Soft Computing
https://doi.org/10.1007/s00500-023-08923-1

APPLICATION OF SOFT COMPUTING

A phenotype-based multi-objective evolutionary algorithm for maximizing lifetime in wireless sensor networks with bounded hop

Bui Hong Ngoc[1] · Nguyen Thi Tam[1,2] · Huynh Thi Thanh Binh[1] · Le Trong Vinh[2]

Accepted: 20 June 2023

Abstract
Relay node placement with a hop count bound is a crucial problem in enhancing connectivity, lifetime, and reliability in multi-hop wireless sensor networks. However, existing approaches focus solely on minimizing the number of used relay nodes without considering the energy consumption among nodes. This work investigates a relay node placement problem in multi-hop wireless sensor networks with two objectives: minimize the number of used relay nodes, and minimize the maximum node energy consumption to prolong the network's lifetime while still ensuring the network's connectivity. In particular, we consider a hop count bound as a delay constraint to elevate the network's reliability. We propose a multi-objective evolutionary algorithm called GPrim to solve our problem. The algorithm is a combination of edge-set encoding and NSGA-II framework. Leveraging problem-specific properties, we introduce objective-oriented heuristics incorporated into initialization, crossover, and mutation operators to improve the algorithm's convergence. Simulation results on 3D datasets show that the proposed algorithm performs significantly better than existing algorithms on all measured metrics.

Keywords Multi-hop wireless sensor networks · Network lifetime · Multi-objective evolutionary algorithm

1 Introduction

Wireless sensor networks (WSNs) can be defined as self-configured and infrastructure-less wireless networks comprised of spatially dispersed sensors for monitoring and analyzing environmental conditions. The sensed data will be cooperatively transmitted using either single-hop or multi-hop communication to a base station, also known as a *sink*, where the data can be observed and analyzed. WSNs can be built of a few to hundreds or thousands of low-cost, low-power sensor nodes (SNs), typically dispersed at random in a specified sensor field. Those features enable the network to be deployed on the fly and operate unattended, self-organizing without any pre-existing infrastructure. Thereby, WSNs have gained significant attention from researchers for their wide range of applications, including military usage (Singh et al. 2021), environmental monitoring (Lombardo et al. 2017; Pule et al. 2017), surveillance systems (Abdulkarem et al. 2020), health care (Dey et al. 2017), and public safety (Muduli et al. 2018; Ramson and Moni 2017; Rashid and Rehmani 2016).

In the design of any application, prolonging the network's lifetime is a crucial problem due to the limited energy power of the battery deployed in the sensors. When the battery is fully consumed, a sensor can no longer monitor targets; thus, the network can become fragmented and non-functional. Since battery replacement is impossible in many applications, lowering power consumption is often the most viable (Yetgin et al. 2017; Chan et al. 2020; Kumar and Agrawal 2021). One of the prominent techniques is to deploy non-sensing relay nodes (RNs) to communicate with the SNs (Verma et al. 2015; Lin et al. 2020; Priyadarshi et al. 2020; Farsi et al. 2019). The key idea is to increase the network's capability and balance the energy consumption among nodes, enhancing the connectivity, lifetime, and fault tolerance of the WSN. Besides, deploying additional relay nodes could

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

Bui Hong Ngoc
ngocbh.pt@gmail.com

Nguyen Thi Tam
tamnt@vnu.edu.vn

Le Trong Vinh
vinhlt@vnu.edu.vn

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

also shorten a long-hop transmission, which is much more expensive than multiple short-hops (Haenggi and Puccinelli 2005).

Relay node placement problems are generally grouped into unconstrained and constrained strategies. In the former strategy, relay nodes can be placed anywhere in the terrain. In contrast, to avoid unrealistic relay deployments due to physical constraints, the latter restricts the position of the additional relay nodes at certain locations, which are determined in advance. However, both strategies typically form NP-hard problems (Lloyd and Xue 2006; Misra et al. 2009); thus, in practical settings, it is hard to obtain an optimal solution in a suitable amount of time.

Several works have been carried out to find the minimum number of RNs to enhance the connectivity and fault tolerance of a WSN. Ma et al. (2015) investigated the constrained relay node placement in WSNs and proposed a connectivity-aware local search algorithm to find the minimum number of relay nodes so each sensor is covered by at least one relay node (or two in the double cover setting). In Lee et al. (2015), the authors assured the fault tolerance of a partitioned WSN by establishing a bi-connected inter-partition topology while still deploying the least count of relay nodes. Bagaa et al. (2017) later studied the constrained placement of relay nodes and leveraged a Rayleigh block-fading channel and weighted communication graph to construct a realistic routing tree with a minimum number of additional relay nodes. In Hanh et al. (2019), the authors introduced a relay node placement problem in which three objectives are considered: target coverage with connectivity and fault tolerance. They proposed a heuristic algorithm to minimize sensor and relay nodes while ensuring three objectives. The authors in Sheikhi et al. (2021) proposed the two-phases approach based on relay node placement to provide multi-path routing and fault tolerance with higher network connectivity in heterogeneous WSNs.

In practical settings, multi-hop communication is often used in large networks to ensure connectivity and prevent long-hop transmission. However, unlimited hop communication could cause critical issues in terms of quality of service (QoS) due to a close relationship between the hop count and the network's latency and reliability (Bhattacharya and Kumar 2014). Thus, a hop count bound is commonly utilized as a delay constraint in multi-hop WSNs (Bhattacharya and Kumar 2014; Ma et al. 2017, 2018; Liang et al. 2019). Bhattacharya and Kumar (2014) laid the groundwork for a constrained relay placement problem with hop count bound, namely rooted Steiner tree-minimum relays-hop constraint (RST-MR-HC). The authors pointed out the NP-hardness of the RST-MR-HC problem and proposed a polynomial time approximation algorithm for the problem. In Ma et al. (2017) and Ma et al. (2018), the authors used the hop count to measure delay and reliability to formulate the 2-connected hop-constrained relay node placement (HCRNP) problem. Two algorithms, cover-based 1-connected relay placement (C1NP) and cover-based 2-connected relay placement (C2NP), are proposed to solve this problem. Liang et al. (2019) later conducted extensive real-world deployments of WSNs using existing algorithms and then proposed a set-covering-based algorithm (SCA) to ensure the quality of communication in the network with a hop count bound as a delay constraint. In Tam et al. (2021), Tam et al. (2021), the authors consider maximizing network lifetime in multi-hop sensor networks in three-dimensional terrains. A hybrid local search algorithm is proposed to find the near-optimal solution. In Tam et al. (2021), the multifactorial evolutionary algorithm is presented to optimize both wireless single-hop and multi-hop sensor networks simultaneously. The authors in Tam et al. (2021) formulate maximizing network lifetime as a multi-objective problem. They propose the multi-objective evolutionary algorithm based on decomposition to find the set of non-dominated solutions.

