results in high data redundancy and a high communication load. Fortunately, data collection can address this problem through the combination of data packets. Recently, data aggregation methods are attracting special attention from loads of researchers and experts. In the methods, the sensor node must receive packages from its children and then forward the received packets and its sensing packet to its parents. The key idea generally is to combine the data coming from different sources, which helps to reduce redundancy and minimize the number of transmissions, thereby prolonging the network lifetime.

In Kuo et al. (2015), the authors have proven that finding a data aggregation tree with minimum energy cost in wireless sensor networks is NP-complete. They proposed the 2-approximation to solve this problem. Although this algorithm is near-optimal, it may incur more costs if the number

✉ Huynh Thi Thanh Binh
binhht@soict.hust.edu.vn

Tran Cong Dao
daotc.bk@gmail.com

Nguyen Thi Tam
tamnt@vnu.edu.vn

Nguyen Quang Quy
pkioqcbn@gmail.com

[1] Hanoi University of Science and Technology, Hanoi, Vietnam

[2] VNU University of Science, Hanoi, Vietnam

of redundant packets in the shortest path tree increases. Following that, Tam et al. (2020) proposed a novel multi-factorial evolutionary algorithm using edge set tree representation to construct near-optimal data aggregation trees simultaneously, in which each gene represents an edge, each taking a value of 0 or 1. However, this algorithm has not thoroughly explored the characteristics of the problem, for example, the effect of the number of packets transmitted in the networks.

The contributions of the paper can be summarized as follows:

- We first present the problem of maximizing the lifetime of the wireless sensor network by minimizing the total transmission and reception energy consumption by all sensor nodes.
- We propose the heuristic algorithm called RRPT to reduce the redundant packets in the network. This algorithm is based on the observation that the fewer the number of packets are, the less energy consumption is. Based on the RRPT algorithm, we also propose an Evolutionary Algorithm with RRPT initialization for further improvement.
- We carry out extensive simulations to evaluate our approaches. The empirical results show that our approaches have greatly improved energy consumption.
- We apply statistical tests to check whether the proposed algorithm is better than the previous algorithms or not.

The remainder of this paper is organized as follows: Sect. 2 gives an overview of state-of-the-art data aggregation in wireless sensor networks. Section 3 presents the system model and defines the problem. In section 4, we highlight the implementation of the proposed algorithm. Section 5 presents the performance evaluation of our algorithms. Finally, the paper concludes with Sect. 6.

2 Related works

There are many related works on the problem of building a data aggregation tree for maximizing the network lifetime in WSNs. In this work, we classify the related works into three categories, namely cluster-based approach (or centralized approach) and in-network aggregation, and tree-based approach.

Among various data aggregation methods, the clustering approach is considered a promising choice. In the cluster-based data aggregation approach, sensor nodes are grouped into clusters. The clustering of sensor nodes can be achieved by either of the two methods: centralized or distributed. In this section, we brief the centralized-based approaches. With this approach, the base station has information on the entire network. In each cluster, the cluster head (CHs) node is responsible for collecting, processing, and aggregating sensed data from the member nodes and forwarding it via the base station. An outstanding advantage of this approach is that it reduces the amount of information performing data aggregation at the cluster top nodes before forwarding to the base station. Besides, the ability to reuse bandwidth, increase resource allocation, and improve capacity control are also benefits of adopting cluster-based data aggregation approaches. Recent studies (Abirami and Anandamurugan 2016; Lee et al. 2015; Mantri et al. 2015; Mosavvar and Ghaffari 2019; Rida et al. 2019) have applied the methods based on this approach to solve practical problems in wireless sensor networks with the above advantages. Mantri et al. (2015) proposed an algorithm called Bandwith Efficient Cluster-based Data Aggregation (BECDA) to provide an efficient data collection solution in wireless sensor networks. With the goals of increasing the use of available bandwidth, reducing communication costs, and minimizing energy consumption in the system, the BECDA algorithm is proposed to reduce the number of packet transmissions from nodes to the base station. This proposed algorithm shows significant improvement in terms of PDR and throughput when compared with previous solutions. In Abirami and Anandamurugan (2016), the authors presented a shuffled frog meta-heuristic algorithm for CHs selection. This algorithm chooses CHs based on the energy remaining in the nodes. The shuffling strategy of frogs is used to exchange information between local searches to attain global optimum. Mosavvar and Ghaffari (2019) proposed combining the firefly algorithm and LEACH in Singh et al. (2017) protocol for aggregating data in WSNs. According to the proposed method, sensor nodes are divided into clusters. In each cluster, nodes are periodically active and inactive. Nodes with more remaining energy will be selected as active nodes. The cluster is activated and deactivated based on their energy level and distance. In addition, Lee et al. (2015) proposed a two-layer cluster head selection based on distance. A CH is selected in the second layer without communicating with other nodes. As presented in Rida et al. (2019), a new data handling approach was proposed to reduce data transmission without data integrity loss.

In-network aggregation is the global process of gathering and routing information through a multi-hop network, processing data at intermediate nodes to reduce energy consumption, thereby prolonging network lifetime (Ennajari et al. 2016). In Zhang et al. (2018), a novel ring scheme is proposed for the problem, in which the original packets of N nodes are aggregated into M packets ($1 < M < N$). Based on in-network data aggregation, this scheme improves energy efficiency and reliability of transmission in the network. Furthermore, the authors also proposed a fuzzy logic system. During data transmission, the network is divided into rings, and data is gathered from the outer ring to the inner circle. The copies of the packets are used to increase transmission reliability and reduce packet loss. The proposed fuzzy logic system regulates the number of these copies. The data collection mechanism in the network is also used in Vehicular Ad-hoc Networks (VANETs). It is designed by Dietzel et al. (2016) to securely collect data from traffic devices and increase the system's scalability.

in addition to the above approaches, another practical approach is to aggregate data based on a tree structure, namely, a data aggregation tree. In the data aggregation tree, the sensor nodes are organized into a tree, where data gathering is performed at intermediate nodes along the tree. The leaf nodes send the collected information to the root node (sink node) through the intermediate nodes (parent node). Recently, related studies (Lin and Chen 2017; Kuo et al. 2015; Nguyen et al. 2016) have proposed methods based on this approach for data aggregation problems in WSNs in terms of lifetime maximization. In the study of Nguyen et al. (2016), the Maximum Lifetime Data Aggregation Tree Scheduling (MLDATS) problem is introduced and is justified as the NP-complete problem. The authors proposed a tree-based algorithm called the local-tree-reconstruction-based scheduling algorithm (LTRBSA) to solve the MLDATS problem. Moreover, in the context of maximizing WSN's lifetime, Lin and Chen (2017) proposed an approximation algorithm to solve a data aggregation tree's construction with lifetime inverse of the networks is guaranteed to be within its optimal bounds. In Kuo et al. (2015), Kuo et al. studied the problem of building a data aggregation tree where the total energy cost of data transmission in the WSN is minimized. Such issues are also explored and solved when there are relay nodes and poor link quality in the networks. The authors have proved that these problems are NP-complete and proposed $O(1)$-approximation algorithms to solve them. The simulations have shown the proposed algorithms to be effective regarding energy costs. However, in the case of a large problem size, e.g., a network of many sensor nodes, the proposed approximation algorithm does not give a good solution with a significant difference from the optimal solution.

In summary, as the related studies introduced, there are many approaches to building a data aggregation tree for maximizing the network lifetime in WSNs. However, in this paper, we focus on the model introduced in Kuo et al. (2015) and propose a meta-heuristic algorithm to solve the problem based on the tree-based approach. The proposed algorithm achieves significantly better results than the previous solid algorithms, including the 2-approximation algorithm (SPT) in Kuo et al. (2015) and the multi-factorial evolutionary algorithm (ESMFA) in Tam et al. (2020).