Despite the promising results, a drawback of the aforementioned works is the lack of considering the energy consumption of nodes in the placement. The energy consumption of the nodes in a WSN is well known to be imbalanced since it depends heavily on the number of relayed packets and the distance to the next node in the network topology; thus, a balanced load among nodes is an essential factor in enhancing the network's lifetime (Guleria and Verma 2019). However, the real-world deployment of WSNs requires striking an intricate balance between conflicting criteria: the network's lifetime and the cost of deploying additional relay nodes. Deploying more relay nodes could increase the network's capability and provide more possibilities for load balancing but induce more cost in the deployment.

To solve this problem, Tam et al. (2020) have considered two objectives: minimizing the number of relay nodes and minimizing the maximum node energy consumption to prolong the network lifetime. They proposed a weighted-sum approach to finding a routing tree that maximizes the network's lifetime with minimum additional relay nodes. However, there are also several disadvantages that exist within this work. *First*, they only consider the 2-hop wireless sensor network, which is only suitable for small networks. *Second*, the weighted-sum strategy must make certain assumptions when assigning weight values regarding how 'important' a criterion is compared to the other.

To overcome these issues, we investigate a node-energy bottleneck problem in wireless sensor networks with a hop count bound called NEBP. This paper aims to establish a communication structure (routing tree) with minimal deployed relay nodes that prolong the network lifetime while still ensuring the network's connectivity. Different from Tam et al. (2020), we consider a multi-hop scheme with a delay constraint by limiting the maximum number of communication hops for each SN toward BS. Moreover, we focus instead on using multi-objective evolutionary algorithms to solve two objectives simultaneously.

Multi-objective evolutionary algorithms are commonly used to tackle multi-objective optimization problems. Their advantage is the ability to provide optimal or approximate Pareto fronts of non-dominated solutions (also known as the Pareto-efficient or Pareto set) in the objective function space. From these solutions, decision-makers can select from a diverse range of design options. In evolutionary-based approaches, a population of candidate solutions is maintained and evolved toward better solutions. There are two main types of representation of an individual in the population: *indirect* and *direct*.

In an indirect representation, the candidate solution's space (*phenotypic space*) can be mapped to a *genetic space* where a genetic representation of a solution is the *genotype* and its decoded solution is called the *phenotype*. Several genotypic representations are widely used to represent a routing tree in a WSN, including Prufer encoding (Prüfer 1918), link and node biased (Palmer and Kershenbaum 1994), Network random keys (NetKeys) (Rothlauf et al. 2002). Recently, Prakash et al. (2020) also leveraged a permutation encoding with a heuristic decoder to propose a hybrid multi-objective evolutionary algorithm (HMOEA) to find a minimal spanning tree with a minimum diameter. The advantage of genotypic representation is the simplicity of leveraging standard search operators on the evolutionary paradigms (Nayyar et al. 2018). However, finding the proper representation is the biggest challenge of this approach since the genotypic representations often suffer from the low locality (small changes in the code can lead to large changes in the decoded tree) (Prüfer 1918), or infeasible and redundant representations (Rothlauf et al. 2002; Prakash et al. 2020).

Direct representation can be considered the case when the genotypic space is the same as the phenotypic space (Rothlauf and Rothlauf 2006). In direct representation, we can use a simple encoding method such as edge sets encoding (Raidl and Julstrom 2003) and then perform a problem-specific crossover and mutation directly on phenotypes to create new offspring. The main advantage of this scheme is the ability to apply a heuristic to guide search operators (Hao and Liu 2017). Therefore, this paper shall propose an objective-oriented heuristic aiding the standard operator in the direct tree representation to solve the NEBP. In addition, we conducted an extensive experiment to demonstrate the effectiveness of our approach compared to the other encoding methods mentioned above.

Our contributions are listed point by point:

- First, we present a node-energy bottleneck problem in wireless sensor networks with a hop count bound (NEBP), which considers two objectives: (i) minimize the number of used relay nodes; (ii) minimize the maximum node energy consumption to prolong the network lifetime. In particular, we consider a delay constraint by limiting the maximum number of communication hops for each SN toward BS.
- Secondly, we propose GPrim to solve node-energy bottleneck problem in multi-hop wireless sensor networks (NEBP). The novelties of the proposed guided prim NGSA-II (GPrim) can be summarized as follows: (i) according to the problem-specific characteristics, encoding-based edge-set and decoding methodologies are developed to represent the solution space; (ii) we leverage the problem's energy property to develop a heuristic Prim-based crossover and two mutations including energy-oriented mutation and relay-oriented mutation to reduce ineffective moves from standard search operators.
- The proposed algorithm is validated against different encoding methods, including permutation, prufer code, NetKeys, and edge sets. The comparison is delivered on various standard metrics.

The rest of this paper is organized as follows: In Sect. 2, the problem description and notations are given. In Sect. 3, we present our proposed methods, while Sect. 4 illustrates the simulation results. Finally, we conclude the paper by summarizing our main contributions and giving an idea of our future work.

2 Problem statement

2.1 Network structure

We consider the deployment of a wireless underground sensor network as in Tam et al. (2020), with a multi-hop network structure as described in Wu and Liu (2013). A network includes a base station (BS), a set of SNs, and a set of RNs deployed in three-dimensional terrains (this paper uses notations per the digital elevation model (DEM) standards Florinsky 2016). We consider both types of connection: relay nodes—base station and sensor nodes—sensor nodes/relay nodes. Sensing data are gathered by SNs and sent to a BS through a relay node or other sensors. The data transmitted to relay nodes can only be forwarded directly to the base station rather than other sensors or relays.

We assume the sensor network is static, meaning SNs have already been deployed, and a finite set of potential positions for RNs is known in advance. The base station is a sink node deployed at the central terrain with an unlimited power supply while SNs and RNs have the same initial energy, which cannot be replenished. Energy consumption is composed of

Table 1 Network parameters

Parameter	Value
ϵ_{elec}	$50nJ/bit$
ϵ_{fs}	$10pJ/bit/m^2$
ϵ_{mp}	$0.0013pJ/bit/m^4$
ϵ_{DA}	$5pJ/bit$

dissipated energy by the receiver and the transmitter. The amount of consumption depends on the number of SNs sending data through the node, whereas the transmission power is proportionate to the distance of the transmit-receive pair.

2.2 Energy model

Numerous energy dissipation models in WSNs are studied with different assumptions. In this work, we use the same energy model as in Gawade and Nalbalwar (2016), which accounts for the dissipated energy at both the receiver and transmitter during a transmission. The free space model (d^2 power loss) is used for proximal transmissions, and the multi-path fading model (d^4 power loss) is considered for large-distance transmissions. Thus the energy dissipated by the transmitter for transmitting an l-bit packet to a distance d is given by:

$$\tilde{E}_t(d) = \begin{cases} l\epsilon_{\text{elec}} + l\epsilon_{\text{fs}}d^2 & \text{if} \quad d \le d_0 \le r_c, \\ l\epsilon_{\text{elec}} + l\epsilon_{\text{mp}}d^4 & \text{if} \quad d_0 < d \le r_c, \\ \infty & \text{if} \quad r_c < d, \end{cases} \tag{1}$$

where $d_0 = \sqrt{\frac{\epsilon_{\text{fs}}}{\epsilon_{\text{mp}}}}$ is the distance threshold for swapping amplification models and r_c indicates the range with which a node can communicate. In other words, no connection will be established among the nodes out of this range.

The energy consumption of the receiver to receive an l-bit packet is calculated as follows:

$$\tilde{E}_r = l\epsilon_{\text{elec}}. \tag{2}$$

The dissipated energy of a node receiving η packets and transmitting them to the parent node is calculated by the following formula

$$\tilde{E}(\eta, \zeta, d) = \eta\tilde{E}_r + (\eta + \zeta)\tilde{E}_t(d), \tag{3}$$

where $\tilde{E}_t(d)$, $\tilde{E}_r$ are calculated as in Eq. 1 and 2, respectively. The argument ζ is equal to 1 if the node is a sensor node, and 0 otherwise. The argument d is the transmission distance. The network parameters shown in Table 1 are set as in Wu and Liu (2013).

2.3 Problem formulation

We consider a wireless sensor network including a set of deployed sensor node $S = \{s_1, s_2, ..., s_{n_s}\}$, a set of potential relay nodes $R = \{r_1, r_2, ..., r_{n_r}\}$ and a base station denoted as s_0. The position of each node is represented as a single point in a 3D space that is interpolated from the DEM model. The communication between two nodes can only be established if the Euclidean distance between them does not exceed the communication range r_c.

This paper aims to establish a communication structure (routing tree) with a minimal number of deployed relay nodes that prolongs the network lifetime while still ensuring the network's connectivity. In other words, we aim to construct the unique path for each sensor transmitting sensing data to BS such that with the minimum number of deployed relays, the maximum energy consumption of every node in the network is minimized. Besides, to avoid the problem of unbalanced energy consumption caused by multi-hop transmission, we also consider the max-hop constraint $\hbar$ that limits the maximum number of communication hops for each SN toward BS.

We denote the problem as NEBP. The formal formulation below models the desired structure as a Steiner tree (Hwang and Richards 1992).

Input:

- $G = (V, E)$ is an undirected graph, where $V = S \cup R \cup \{s_0\}$ is set of vertices in the graph, S is set of sensor nodes, R is set of relays, and s_0 corresponds to base station.
- $N = S \cup \{s_0\}$ denotes the set of terminal nodes.
- $r_c \in \mathbb{R}^+$ is the communication range.
- $d : V \times V \to \mathbb{R}^+$ is the distance function. An edge $e = (u, v) \in E$ only if $d(u, v) \le r_0$.
- $\hbar \in \mathbb{N}^+$ is the max-hop constraint.

Output: A valid output is a Steiner tree (Definition 2.1) $T = (V_T, E_T)$ of graph G that spans the set of terminal nodes $N = S \cup \{s_0\}$.

Constraints:

- Every selected RNs (Steiner points) directly connect to base station s_0 (denoted as node 0)

  $(s_0, v) \in E_T \quad \forall v \in V_T \cap R.$

- The unique path from a specified root s_0 to any other node has no more than $\hbar$ hops (edges)

  $|\text{path}(s_0, v)| \le \hbar \quad \forall v \in V_T,$

  where $|\text{path}(u, v)|$ denotes the length of the unique path between two nodes u and v.

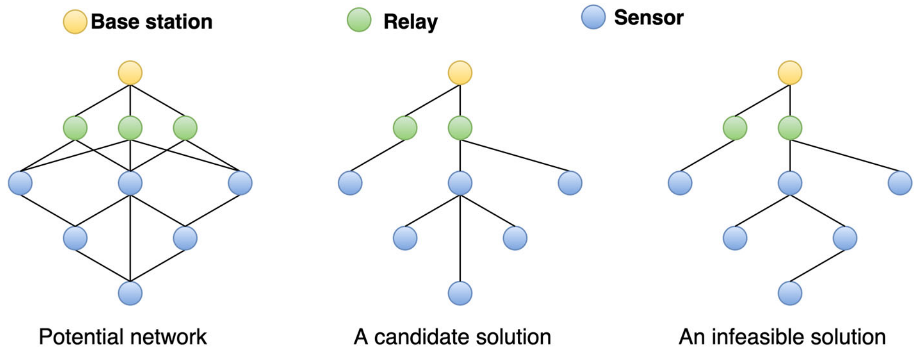


Fig. 1 An example of a network with 3 relays, 6 sensors, and max-hop constraint is 3

Figure 1 shows an example of a network with three relays and six sensors.

Objectives: The NEBP seeks a Steiner tree $T = (V_T, E_T)$ in the valid output space that optimizes two following objectives:

- Minimize the number of selected RNs (Steiner node):

$$|V_T \setminus N| \rightarrow \min . \tag{4}$$

- Minimize the maximum energy consumption of each node:

$$\max_{v \in V_T \setminus \{s_0\}} \tilde{E}_v(\eta, \zeta_v, d) \rightarrow \min, \tag{5}$$

where $\tilde{E}_v$ is calculated as Eq. 3 in which η is the number of descendants of v, ζ_v is a binary number that indicates whether v is the sensor (1) or not (0) and d denotes the distance from v to its parent.

Definition 2.1 (Steiner tree) Given an undirected graph $G = (V, E)$ and a set of terminal nodes $N \subseteq V$. A tree $T = (V_T, E_T)$ is called a Steiner tree if it contains no cycles and spans all terminal nodes, $N \subseteq V_T \subseteq V$. The set of nodes $V_T \backslash N$ is called Steiner nodes.

3 Proposed approach

We propose a phenotype-based multi-objective evolutionary algorithm (MOEA) named *GPrim* to solve the NEBP. In this scheme, the population is first initialized by a random-tree algorithm in which the candidate solutions are encoded by the edge sets encoding method. Then, we apply the NSGA-II strategy (Deb et al. 2002) to maintain and evolve the population toward better solutions using the problem-specific search operators. We leverage the problem's energy property to develop a heuristic Prim-based crossover and two objective-oriented mutations to reduce ineffective moves from standard search operators. The specifics of solution representation, initialization, crossover, and mutation are described below.