3 Problem statement

In this paper, we work on the problem of constructing a data aggregation tree that minimizes the total energy cost for data transmission in a wireless sensor network, which is a simple way to prolong the network lifetime. We consider this problem under the scenario where nodes have the same transmission range and can do in-network aggregation. The problem proposed by Kuo et al. (2015), termed MECAT, is an NP-completed problem. In this section, we first recall the network model and the assumption of the network and then state the problem as a minimal optimization problem. Eventually, we show a motivating example for properly understanding this problem.

3.1 Network model

The network is modeled as a connected, undirected graph $G = (V, E)$ where:

- $V = \{v_0, v_1, \dots, v_n\}$, is the set of nodes where v_0 is the sink and the other ones are sensor nodes.
- E is the set of edges. Two nodes v_i and v_j are connected if and only if the distance between them is less than or equal to the sum of their communication radius.
- Each node v_i has to send a report of size $s(v_i)$ to sink v_0 periodically in a multi-hop fashion based on a routing tree. A routing tree or a data aggregation tree constructed for a network $G = (V, E)$ is a directed tree $T = (V_T, E_T)$ with root v_0, where $V_T = V$ and directed edge $(u, v) \in E_T$ only if an undirected edge $(u, v) \in E$. A node u can send packets to a node v only if $(u, v) \in E_T$.
- Let T_x and R_x be the energy needed to transmit and receive a packet, respectively. As long as the packet size is sufficiently small, T_x and R_x are constants.
- While routing, a hop-by-hop aggregation is performed according to the aggregation ratio $q \in \mathbf{Z}^+$, which is the maximum size of reports that can be aggregated in a packet.
- In a routing tree T, let:
  - $child_T(v_i)$ be a set of v_i's children in T.
  - $des_T(u, v_i)$ be the size of reports to be sent by u which is the child of v_i in T.
  - $des_T(v_i)$ be the total size of reports to be sent by v_i's descendants in T, $des_T(v_i) = \sum_{u \in child_T(v_i)} des_T(u, v_i)$.
  - $c_T(v_i)$ be the energy consumption of v_i. This one accounts for the energy cost of the radio, which is the sum of transmitting and receiving energy in the communication process.

$$c_T(v_i) = \left\lceil \frac{des_T(v_i) + s(v_i)}{q} \right\rceil \times T_x + \sum_{u \in child_T(v_i)} \left(\left\lceil \frac{des_T(u, v_i)}{q} \times R_x \right\rceil \right).$$

3.2 Problem formulation

Based on the network model, we can now formulate the problem as follows:

Input:

- A WSN is modeled as a connected and undirected graph $G = (V, E)$, the sink node is v_0.
- r is the transmission range of a node.
- q is the aggregation ratio.

Output: A data aggregation tree $T = (V_T, E_T)$ rooted at v_0 $(V_T = V)$.

Objective: Minimize the total transmission and reception energy consumption by all nodes in the network

$$\sum_{v_i \in V} c_T(v_i) \rightarrow min. \quad (1)$$

Since the number of packets sent and received by nodes in T are equal, the Eq. 1 becomes:

$$\sum_{v_i \in V} (T_x + R_x) \times \left\lceil \frac{des_T(v_i) + s(v_i)}{q} \right\rceil \rightarrow min. \quad (2)$$

In this formulation, $\left\lceil \frac{des_T(v_i)+s(v_i)}{q} \right\rceil$ is the number of packets sent from v_i to its parent on T.

Example 1 To illustrate the problem more clearly, we present a tiny example in this section. Consider a simple wireless sensor network described as a graph in Fig. 1a. We assume the aggregation ratio q is 3, and both T_x and R_x equal 1. Sensor nodes $(v_1, v_2, v_3, v_4, v_5)$ have sizes of reports (3, 4, 9, 6, 8). Figure 1b and c are two routing trees constructed for this graph. In Fig. 1b, the number of packets sent by $(v_1, v_2, v_3, v_4, v_5)$ is (5, 4, 6, 2, 3), respectively. In Fig. 1c, those are (1, 6, 3, 2, 3). By using Eq. 2, the total energy consumption of two trees in Fig. 1b and c are 40 and 30, respectively. Therefore, the routing tree in Fig. 1c gives a better result.

4 Proposed method

The MECAT problem has been proven to be an NP-hard problem in Kuo et al. (2015). The authors proposed a 2-approximation algorithm to tackle the MECAT problem. They found that a shortest-path tree turned out to be a $2-$approximation algorithm and could be easily implemented in a distributed manner. In their solution, a sensor node is connected to the sink node when they have the least number of connected edges. This strategy may be ineffective if a sensor node can connect to a large number of other sensors in the network, and it also omits other criteria for energy consumption, such as the number of packets, etc.

Hence, in this section, a new approach based on genetic algorithms called PGA is proposed to address the MECAT problem. We also come up with a heuristic algorithm and apply it to the initializing population of PGA to illustrate the effectiveness of the mentioned ones. Details of the heuristic algorithm and PGA are presented below.

4.1 Proposed heuristic algorithm

The objective function of the MECAT problem is to minimize the total energy cost for transmitting and receiving packets of all nodes in a wireless sensor network. We have observed that a larger number of packets will give rise to a higher amount of energy consumption in the network. In contrast, a smaller number of packets will lead to a lower amount of energy consumed. Therefore, the total energy cost is affected by the number of packets circulating in the network. Reducing the number of packets will reduce the total energy consumption. One way to reduce it in the network is

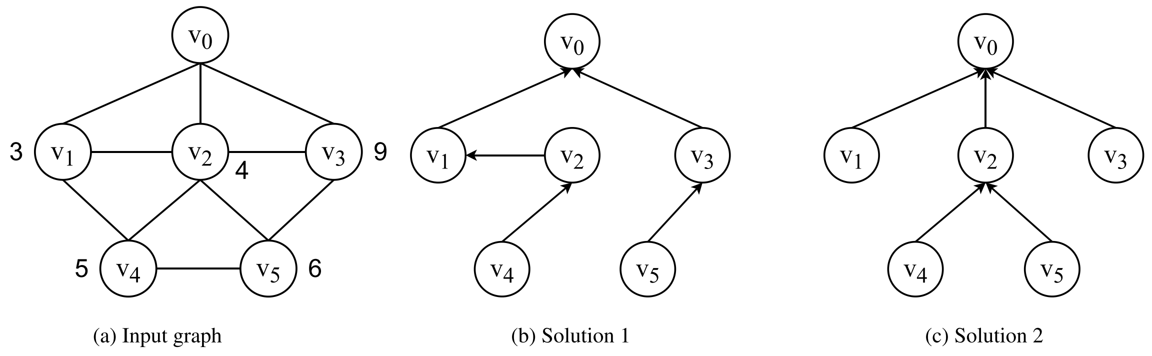


Fig. 1 An illustration of the problem with two routing trees constructed for a given network

to reduce the number of "redundant packets". The concept and an example of a redundant packet are presented in Definition 1 and Fig. 2, respectively. Inspired by our observation, a new heuristic algorithm, namely RRPT, is proposed to construct an aggregation tree with the least number of redundant packets.

Definition 1 Given an aggregation tree $T = (V_T, E_T)$ with root v_0 as a sink, nodes $v_i, v_j \in V_T \setminus \{v_0\}$. A ***redundant packet*** is a packet sent from node v_i to node v_j whose size is less than the aggregation ratio q.

The main idea of the RRPT algorithm is to build an aggregated tree with as few redundant packets as possible. The algorithm is implemented based on a parent node selection strategy for a new node added to the data aggregate tree. The criteria for selecting the parent node of a node being considered is the minimum number of redundant packets on the path from that parent node to the sink. Details and an example of the RRPT algorithm are presented in Algorithm 1 and Fig. 3.