3.1 Solution representation

Although a direct representation needs no mapping between the phenotypic and genotypic space, a data structure is still necessary for processing (Li 2001; Rothlauf and Rothlauf 2006). Since the search operators are performed directly on the phenotype representation, limiting infeasible and redundant outputs depends on search operators rather than the representation itself. The representation now only affects the memory and running time of the algorithm. Thus, we use edge-set encoding on this problem for its simplicity. This encoding can act as the basis for evaluating the solution or be converted to an adjacency list in linear time.

Since the aim of the NEBP, the problem is to find a Steiner tree that can connect all SNs to the BS, the number of vertices in the solutions is not consistent. For simplicity, we initialize solutions with the connections from the BS to every RN, and this structure is maintained in all candidate solutions in the population. The RNs with no connection to any SNs are later removed from the output structure by the decoder.

3.2 Initialization

In the hop-constrained spanning tree problem, the number of hops in a rooted tree (h) is bounded by its diameter (d):

$$d/2 \leq h \leq d.$$

Therefore, we leverage PrimRST (Raidl and Julstrom 2003) to initialize the candidate solutions, as PrimRST tends to generate low-diameter trees which are more likely to satisfy the max-hop constraint. The PrimRST algorithm uses *Prim's* scheme to greedily create a spanning tree from a start node by adding an adjacent node at random, regardless of its weight. Moreover, we adapt PrimRST to consider max-hop constraint by maintaining nodes' depth while creating a tree. We call this algorithm as *HCPrimRST* (Algorithm 1). Applying HCPrimRST with max-hop constraint may lead to an invalid, non-connected structure. The initialization is thus divided into two phases. The first phase initializes edges $T = \{(0, v) \in E | v \in R\}$ and runs the algorithm with max-hop constraint; and in the second phase, we relax its constraint and continue to build the tree obtained from the first phase to get a valid connected tree.

Algorithm 1: Random spanning tree generation with max-hop constraint

```
Input : The set of initialized edges $T$
        The set of vertices $V$
        The set of potential edges $E$
        The max-hop constraint $\hbar$
Output: The set of used edges $T$
function HCPrimRST( $T, V, E, \hbar$ )
    $C \leftarrow \{u, v | (u, v) \in T\}$ ;            // set of connected nodes
    $d \leftarrow$ depth of vertices in partial tree $T$ ; // by dfs from root 0
    $A \leftarrow \{(u, v) \in E | u \in C, v \notin C\}$ ; // eligible edges
    while $A \neq \emptyset$ do
        Choose $(u, v) \in A$ at random ;
        $A \leftarrow A \setminus (u, v)$ ;
        if $v \notin C$ and $d_u < \hbar$ then
            $T \leftarrow T \cup (u, v)$ ;
            $C \leftarrow C \cup \{v\}$ ;
            $A \leftarrow A \cup \{(v, w) \in E | w \notin C\}$ ;     // add v's adjacency to A
            $d_v \leftarrow d_u + 1$ ;
    return $T$
```

3.3 Crossover operator

For crossover, we could apply directly the HCPrimRST algorithm to create an offspring T_{cr} from a combined graph $G_{cr} = (V, E_{T_1} \cup E_{T_2})$, where T_1, T_2 are the parental trees. However, this algorithm might create many infeasible or ineffective offsprings due to its random nature. We delineate in this section an energy-aware modification of the HCPrimRST-based crossover to prioritize the offspring's structures that use less energy power than their parents while maintaining the diversity of the offspring.

Following the energy model (3), we notice that the dissipated power of a node is affected by three factors, including the number of packets it carries (η_u) (descendant nodes), whether it is a sensor or relay ($\zeta_u \in \{0, 1\}$), and the distance of the transmit-receive pair to its parent ($\xi(d)$):

$$\tilde{E}_u = \eta_u(\tilde{E}_r + \tilde{E}_t(d)) + \zeta_u \tilde{E}_t(d) \tag{6}$$

$$\Leftrightarrow \quad \eta_u = \frac{\tilde{E}_u - \zeta_u \tilde{E}_t(d)}{\tilde{E}_r + \tilde{E}_t(d)}. \tag{7}$$

Equation (6) suggests that if we know the maximum energy a node can use and its parent, we also know the maximum number of packets it can carry. Let us assume that the network has an energy limit of $\tilde{E}_{max}$. Before adding an edge (u, v) where $u \in C, v \notin C, d_u < \hbar$, we can check if this causes the network to exceed the energy limit by tracking the number of descendants that a node in the partial tree can receive.

We denote Ψ_u as the number of descendants a node can have without exceeding $\tilde{E}_{max}$. P_u is a set of u's parent nodes and itself. An edge (u, v) is considered valid if $\Psi_u > 0$. Connecting (u, v) reduces the capacity of the nodes in P_u by 1 unit:

$$\Psi_t = \Psi_t - 1 \quad \forall t \in \mathrm{P}_u. \tag{8}$$

Thus, Ψ_v is calculated by:

$$\Psi_v = \min\ \{\eta_v, \Psi_u\}, \tag{9}$$

where η_v is calculated as Eq. (6) with the energy limit $\tilde{E}_{max}$. For example, with the graph presented in Fig. 2, two potential edges connect node 9 to the partial tree: (7, 9) and (8, 9). However, due to $\Psi_7 = 0$, connecting the edge (7, 9) inevitably causes one of the parents of node 7 to be exhausted. Thus, the edge (8, 9) is prioritized in this case. Note that adding a child affects all parent nodes. Thus, the maximum number of descendants of a node does not exceed its parents, that is

$$\Psi_u = \min_{t \in \mathrm{P}_u} \Psi_t. \tag{10}$$

To reduce computation, an update of node u (Eq. 10) is only executed when it is necessary. In the example in Fig. 2, Ψ_4 should be updated to 2. However, it is unnecessary until another edge of node 4 is involved. The operator can be done in $O(n\hbar)$.

3.4 Mutation operator

One of the most widely used mutations for tree-based problems is *edge insertion mutation* (Raidl 2000). The tree is mutated by randomly adding a non-tree edge (creating a cycle) and then randomly deleting a tree edge from the created cycle (removing the cycle). This mutation can be applied to most tree-based problems. However, exploring neighbors randomly also becomes the drawback of this mutation. In most cases, it produces ineffective moves, which lead to invalid or lower-quality solutions, especially when approaching optimal PF.

This subsection presents two problem-specific mutations that improve the algorithm's convergence speed. The first mutation targets energy usage, while the second aims to

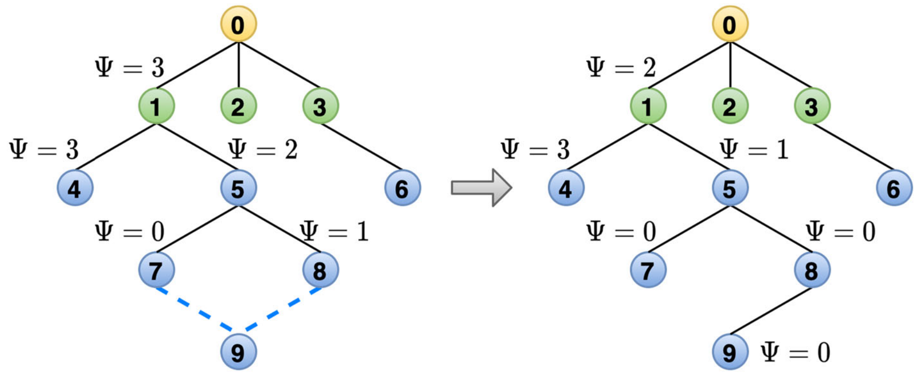


Fig. 2 An example of calculating maximum number of children. Dashed lines denote potential edges

reduce the number of used relays. These mutations are used simultaneously and chosen randomly for each offspring with predefined probabilities.

3.4.1 Energy-oriented mutation

Let us look closer into the edge insertion mutation. A mutated solution can be represented by its direct parent and a pair of added edge $e^+ = (u^+, v^+)$ and deleted edge $e^- = (u^-, v^-)$. Both edges are chosen at random, most of which lead to a worse solution or violate the max-hop constraint. However, we can leverage each node's maximum number of descendants Ψ (Sect. 3.3) to find and prioritize pairs of edges that ensure a better solution.

Denote $\tilde{E}_{\max}$ as the maximum energy usage on the parent network T and ϕ as the node using the most energy. The energy usage of ϕ is composed of two factors. One is the number of descendants that ϕ has to carry, and one is ϕ's distance to its parent in the network. Reducing the maximum energy usage of the network requires one of two factors to decrease. Consider the subtree $T_\phi = (V_\phi, E_\phi)$, which includes ϕ and its children. We define potential added edge (u, v) as the edge that connects a part of subtree T_ϕ to the remaining subtree $T \backslash T_\phi$ and satisfies the following constraints:

$$u \in V_\phi, \quad v \in V \setminus V_\phi, \quad \eta_u \le \Psi_v, \quad h_u + d_v + 1 \le \hbar,$$

where d_u, h_u, η_u is depth (from root to u), maximum hop (from u to farthest leaf) and the number of children of node u, respectively. Also, we refer to p_u as the parent of u in the parental tree. Then, adding a potential added edge (u, v) and deleting (u, p_u) creates a new tree that satisfies the max-hop constraint and has lower energy usage in node ϕ. Figure 3 shows an example of this process.

When combining the above heuristic with edge insertion mutation, we find a set of potential edge pairs in the tree. If this set is not empty, we randomly choose one pair and create a mutated tree. Otherwise, the original edge insertion mutation is used to complete the mutation operator. Algorithm 2 presents this energy-oriented mutation procedure. The complexity of this mutation is $O(max(n\hbar, m))$.

Algorithm 2: The energy-oriented mutation algorithm

```
Input  : The edge-set of parental tree $T$
         The set of vertices $V$
         The set of potential edges $E$
         The energy limit $\tilde{E}_{max}$
         The max-hop constraint $\hbar$
Output: The set of edges in the offspring $T$
1  Find a node that uses most energy $\phi$ in parental tree $T$;
2  Run depth-first search in parental tree $T$ calculate
3      $V_\phi \in V$ is a set of children node of $\phi$ in parental tree;
4      $\Psi \leftarrow$ the maximum number of children with the energy limit $\tilde{E}_{max}$;
5      $d \leftarrow$ the depth of nodes ;      // from root to node
6      $h \leftarrow$ the maximum hop of nodes ;     // from node to leaf
7      $\eta \leftarrow$ the number of children of nodes;
8  $F = \{(u, v) \in E | u \in V_\phi \wedge v \in (V \setminus V_\phi) \wedge \eta_u \le \Psi_v \wedge h_u + d_v + 1 \le \hbar\}$; // A set of potential edges
9  if $F \ne \emptyset$ then
10     Choose an edge $(u_\phi, v_\phi) \in F$ at random;
11     $T \leftarrow T \setminus (u_\phi, p_{u_\phi})$ ;   // $p_{u_\phi}$ is parent of $u_\phi$ in parental tree
12     $T \leftarrow T \cup (u_\phi, v_\phi)$;
13 else
14     Choose an edge $(u^+, v^+) \in E \setminus T$ at random;
15     $T \leftarrow T \cup (u^+, v^+)$;
16     Find a set of edges $C$ in the created cycle;
17     Choose an edge $(u^-, v^-) \in C$;
18     $T \leftarrow T \setminus (u^-, v^-)$
19 return $T$
```

3.4.2 Relay-oriented mutation

Due to the tendency toward star-like structures, PrimRST and HCPrimRST uses more relays on average than KruskalRST and RandWalkRST (Fig. 5). In comparison, both crossover and energy-oriented mutation focus on optimizing the energy objective. The optimization of the number used relays

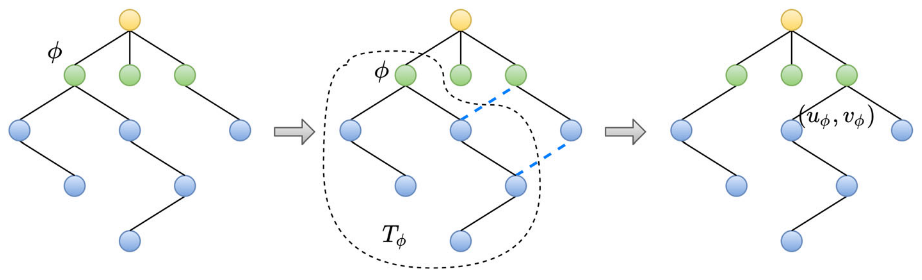


Fig. 3 An example of the energy-oriented mutation. The dashed blue lines denote the potential added edges