Algorithm 1: The general pseudo-code for our heuristic RRPT.

```
Input  : A given graph $G = (V, E)$. The sink node $s$ with $depth = 0$.
Output: A tree $T = (V, E_T)$ rooted at $s$ with minimum number of redundant packets.
begin
    foreach $u \in V$ do
        $u.depth \leftarrow \infty$;
        Mark u as visited;
        $P(u) \leftarrow \emptyset$;            // $P(u)$ is the possible parents set of node $u$
    end
    Initialize a queue $Q = \{s\}$, which stores some visiting nodes;
    while $Q \neq \emptyset$ do
        $u \leftarrow$ Pop next visiting node from $Q$ and mark $u$ as visited;
        foreach edge $(u, v)$ do
            if ($v$ is marked not visited and ($u.depth + 1 \leq v.depth$)) then
                if ($u.depth + 1 < v.depth$) then
                    $v.depth \leftarrow (u.depth + 1)$;
                    $P(v) \leftarrow \{u\}$;
                else
                    $P(v) \leftarrow P(v) \cup \{u\}$;
                end
                if $v \notin Q$ then
                    Add $v$ to $Q$;
                end
            end
        end
        if $u \neq s$ then
            foreach $v \in P(u)$ do
                Calculate the number of redundant packets on the path from $v$ to $s$; // See Algorithm
                    2
            end
            $Parent(u) \leftarrow$ Get the node from $P(u)$ which has the minimum number of redundant packets.
              If there is more than one node, randomly pick one of them;
            Update $u$'s ancestors' aggregation data size;
        end
    end
end
```

Algorithm 2: Calculating the number of redundant packets on a path from a node to the sink.

```
Input  : A node $u$ would be added to an aggregation tree
         A parent node $v$ of node $u$
         The sink node $s$, the aggregation ratio $q$
Output: The number of redundant packets on the path from $v$ to $s$
begin
    $w \leftarrow v$;
    $number \leftarrow 0$;
    while $i \neq s$ do
        if (Total aggregation data size of i and u) mod $q \neq 0$ then
            $number \leftarrow number + 1$;
        end
        $w \leftarrow Parent(w)$;
    end
    return $number$;
end
```

Example 2 Fig. 3 illustrates an example of selecting a parent node of a node in RRPT. Figure 3a depicts an aggregation tree T with 9 nodes and node v_9 being considered to be added to the tree. The size of the sensor node's own report is shown in parentheses. Figure 3b depicts the tree when appending node v_9 to its parent nodes. The values in parentheses are the size of the sensor node's aggregated reports and the number of redundant packets, respectively. Assuming that node v_6 is selected as the parent node, node v_6 can aggregate its own report and the received report from node v_9 (total size of reports is $3 + 2 = 5$) into $\lceil \frac{5}{3} \rceil = 2$ packets (including 1 redundant packet). Repeating the process with other nodes on the path from node v_6 to the root v_0, we calculate the number of redundant packets on this path as 3. Using a similar calculation, the number of redundant packets in cases nodes v_7, and v_8 are selected as the parent nodes are 1 and 2, respectively. Comparing packet numbers, node v_7 is chosen as the parent of node v_9 because the path from node v_9 to the root node passing through node v_7 has the smallest number of redundant packets. After inserting node v_9, the obtained tree is shown in Fig. 3c.

4.2 Proposed genetic algorithm

Genetic Algorithm (GA) has been proven effective on NP-hard problems (Holland 1992; Mirjalili 2019; Binh et al. 2021; Hien et al. 2022). In this section, we propose a new approach based on GA, namely PGA, to solve the MECAT and describe how the main steps of the PGA are implemented in the proposed algorithm.

4.2.1 Individual representation

An essential step in the design of a GA is to find an appropriate individual representation. In the MECAT, several methods are suitable for representing the solution, such as Cayley representation, network random keys (Netkeys) representation, edge representation, etc. However, with our problem approach, the Netkeys representation method is used to represent the individual.

For the MECAT problem, consider a given graph $G = (V, E)$ with $|V| = n$ and $|E| = m$. A possible solution of the problem is an aggregation tree $T = (V_T, E_T)$ ($V_T = V$). Based on Netkeys representation, a solution to the MECAT problem is represented by a real number vector W with dimension m. Each element in W is linked to an edge in E whose value is a real number in the range [0, 1]. For instance, a complete representation of a possible solution in an input graph in Fig. 4 with 5 nodes is illustrated in Fig. 5.

With the above-mentioned encoding mechanism, a corresponding decoding method is also proposed. The proposed method includes four steps to generate the spanning tree $T = (V, E_T)$ of graph G from the representation Netkeys W. The detailed steps are given as follows:

Step 1: Initially initialize T with $E_T = \emptyset$.

Step 2: Traverse each element in W in descending order of value.

Step 3: With the *ith* element traversed, the link edge is checked. If the insertion of the link edge in E_T would not create a cycle, then insert the link edge in T.

Step 4: Check the number of edges in E_T. If $|E_T| = n - 1$, then stop the process, else, continue with **Step 2**.

With this decoding from every possible Netkeys vector, the spanning tree constructed is unique and valid.

An example of the decoding method is shown in Fig. 6. For the given representation in Fig. 5, the traversal order of elements in descending order will be 10, 8, 6, 9, 2, 7, 1, 5, 4, 3. Firstly, the link edge (v_3, v_4) (position 10) is inserted into the spanning tree. Next to that is the link edges (v_2, v_3) (position 8) and (v_1, v_3) (position 6). If we added the link edge (v_2, v_4) (position 9), the cycle (v_2, v_4, v_3, v_2) would be created, thus the link edge (v_2, v_4) is not added to the spanning tree. Next, edge (v_0, v_2) (position 2) is inserted into the spanning tree. The number of edges of the spanning has reached a maximum of 4. Therefore, the decoding algorithm is terminated, and we have a constructed tree as shown in Fig. 6.

4.2.2 Initialization

In this paper, we use a combination of two initialization methods: random initialization and heuristic initialization to generate initial population P_0 with N Netkeys vectors of m-dimension. Herein, we use the proposed RRPT algorithm for heuristic initialization. The initialization procedure, including three steps, is described as follows:

Step 1: The population P_0 includes the individuals generated with random initialization and heuristic initialization.

Step 2: Eliminate the identical individuals in P to obtain a population of distinct individuals.

Step 3: Generate individuals by using the Local search algorithm that is described in Algorithm 4, then add them to P_0 until the population size is reached.

4.2.3 Fitness function

In the genetic algorithm, the fitness function, by definition, is a process for scoring each chromosome based on its qualification. In the MECAT problem, the fitness function value is the total energy consumed by the nodes in the data aggregation tree.

After obtaining the spanning tree from the decoding individual, to calculate the value of the fitness function, it is necessary to create a tree to synthesize the data from that spanning tree and calculate the value of the target function according to the total energy consumption formula of the node on the data aggregation tree. The construction of a tree that aggregates data from the spanning tree is described in detail in Algorithm 3.

Algorithm 3: Data aggregation tree construction

```
Input: A spanning tree $T$ in graph $G$ and a sink node $s$
Output: A data aggregation tree $T'$
1  begin
2      Initialize empty stack $S$;
3      Mark the vertices of $G$ as unvisited;
4      Add sink node $s$ to $S$;
5      while $S$ is not empty do
6          $u \leftarrow$ Get information of elements on top of stack $S$;
7          for $v \leftarrow$ a child node of $u$ do
8              if $v$ has not been visited then
9                  Add $v$ to the stack $S$;
10             end
11         end
12         if all children of $u$ have been visited or $u$ is a leaf node then
13             Removes $u$ from the stack $S$;
14             Mark $u$ as visited;
15             Update aggregate information of parent node of node $u$;
16         end
17     end
18 end
```

4.2.4 Crossover operator

This paper describes a crossover operator based on the Simulated Binary Crossover (SBX) Deb et al. (1995). SBX is a pseudo-hybrid that repeats the principle of single-point crossover in binary representation. It has some essential properties, such as it does not change the genotype value on average, and the offspring's genotype is inherited close to their parents. Thanks to these properties, EA can maintain the basic genetic blocks (original term: building block) well over generations during evolution.

```
Algorithm 4: Local search
Input: An individual x
Output: A new individual y
begin
    Spanning tree $T$ ← Decode the individual $x$;
    repeat
        Node $u$ ← Select randomly a node in the spanning tree $T$;
        Node $v$ ← a parent node of the node u;
        Remove the link edge $(u,v)$ to obtain 2 connected components S and A;
        $CS$ ← the cut-set of a cut $(S,A)$;
        Edge $e$ ← Select randomly in the set CS;
    until $e \neq (u,v)$;;
    Spanning tree $T'$ ← Connect two connected components according the link edge $e$;
    Individual $y$ ← Encode the spanning tree $T'$;
end
```

4.2.5 Mutation operator

In this paper, we use a new mutation operator based on Local search to introduce new genetic material into an existing individual in order to add diversity to the genetic characteristics of the population. Details of the Local search algorithm are described in Algorithm 4. Figure 7 illustrates an example of the Local search algorithm.

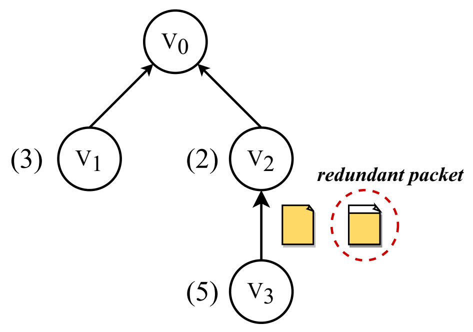


Fig. 2 An example of a redundant packet in an aggregation tree. It is assumed that an aggregation tree T with the sink node v_0, three nodes v_1, v_2, v_3, and the aggregation ratio q is 3. The size of the sensor node's own report is shown in parentheses. For node v_3, its own report of size 5 is aggregated into $\lceil \frac{5}{3} \rceil = 2$ packets, one of maximum size 3, and one of smaller size 2. The packet of size 2 is a redundant packet, which can aggregate more data

4.2.6 Selection method

The **GA**'s selection consists of two main types: select to breed and select to build populations for the next generations. The details of the selection methods are shown as follows:

- Selection for reproduction: In this proposal, tournament selection is used. Specifically, two individuals are randomly selected to take part in a tournament, i.e. the individual with the best fitness is selected. For crossover, two tournaments are held to select two parents. The advantage of this selection is that the worst individuals in the population will not be selected, and the best individual will not dominate in the reproduction process.
- Selection for building populations of the next generation: The proposed algorithm uses elitism selection to select the best individuals from the current population to survive the next generation. This strategy will ensure that the maximum fitness value does not decrease.

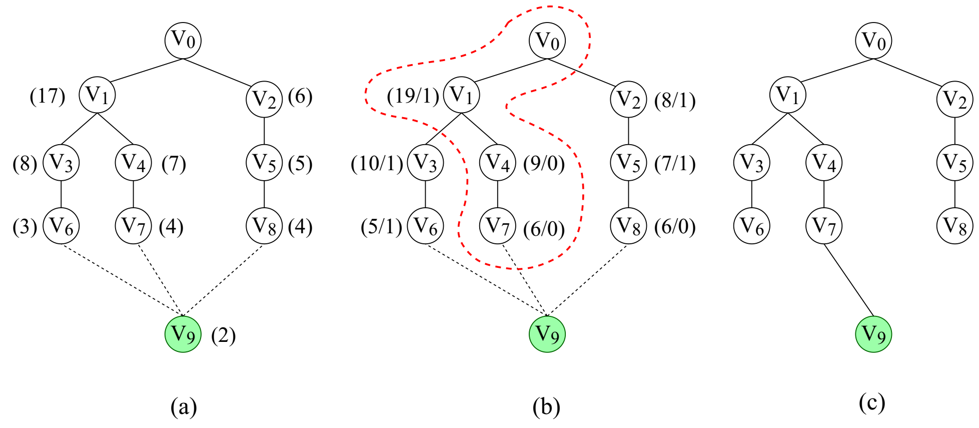


Fig. 3 An illustration of a strategy for selecting a node's parent node in the RRPT algorithm

5 Experiment results

5.1 Experimental settings

Two datasets are created for our experiments based on Tam et al. (2020), namely Type 1 ($200m \times 200m$) and Type 2 ($100m \times 100m$). For each dataset, 30 instances are generated. Type 1's instances have formatted as l_n_r, while Type 2's instances are m_n_r. In which, n is the number of sensor nodes and r is the transmission range. Parameters for the problem are shown in Table 1.

We evaluate the performance of our proposed algorithm by comparing PGA to the strong baselines, including the previous state-of-the-art algorithm ESMFA and the 2-approximation algorithm SPT. For a fair comparison, we set up two algorithms PGA and ESMFA with the same population size, the same number of generations, and the same number of evaluation calls. All results of PGA and ESMFA are averaged and reported on 30 independent runs with different random seeds.

The details of configurations for different algorithms are summarized in Table 2.

All experiments are implemented in Java program language and executed on the same machine running Ubuntu Linux 16.04 with Intel(R) Core(TM) i7-4800MQ CPU @ 2.70GHz. The programs, data, and resources are made available at https://github.com/DaoTranbk/PGA4Mecat.git.

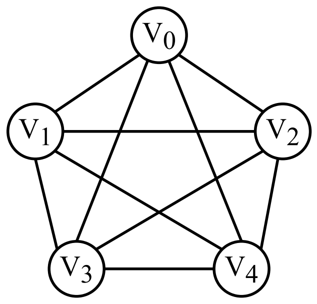
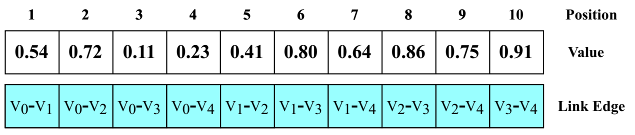


Fig. 4 A graph G with five vertices and ten edges



1	2	3	4	5	6	7	8	9	10	Position
0.54	0.72	0.11	0.23	0.41	0.80	0.64	0.86	0.75	0.91	Value
V0-V1	V0-V2	V0-V3	V0-V4	V1-V2	V1-V3	V1-V4	V2-V3	V2-V4	V3-V4	Link Edge

Fig. 5 A representation for a possible tree in graph G

Table 1 Parameters of the problem

Parameter	Description	Value
n	Number of sensor nodes	100, 110, 120,⋯, 190
r	Transmission range	10, 15, 20
$s(v_i)$	Non-uniform report size	1, 2, 3, 4, 5
T_x	Transmission energy constant	2
R_x	Receiver energy constant	1
q	Aggregation ration	4

Table 2 Parameters in PGA and ESMFA

Parameter	ESMFA	PGA
Population size	100	100
Maximum generations	1000	1000
Number of evaluation calls	100,000	100,000
rmp	0.3	–
Mutation rate	–	0.2
Crossover rate	–	0.7

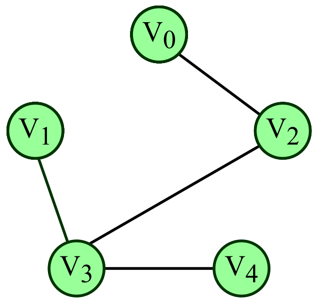


Fig. 6 An example of decoding the representation in Fig. 5

Table 3 Three versions of PGA with different rates of individuals created by heuristic initialization and random initialization

Initialization	PGA versions		
	H-PGA	R-PGA	M-PGA
Heuristic	100%	0%	60%
Random	0%	100%	40%