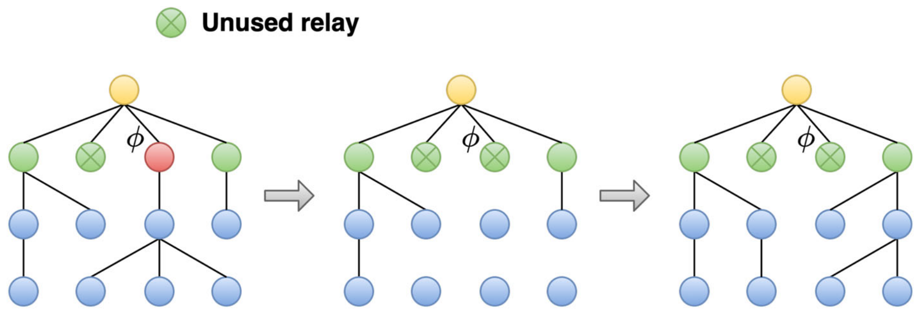


Fig. 4 An example of the relay-oriented mutation

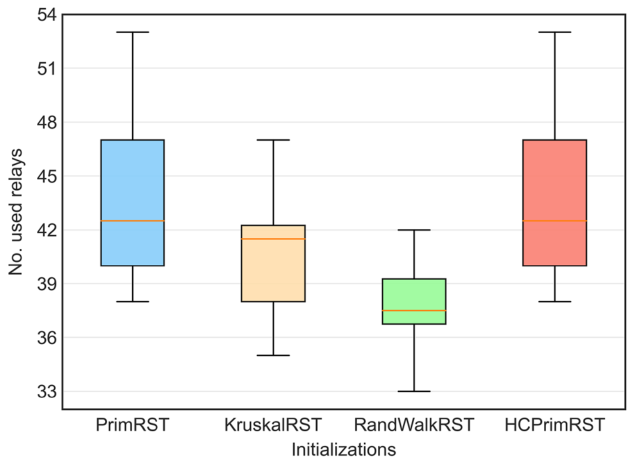


Fig. 5 The distribution the number of used relays generated by different random-tree algorithms on a graph with 100 relays and 100 sensors

Algorithm 3: The relay-oriented mutation algorithm

```
Input  : The edge-set of parental tree $T$
         The set of vertices $V$
         The set of potential edges $E$
         The energy limit $\tilde{E}_{max}$
         The max-hop constraint $\hbar$
Output: The set of edges in the offspring $T$
1  Find a set of used relays $\Phi$ in the parental tree $T$;
2  Choose one relay $\phi \in \Phi$ at random.;
3  $\Phi \leftarrow \Phi \setminus \{\phi\}$;
4  Find a set of children node $V_\phi$ of $\phi$;
5  $T \leftarrow T \setminus \{(u, v) \in T | u \in V_\phi \vee v \in V_\phi\}$;
6  $E \leftarrow E \setminus \{(u, v) \in E | u \in (V_r \setminus \Phi) \vee v \in (V_r \setminus \Phi)\}$;
   // disable unused relays
7  HCPrimRST($T$, $S \cup \Phi$, $E$, $\hbar$, $\tilde{E}_{max}$ );
8  HCPrimRST($T$, $S \cup \Phi$, $E$, $\hbar$, $\infty$);
9  HCPrimRST($T$, $S \cup \Phi$, $E$, $\infty$, $\infty$);
10 return $T$
```

depends on the distribution of the random-tree algorithm implemented in the initialization and crossover operator. We propose a more direct mutation strategy for reducing the number of used relays: one random relay from the parental tree is disabled, while its children (sensors) are connected to the remaining relays. The HCPrimRST algorithm is first applied with an energy limit as in the crossover operator. Constraints are later relaxed to ensure that all sensors are connected. An example of this mutation is shown in Fig. 4 and a pseudocode is provided in Algorithm 3 (Fig. 5). This mutation can be done in $O(n\hbar)$.

4 Experiments and results

4.1 Experiment settings

In our experimental studies, network parameters are set as in Tam et al. (2020) where network constants are as in Table 1 with parameters $l = 4000$ and $d_0 = \epsilon_{\text{md}} \div \epsilon_{\text{fs}}$. We also assume that all sensors and relays have the same radius $r = 25m$. All algorithms were implemented in Python using the GeneticPython (Bui 2020) and NumPy (Van Der Walt et al. 2011) frameworks.[1] All experiments are performed in a single Intel Xeon(R) E-2124 G 3.40 GHz CPU with 16 GB

[1] Source codes and datasets: https://github.com/ngocbh/nebp_wsn.

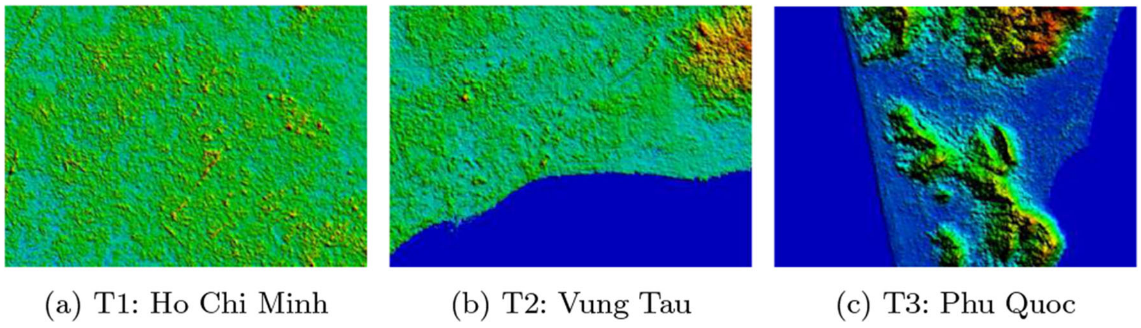


Fig. 6 Height heatmaps of terrains

Table 2 Description of network instances. The last column refers to the density of the communication graph

	Instance	Node distribution	Terrain	Terrain size	No.relays	No. sensors	Density
$S1$	NIn1	Gaussian	T1	200×200	20	20	0.53
	NIn2	Gaussian	T2	200×200	20	20	0.61
	NIn3	Gaussian	T3	200×200	20	20	0.53
	NIn4	Uniform	T1	200×200	20	20	0.53
	NIn5	Uniform	T2	200×200	20	20	0.53
	NIn6	Uniform	T3	200×200	20	20	0.48
$S2$	NIn7	Gaussian	T1	200×200	40	40	0.28
	NIn8	Uniform	T2	200×200	40	40	0.1
	NIn9	Gaussian	T2	500×500	100	100	0.19
	NIn10	Uniform	T3	500×500	100	100	0.17
	NIn11	Gaussian	T3	1000×1000	200	200	0.52
	NIn12	Uniform	T1	1000×1000	200	200	0.15
$S3$	NIn13	Gaussian	T1	200×200	40	80	0.2
	NIn14	Uniform	T2	200×200	40	80	0.18
	NIn15	Gaussian	T2	500×500	100	200	0.1
	NIn16	Uniform	T3	500×500	100	200	0.09
	NIn17	Gaussian	T3	1000×1000	200	400	0.28
	NIn18	Uniform	T1	1000×1000	200	400	0.28

RAM running on Ubuntu Linux 16.04. No extra parallelization apart from the default NumPy acceleration is used.