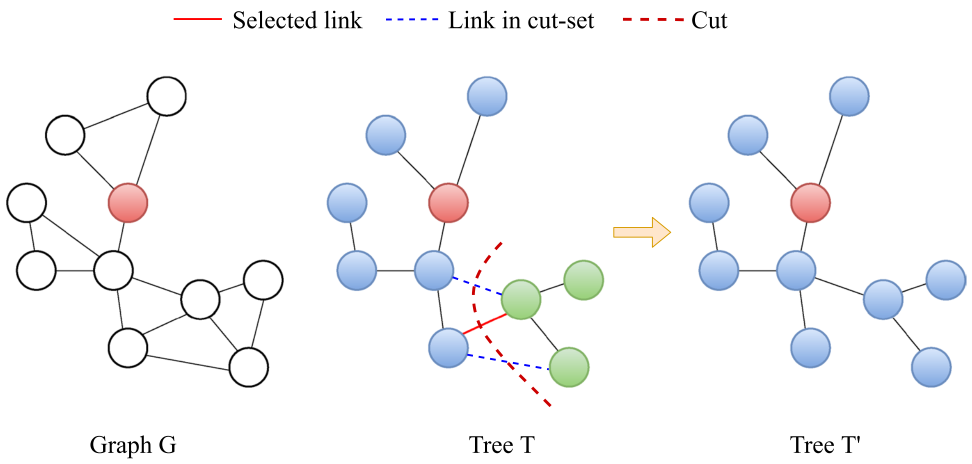
5.2 Experimental criteria

We evaluate the performance of our proposed algorithms on several criteria: average (Avg) and standard deviation (Std) of objective value, running time (Time), and convergence trend. In addition, we use the relative percentage difference (Binh et al. 2019; Dao et al. 2021) to analyze the improvements in the performance of the proposed algorithms. Namely, the relative percentage difference between algorithm A and algorithm B, *RPD (A, B)*, is the difference between the average objective values obtained by two algorithms *A* and algorithm *B*. In the MECAT problem, which is a minimum optimization problem, the formula of the RPD metric is as follows:

$$RPD(A,B) = \frac{C_B - C_A}{C_B} \times 100\%.$$

where C_A and C_B are the average cost of solutions obtained by the algorithm A and algorithm B, respectively.

Fig. 7 An example of the Local search algorithm



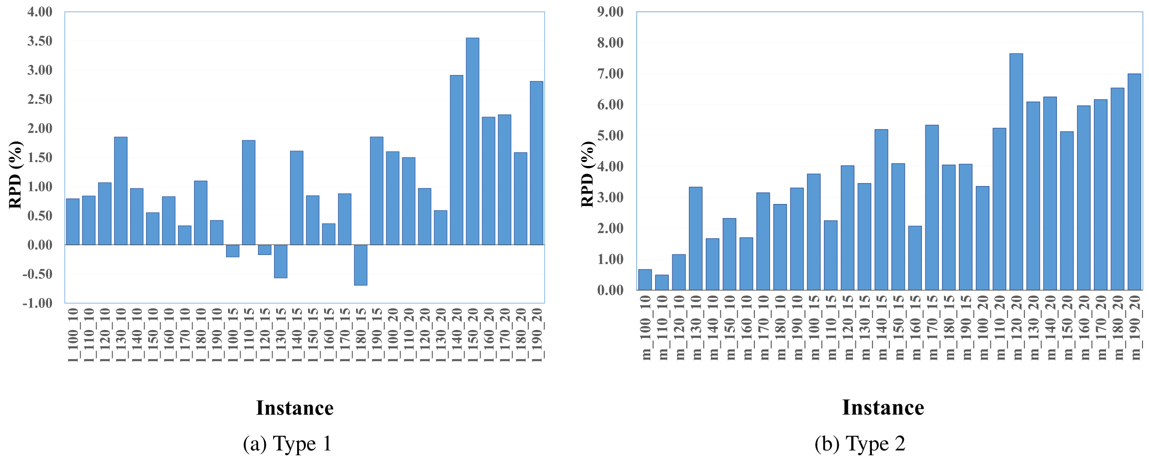


Fig. 8 RPD value calculated between RRPT and SPT on two types

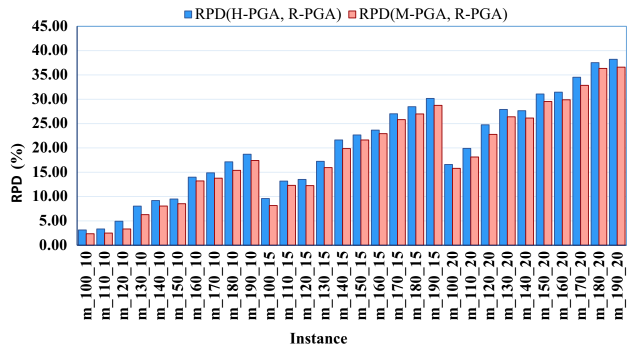


Fig. 9 Comparison of two versions H-PGA and M-PGA to R-PGA in terms of RPD value (higher is better)

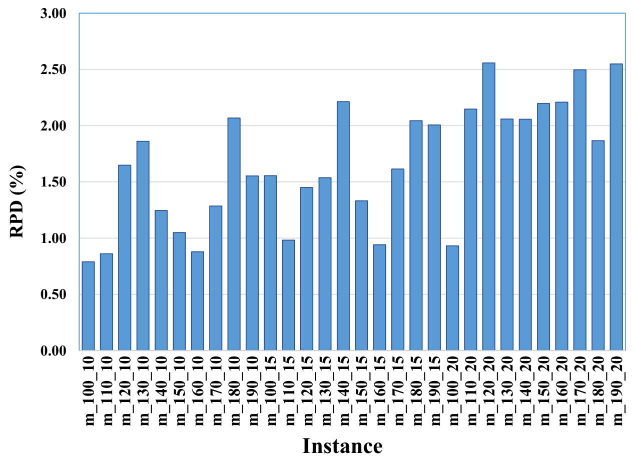


Fig. 10 Comparison between H-PGA and M-PGA in terms of RPD

Table 4 Average objective value, standard deviation, and execution time obtained by the H-PGA and ESMFA on the large dataset

Instances	H-PGA			ESMFA		
	Avg ↓	*Std* ↓	*Time(s)* ↓	*Avg* ↓	*Std* ↓	*Time(s)* ↓
l_100_10	**1092.00**	**0.26**	**5.74**	1101.40	0.90	252.91
l_100_15	**997.10**	**0.32**	**5.20**	997.40	0.52	252.91
l_100_20	**647.70**	**0.37**	**6.41**	654.50	0.68	251.20
l_110_10	**942.50**	**0.25**	**6.71**	950.50	0.63	271.42
l_110_15	**907.00**	**0.35**	**6.80**	915.00	0.44	271.48
l_110_20	**793.10**	**0.39**	**8.33**	802.30	0.66	274.02
l_120_10	**1277.80**	**0.22**	**6.62**	1286.90	0.62	288.91
l_120_15	**920.40**	**0.41**	**7.07**	927.00	0.76	293.86
l_120_20	**812.90**	**0.49**	**9.25**	819.50	0.52	296.73
l_130_10	**1021.00**	**0.24**	**8.55**	1037.50	1.03	315.81
l_130_15	**1122.50**	**0.57**	**7.92**	1127.30	0.80	317.84
l_130_20	**981.30**	**0.40**	**10.58**	990.90	0.54	325.23
l_140_10	**2069.90**	**0.13**	**7.50**	2076.80	0.25	335.37
l_140_15	**1292.00**	**0.22**	**8.93**	1314.00	0.61	337.56
l_140_20	**910.10**	**0.44**	**10.67**	929.10	0.78	339.18
l_150_10	**1573.70**	**0.21**	**9.43**	1590.90	0.64	252.91
l_150_15	**1375.60**	**0.37**	**9.43**	1382.30	0.48	252.91
l_150_20	**1040.00**	**0.32**	**12.87**	1055.90	0.46	251.20
l_160_10	**1510.00**	**0.22**	**10.50**	1523.00	0.59	271.42
l_160_15	**1384.40**	**0.34**	**11.61**	1408.60	0.71	271.48
l_160_20	**1099.70**	**0.37**	**15.73**	1132.30	0.67	274.02
l_170_10	**1918.30**	**0.21**	**10.81**	1941.00	0.55	288.91
l_170_15	**1346.00**	**0.34**	**12.99**	1367.60	0.64	293.86
l_170_20	**1116.20**	**0.36**	**16.80**	1248.10	0.59	296.73
l_180_10	**1477.60**	**0.22**	**13.03**	1502.20	0.55	315.81
l_180_15	**1507.30**	**0.37**	**14.23**	1525.30	0.65	317.84
l_180_20	**1229.60**	**0.38**	**19.24**	1259.40	0.58	325.23
l_190_10	**2483.10**	**0.16**	**12.29**	2512.10	0.51	335.37
l_190_15	**1754.60**	**0.25**	**15.32**	1790.10	0.61	337.56
l_190_20	**1238.10**	**0.36**	**19.34**	1259.40	0.58	339.18

The best results are in bold

5.3 Performance analysis for the proposed heuristic RRPT

In this section, we evaluate the performance of our heuristic algorithm RRPT by comparing RPD values between RRPT and the 2-approximation SPT. Figure 8 presents the obtained results by RRPT and SPT over all instances of each type in terms of RPD value. It can be seen that on Type 1, RRPT produces better results than SPT over 26/30 in total instances, however, SPT performs better than RRPT on 4/30 instances by small margins (from about 0.15% to 0.7%). In the meantime, RRPT outperforms SPT on all instances of Type 2. The largest improvement in objective values obtained by RRPT is about 7.7% on the instance *m_120_20*.

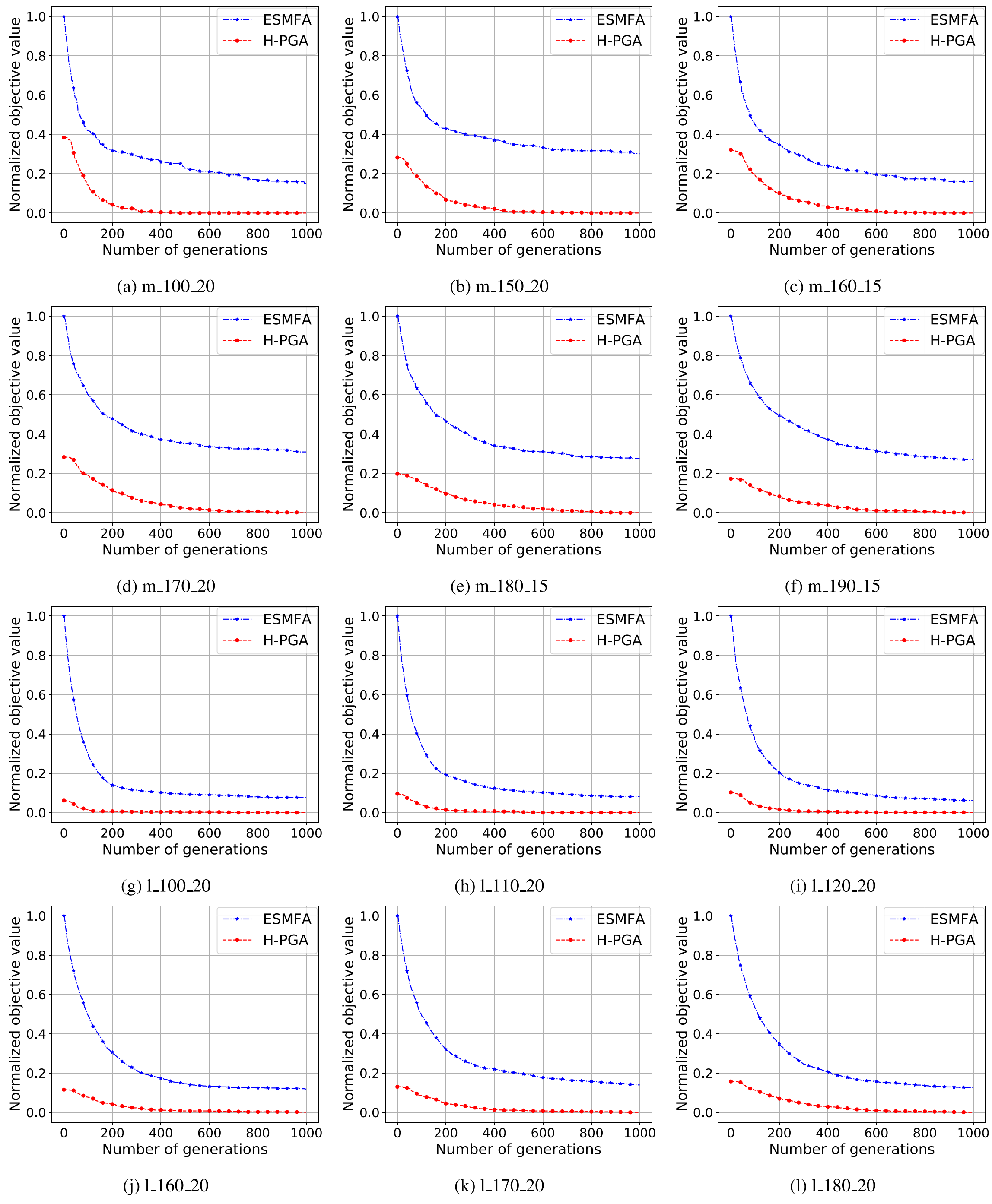


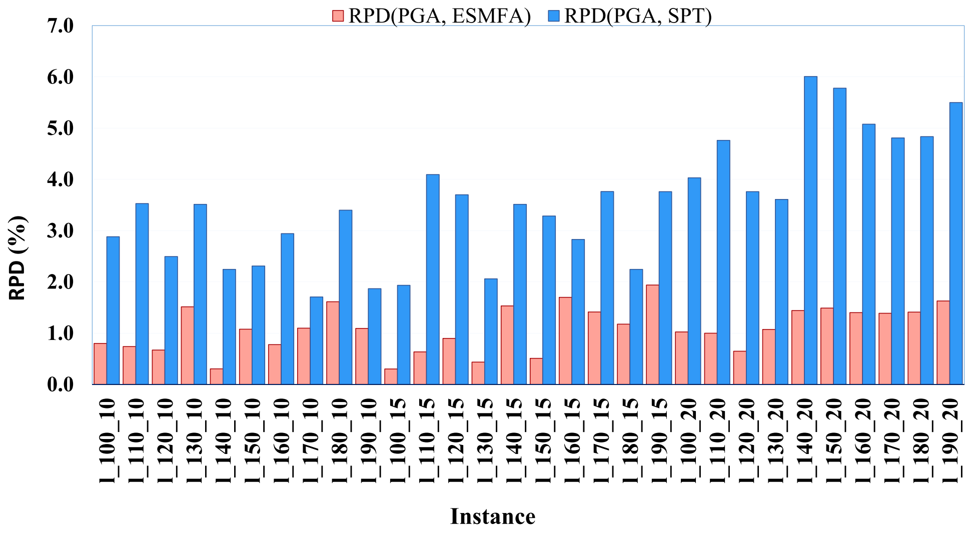
Fig. 11 The convergence trend of instances in the benchmark datasets

The average RPD values on Type 1 and Type 2 are 1.38% and 3.83%, respectively.