4.2 Datasets

We generate 18 network instances according to previous works (Hai and Le Vinh 2017; Tam and Hai 2018; Tam et al. 2020). Three real 3D terrain datasets in Vietnam (Fig. 6) are used as the area to place sensors and relays. All terrains are defined according to the digital elevation model (DEM) standard. The sensors and potential relay nodes are deployed in each terrain according to two distributions (Gaussian and Uniform).

We split the data into three sets: $S1$, $S2$, and $S3$. $S1$ includes 6 instances with 20 RNs and 20 SNs. $S2$ contains 6 instances with increasing SNs in the network. $S3$ uses the same settings as $S2$ with twice the SNs in the network. The usage of these sets is discussed in detail in Sect. 4.5. Details of the network instances are shown in Table 2.

4.3 Performance metrics

Comparing Pareto fronts is unfortunately not very straightforward. No single metric can best cover all aspects (cardinality, convergence, diversity) (Konstantinidis and Yang 2011; Riquelme et al. 2015; Audet et al. 2020). Thus, this study considers the following five metrics:

- **Inverted Generational Distance** (IGD) (Coello and Cortés 2005): Given an optimal Pareto front (PF) P, IGD of a approximation set S is calculated as:

$$IGD(S, P) = \frac{1}{|P|}\left(\sum_{p \in P} d_p^l\right)^{\frac{1}{l}},$$

where $d_p = \min_{s \in S} \|f(s) - f(p)\|_2$ and $l = 2$ (in general). IGD can capture both the convergence and diversity of approximation PFs. However, this metric cannot be used without an optimal PF.

- **Hypervolume** (HV) (Zitzler and Thiele 1999): As described in Audet et al. (2020), the hypervolume indicator is the volume of the space dominated by the Pareto front approximation S and delimited from above by a reference point $r \in R^m$ such that for all $s \in S, s \prec r$. The hypervolume is given by:

$$HV(S, r) = \lambda_m \left(\bigcup_{s \in S} [s, r] \right),$$

where λ_m is the m-dimensional Lebesgue measure. In a bi-objective problem ($m = 2$), hypervolume can easily be obtained in linear time.

- **Convergence of two sets** (C) (Zitzler and Thiele 1998): This metric is widely used to capture the convergence of two approximation sets. It is defined by the ratio of a set dominated by others divided by its cardinality:

$$C(A, B) = \frac{|\{b \in B | \exists a \in A : b \prec a\}|}{|B|}.$$

If $C(A, B) = 1$, all solutions of B are dominated by solutions of A. Note that we have to compute both $C(A, B)$ and $C(B, A)$ since their sum is not always equal to 1.

- **Delta-metric** (Δ) (Deb et al. 2002): This is a unary metric used to measure a PF's diversity. In bi-objective problem, $\Delta(S)$ of a PF S is defined as:

$$\Delta(S) = \frac{d_f + d_l + \sum_{i=1}^{|S|-1} |d_i - \bar{d}|}{d_f + d_l + |S|\bar{d}},$$

where d_f and d_l are the Euclidean distances between the extreme solutions in one objective and the boundary solutions of S. d_i is the Euclidean distance between consecutive solutions of approximation set S and $\bar{d}$ is the mean of d_i. The smaller value of $\Delta(S)$ gives a better spread of the PF.

- **Cardinality** ($ONVG$) (Schott 1995): This is a straightforward metric computing a PF's cardinality

$$ONVG(S) = |S|.$$

Since the NEBP is a bi-objective problem, one of which is discrete with a low range, $ONVG$ is essential to measure the spread of PFs.

4.4 Baselines

We compare our proposed method with five algorithms, including different encoding methods and search operators that are widely used in tree encoding problems:

- *HMOEA*: We adopt a hybrid multi-objective evolutionary algorithm (HMOEA) proposed in Prakash et al. (2020) for bi-Objective Minimum Diameter-Cost Spanning Tree (bi-MDCST) problem to tackle the NEBP. A permutation encoding with a heuristic CBTC-based decoder is proposed to represent several solutions of varying relay's constraint in which the *order crossover* and *swap mutation* are used for reproduction.
- *Prufer encoding (Prufer)*: Prufer encoding is used in conjunction with two common search operators in integer chromosomes: *uniform crossover* and *swap mutation*. To reduce the infeasible ratio caused by incomplete graphs, we repair the decoded edges using only valid edges with KruskalRST to complete the candidate tree.
- *NetKeys encoding (Netkeys)*: In this scheme, we encode a tree using general Network Random Keys with *uniform crossover* and *swap mutation*, as suggested in Rothlauf et al. (2002).
- *Edge-set and PrimRST (Prim)*: This algorithm is based on the application of edge sets to the degree-constraint minimum spanning tree problem (d-MST) as presented in Raidl and Julstrom (2003). In the recombination process, we use *PrimRST* to create offsprings from the parents' edges. In the mutation, we use the *edge insertion mutation*, as described in Sect. 3.4.
- *Edge-set and KruskalRST (Kruskal)*: All settings are the same as in the Prim-based approach, but *KruskalRST* is used instead of *PrimRST* in the recombination.
- *Edge-set and Guided Prim (GPrim):* This is our proposed algorithm presented in Sect. 3.

In these approaches, the first three algorithms (HMOEA, Prufer and NetKeys) are indirect representations, while the rest (Prim, Kruskal, and GPrim) are direct representations. Leveraging the discreteness of one of the objectives, HMOEA maintains an external population to store the Pareto front. Each offspring will be evaluated in different constraints of relays to update the Pareto front as well as the main population, where the tournament of size three is used to select the parents. Meanwhile, the remaining algorithms are applied to the NSGA-II structure with *binary tournament selection*. Note that our methods can also apply to other multi-objective optimizations (MOOs) algorithm structures such as MOEA/D (Cheng et al. 2015), SPEA2 (Zitzler et al. 2001). However, such combinations are reserved for future works.