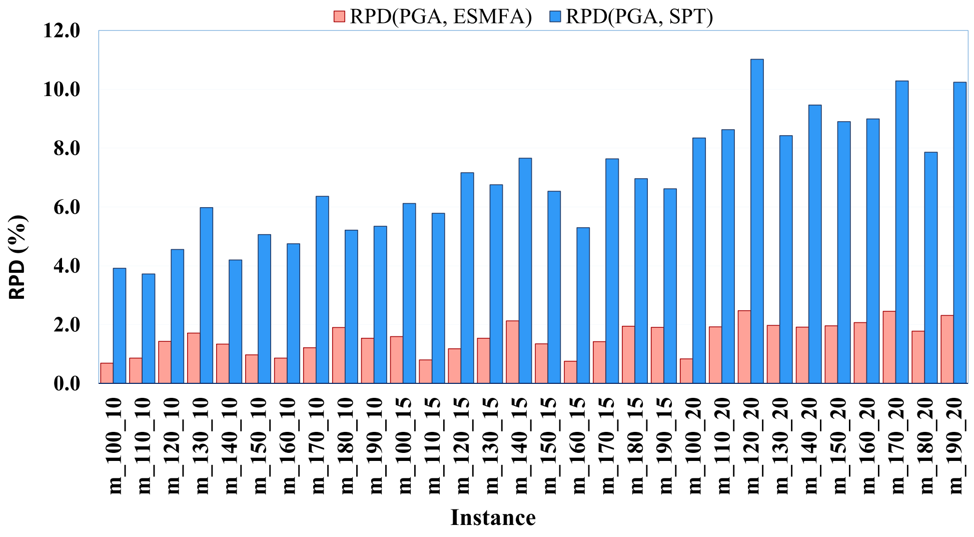
In addition, on Type 2, we see a growing trend of RPD values when the number of sensors increases and find that the RPD values are larger than the ones on Type 1. It is worth noting that the number of edges of input graphs in Type 2 is larger than in Type 1. As a result, RRPT is more effective than the SPT, especially on dense graphs with a large number of edges and a number of nodes.

5.4 Performance analysis for PGA with different initialization

To deeply investigate the performance of PGA, we implement PGA with three versions: H-PGA, M-PGA, and R-PGA, in which each version adopts a different initialization scheme with a different rate of individuals created by heuristic initialization using RRPT and random initialization as shown in Table 3. In M-PGA, the rate of heuristic initialization using RRPT is chosen to be 60% because our experiment results show that this rate helps M-PGA produce the best results. To evaluate the performance of the three versions, we use all instances on the Type 2 dataset.



(a) Type 1.



(b) Type 2.

Fig. 12 RPD values between PGA, ESMFA and SPT on two types data

Figure 9 shows that H-PGA and M-PGA with heuristic initialization outperform R-PGA with a random one. In particular, M-PGA, which is 60% individuals created by RRPT, achieves a better result than R-PGA, approximately 18.67% in each Type 2's instance. Meanwhile, H-PGA, which used RRPT to create all individuals in the population, gets a better result than R-PGA overall Type 2's instances, approximately 19.99% in each instance.

The comparison results between M-PGA and H-PGA are shown in Fig. 10. As a result, the RPD values between H-PGA and M-PGA in all instances are positive, which means H-PGA outperforms M-PGA in general. In each instance, H-PGA gets average better results than M-PGA, approximately 1.64%. Therefore, H-PGA is the best version of PGA and is then used to compare the baselines in the following experiments.

5.5 Comprehensive comparison of PGA to the baselines

Here, we report the best results obtained by PGA with the best version H-PGA that uses a 60% heuristic and 40% random in the initialization method.

5.5.1 Objective value and running time

Table 4 shows the results of the H-PGA consisting of the average objective value (Avg), standard deviation (Std), and execution time over 30 independent runs on the large dataset, i,e. Type 2. It can be observed that the proposed algorithm PGA achieves better average objective values and smaller standard deviations than ESMFA in all instances. Especially, our PGA runs far faster than ESMFA but still outperforms the baseline, i.e. on the instance *l_100_10*, PGA achieves better objective value while the running time of PGA is only about 1/44 of ESMFA (PGA: 5.74 s and ESMFA: 4.2 min). These obtained results show that the PGA performs better results, higher stability, and faster execution time than the baseline ESMFA.

5.5.2 Convergence trend

To assess the efficiency of our proposed algorithms, the convergence trend graphs are also investigated. The functions in Gupta et al. (2015) were used for computing normalized averaged objective values for comparing convergence trends between PGA and ESMFA. The convergence trend graphs for several instances of Type 1 and Type 2 are depicted in Fig. 11. In the figure, the y-axis gives the normalized averaged objective value over 30 independent runs, while the x-axis denotes the number of generations made so far.

As can be observed in the convergence trend graphs in Fig. 11a–i, the proposed algorithm obtained good results and fast convergence speed in the initial generations. We observe that our PGA converges much faster than ESMFA in all the instances. Because of the good performance of RRPT algorithm in the initialization method, our PGA can obtain good individuals at the first generation, thus leading to enhance convergence speed of PGA as observed in Fig. 11. The convergence trend graphs also show that the heuristic initialization RRPT method is significantly better than Kruskal random initialization used in ESMFA (Tam et al. 2020).

5.5.3 Relative percentage improvement

Figure 12 presents the RPD values of two pairs: (PGA, ESMFA) and (PGA, SPT) on Type 1 and Type 2. According to the results, it is clear that PGA outperforms ESMFA and SPT on all instances of each dataset type. Particularly, the value of RPD(PGA, ESMFA) on Type 1 has changed from 0.03% to 1.98%. On Type 2, the biggest RPD value is 2.56% and the smallest one is 0.79%. When compared to SPT, the RPD(PGA, SPT) fell into a range from 1.67% to 6.08% on Type 1 and from 3.72% to 11.09% on Type 2. As a result, the improvement of the PGA's result compared to SPT is more significant than the one compared to ESMFA.

Based on the obtained results, we see that the improvement of the PGA's result compared to the remaining algorithms increases on dense graphs. Specifically, in each type, with the higher transmission range, which is equivalent to a

Table 5 Ranks of algorithms achieved by Friedman test

Type	Agorithms		
	H-PGA	ESMFA	SPT
Mean rank	1.00	2.00	3.00

Table 6 Test statistics of algorithms achieved by Friedman test

Test statistics			
N	*Chi-square*	*df*	*p*
60	120.00	2	0.00

Table 7 Test statistics using post-hoc test with Wilcoxon signed-rank test

Values	Pair algorithms		
	PGA-ESMFA	PGA-SPT	ESMFA-SPT
Z	−6.736[a]	−6.736[a]	−6.736[a]
p	0.000	0.000	0.000

*Note: a. Based on positive ranks

higher number of links in the network, the RPD values of PGA compared to the baselines were increased. Furthermore, the PGA effect compared to ESMFA and SPT is more precise when decreasing the deployment area's size from 200×200 (Type 2) to 100×100 (Type 1) which means that the average distance between nodes is also decreased in order to increase their connectivity. The average value of RPD(PGA, ESMFA) in Type 1 and Type 2 is 1.13% and 1.67%, respectively. Meanwhile, the average value of RPD(PGA, SPT) is 3.58% in Type 1 and 7.03% in Type 2.

5.6 Statistical tests

In this section, we examine several well-known non-parametric statistical tests Zar (1999); Sheskin (2020) to analyze obtained results by our PGA and the baselines on two types of datasets and to decide whether the differences between them are significant or not. The analysis consists of two steps:

- The first step is using the Friedman test to examine the significant differences between the results obtained by the algorithms.
- The second step is using the post-hoc test to compare each pair of algorithms in detail after examining a significant difference between the algorithms.

The null hypothesis for the Friedman test is that there are no differences between the results obtained by the algorithms. The Friedman test compares the mean ranks between the algorithms' results and indicates how the results differed. Table 5 shows the mean rank for each of the algorithms. In the ranks table, the proposed algorithm, H-PGA, gets the best rank on all instances of two types of datasets.

Table 6 provides the test statistic χ^2 value ("Chi-square"), degrees of freedom ("df") and the calculated probability value ("p"). As can be observed in Table 6, the results show that the probability value ($p = 0.000$) is less than the selected significance level ($\alpha = 0.01$), thus the null-hypothesis is rejected. It also can be concluded that at least two algorithms' results are significantly different from each other.

To examine where the differences actually occur, we use Wilcoxon signed-rank test as a post-hoc test to compare each pair of results obtained by the algorithms: PGA to ESMFA, PGA to SPT, and ESMFA to SPT. In this experiment, a Bonferroni correction is applied because of multiple comparisons. A new significance level of $0.01/3 = 0.0034$ is calculated. This means that if the *p-value* is larger than 0.0034, we do not have a statistically significant result. Table 7 shows the Wilcoxon signed-rank test's output on each of the algorithms' pair results. It can be observed in Table 7 that all the p values are less than the significance level of 0.0034, as a consequence, all pairs of results obtained by the algorithms ($p = 0.00$) were statistically significantly different. Therefore, it can be concluded that there is an obvious difference between PGA's solution quality and the other algorithms in terms of statistical testing, thus confirming that PGA performance is significantly better than the previous solid baselines with 99% reliability.

5.7 Discussions

According to the performance analysis of the proposed heuristic RRPT, we find that RRPT performs better than the 2-approximation algorithm SPT in most instances. Notably, the obtained results by RRPT are significantly better than SPT on all instances of large datasets, thus indicating our heuristic is more scalable for large datasets. One predictable reason for the better performance of RRPT than SPT is that RRPT aims to build an aggregation tree with as few redundant packets as possible to reduce the number of packets transmitting in the network. Remarkably, our heuristic utilizes a parent node selection strategy for a new node added to the aggregation tree with the minimum number of redundant packets on the path from its parent to the sink, while SPT uses a random selection.

Comparing our PGA to the previous competitive baselines, it is clear that our proposed algorithm steadily outperforms its baselines and runs far faster than ESMFA. The main reason is PGA adopts efficient genetic operators such as crossover based on SBX, which performs directly on genotypes. At the same time, ESMFA uses genetic operators on tree structures, which takes a long time to convert genotypes to corresponding phenotypes in tree structures. Moreover, utilizing RRPT as heuristic initialization helps the search process of PGA finds good solutions quickly.

Although this paper has demonstrated that PGA performs well for MECAT and RRPT can help PGA produce significantly better results than strong baselines, there are still rooms to improve the performance of our algorithms. One promising direction is using other efficient encodings for MECAT as our encoding based on Netkeys still has a weakness of redundancy property for tree representations. Future work will investigate which encoding is more suitable for MECAT by evaluating the performance of each encoding nested in PGA.

6 Conclusion

In this paper, we have presented the MECAT problem of maximizing the lifetime of Wireless Sensor Networks by constructing an energy-efficient data aggregation tree. Based on the observation of the problem's characteristics, we have proposed the heuristic, namely RRPT, to find good solutions for MECAT, which can reduce the redundancy of packages when transmitting data. Notably, the experiment results show that RRPT performed better than the 2-approximation algorithm on almost instances of the datasets. Furthermore, we have proposed an evolutionary algorithm called PGA that utilizes RRPT as heuristic initialization to further improve the performance of PGA. Experimental validations on different datasets have been carried out to verify the performance of our algorithms. The empirical results demonstrated that our PGA surpasses the previous strong baseline, including ESMFA and SPT. In particular, by using PGA, the energy consumption of the data aggregation tree in a WSN is reduced significantly. In future works, we plan to investigate the problems of maximizing the WSN's lifetime in 3D terrains and apply our proposed algorithms to several related optimization problems for WSNs.

Acknowledgements Tran Cong Dao was funded by Vingroup JSC and supported by the Master, PhD Scholarship Programme of Vingroup Innovation Foundation (VINIF), Institute of Big Data, code VINIF.2021.ThS.BK.08. Nguyen Thi Tam was funded by the Postdoctoral Scholarship Programme of Vingroup Innovation Foundation (VINIF), code VINIF.2022.STS.06. This research was funded by the Office of Naval Research under grant number N62909-23-1-2018 for Huynh Thi Thanh Binh.

Data availability We provide the data used in our paper in the GitHub repository: https://github.com/DaoTranbk/PGA4Mecat.git.

References

Abirami T, Anandamurugan S (2016) Data aggregation in wireless sensor network using shuffled frog algorithm. Wirel Pers Commun 90(2):537–549

Binh HTT, Dinh TP, Bao TT (2019) New approach to solving the clustered shortest-path tree problem based on reducing the search space of evolutionary algorithm. Knowledge-Based Syst 180:12–25

Binh HTT, Long NH, Thang TB, Simon S, et al (2021) A two-level genetic algorithm for inter-domain path computation under node-defined domain uniqueness constraints. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 87–94. IEEE

Dao TC, Hung TH, Tam NT, Binh HTT(2021) A multifactorial evolutionary algorithm for minimum energy cost data aggregation tree in wireless sensor networks. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 1656–1663. IEEE

Deb K, Agrawal RB et al (1995) Simulated binary crossover for continuous search space. Complex Syst 9(2):115–148

Dietzel S, Gürtler J, Kargl F (2016) A resilient in-network aggregation mechanism for vanets based on dissemination redundancy. Ad Hoc Netw 37:101–109

Ennajari H, Maissa YB, Mouline S(2016) Energy efficient in-network aggregation algorithms in wireless sensor networks: a survey. In *International symposium on ubiquitous networking*, pages 135–148. Springer

Gupta A, Ong Y-S, Feng L (2015) Multifactorial evolution: toward evolutionary multitasking. IEEE Trans Evolut Comput 20(3):343–357

Hien VQ, Dao TC, Binh HTT (2022) A greedy search based evolutionary algorithm for electric vehicle routing problem. Appl Intell 53:1–15

Holland John H (1992) Genetic algorithms. Sci Am 267(1):66–73

Kuo T-W, Kate Lin C-J, Tsai M-J (2015) On the construction of data aggregation tree with minimum energy cost in wireless sensor networks: Np-completeness and approximation algorithms. IEEE Trans Comput 65(10):3109–3121

Lee S-L, Park JS, Shon JG (2015) A two-layer cluster head selection based on distance in wireless sensor networks. In *Computer science and its applications*, pages 1003–1007. Springer

Lin H-C, Chen W-Y (2017) An approximation algorithm for the maximum-lifetime data aggregation tree problem in wireless sensor networks. IEEE Trans Wirel Commun 16(6):3787–3798

Mantri DS, Prasad NR, Prasad R (2015) Bandwidth efficient cluster-based data aggregation for wireless sensor network. Comput Electr Eng 41:256–264

Mirjalili S (2019) Genetic algorithm. In *Evolutionary algorithms and neural networks*, pages 43–55. Springer

Mosavvar I, Ghaffari A (2019) Data aggregation in wireless sensor networks using firefly algorithm. Wirel Personal Commun 104(1):307–324

Nguyen N-T, Liu B-H, Pham V-T, Luo Y-S (2016) On maximizing the lifetime for data aggregation in wireless sensor networks using virtual data aggregation trees. Comput Netw 105:99–110

Rida M, Makhoul A, Harb H, Laiymani D, Barhamgi M (2019) Ek-means: a new clustering approach for datasets classification in sensor networks. Ad Hoc Netw 84:158–169

Sheskin DJ (2020) *Handbook of parametric and nonparametric statistical procedures*. crc Press

Singh SK, Kumar P, Singh PJ (2017) A survey on successors of leach protocol. IEEE Access 5:4298–4328

Tam NT, Tuan TQ, Binh HTT, Swami A (2020) Multifactorial evolutionary optimization for maximizing data aggregation tree lifetime in wireless sensor networks. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II*, volume 11413, page 114130Z. International Society for Optics and Photonics

Zar JH (1999) *Biostatistical analysis*. Pearson Education India

Zhang J, Peng H, Xie F, Long J, He A (2018) An energy efficient and reliable in-network data aggregation scheme for wsn. IEEE Access 6:71857–71870