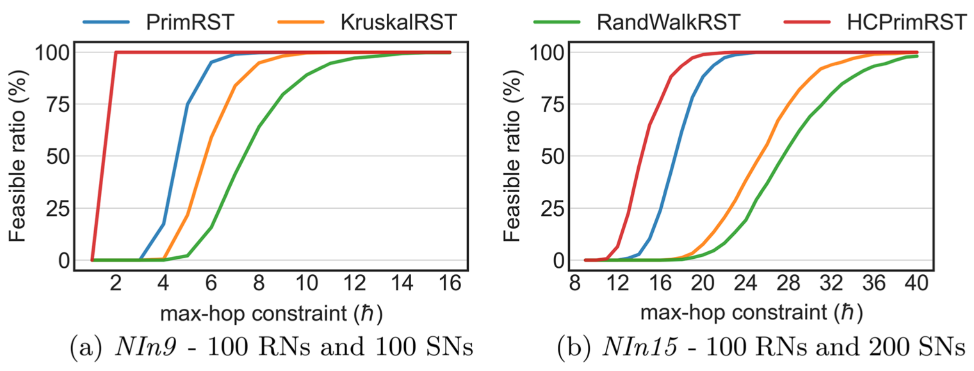


Fig. 7 Feasible ratio of the population on various max-hop constraints

Table 3 Performance of competing algorithms on the set $S1$

Instance	HMOEA	Prufer	NetKeys	Prim	Kruskal	GPrim
(a) Inverted Generational Distance (IGD)						
NIn1	0.039 ± 0.007	0.037 ± 0.005	0.039 ± 0.006	0.033 ± 0.008	0.036 ± 0.004	**0.000** ± 0.000
NIn2	0.043 ± 0.011	0.038 ± 0.004	0.038 ± 0.005	0.033 ± 0.007	0.036 ± 0.007	**0.000** ± 0.000
NIn3	0.035 ± 0.006	0.036 ± 0.005	0.033 ± 0.007	0.034 ± 0.006	0.035 ± 0.003	**0.000** ± 0.000
NIn4	0.033 ± 0.007	0.036 ± 0.006	0.036 ± 0.007	0.034 ± 0.007	0.039 ± 0.005	**0.000** ± 0.000
NIn5	0.034 ± 0.004	0.038 ± 0.007	0.039 ± 0.004	0.032 ± 0.005	0.037 ± 0.007	**0.000** ± 0.000
NIn6	0.035 ± 0.005	0.036 ± 0.004	0.037 ± 0.006	0.033 ± 0.007	0.034 ± 0.006	**0.000** ± 0.000
(b) Cardinality ($ONVG$)						
NIn1	9.90 ± 0.831	7.90 ± 1.375	8.30 ± 1.269	9.90 ± 2.256	8.90 ± 0.943	**20.00** ± 0.000
NIn2	9.40 ± 1.281	8.00 ± 1.095	8.60 ± 1.281	9.60 ± 1.744	9.40 ± 1.625	**20.00** ± 0.000
NIn3	10.30 ± 0.900	8.00 ± 1.183	10.40 ± 1.562	9.30 ± 1.616	9.30 ± 0.900	**20.00** ± 0.000
NIn4	10.30 ± 1.100	8.40 ± 1.428	8.70 ± 1.269	9.10 ± 1.700	8.30 ± 1.100	**20.00** ± 0.000
NIn5	10.30 ± 1.005	8.20 ± 1.400	8.70 ± 0.781	9.70 ± 1.552	8.60 ± 1.625	**20.00** ± 0.000
NIn6	10.70 ± 0.781	7.50 ± 1.025	8.70 ± 1.487	10.00 ± 2.449	9.70 ± 1.900	**20.00** ± 0.000

Bold values indicate the best values

4.4.1 Initialization analysis

As discussed in Sect. 3.2, KruskalRST and PrimRST are biased toward star-like structures while RandWalkRST gives an unbiased initialization (Raidl and Julstrom 2003). In order to examine the ability to generate valid solutions in the hop-constraint problem, we run four initializations (PrimRST, KruskalRST, RandWalkRST, HCPrimRST) on two instances (*NIn9* and *NIn15*) with various max-hop constraint $\hbar$. Each algorithm is invoked 1000 times for each value of $\hbar$. Figure 7 shows the feasible solution ratio of the initializations on a different value of $\hbar$. The HCPrimRST gives the best feasible ratio in all max-hop constraints. The feasible ratio of the remaining methods (PrimRST, KruskalRST, and RandWalkRST) is inversely proportional to the average diameter in the results shown in Raidl and Julstrom (2003). For example, when $\hbar = 14$, HCPrimRST produces valid solutions 45% of the time. This metric for PrimRST is 2.8%. KruskalRST and RandWalkRST produced no feasible solutions on average. If KruskalRST or RandWalkRST was applied in this case, it might leave the algorithm with no valid solutions in the first few generations.

Based on the above results, in the following experiments, HCPrimRST will be used as the initialization for all compared algorithms.

4.4.2 Efficiency evaluation

In this experiment, we look into each algorithm's efficiency compared to an approximation of optimal PF. We use six small instances in the set $S1$. Optimal PF is approximated by running each algorithm 100 times with different seeds, each with 1000 generations. The best solutions are combined to generate a final approximation PF.

Table 3 summarizes the results of competing algorithms through three metrics (IGD, and $ONVG$). GPrim outperforms HMOEA, Prufer, NeyKeys, Prim, and Kruskal with regard to both convergence and diversity. GPrim's PF approaches the approximation PF in most cases. Between standard direct and indirect representations, Prufer showed the worst results in the aforementioned aspects, while

NetKeys and HMOEA provide better results than standard direct representations (Prim, Kruskal). In Fig. 8, we show the convergence profiles of IGD values on network instance *NIn1*. GPrim again shows faster convergence than the remaining approaches, while the standard direct representations (Prim and Kruskal) show promising results.

Considering the performance in the number of non-dominated solutions, GPrim consistently finds the maximum value of $OVNG$. This result demonstrates the effectiveness of the relay-oriented mutation. HMOEA, NetKeys, Prim, and Kruskal perform similarly and fluctuate around 9 solutions for each instance. Meanwhile, Prufer performs worst on $OVNG$.

4.5 Results and discussions

We design four main experiments to study the behavior of the approaches through four aspects:

- *Efficiency*: In this experiment, we use six instances in $S1$ and find an approximation of optimal PF for each instance by combining the results of the six algorithms with a vast number of generations. We later compare the algorithms on the IGD metric, as well as their convergence process through each generation.
- *Scalability*: This experiment uses 12 instances in sets $S2$ and $S3$, to study the behavior of algorithms as network complexity increases.
- *Sparsity of feasible solution*: This experiment aims to investigate the ability to handle sparse solution spaces by reducing the max-hop constraint on each instance in all datasets.
- **Density of the network**: Various communication ranges are used to examine the impact of the graph's density on the algorithms.

4.5.1 Parameter selection

We set the population size to 100 while the number of generations is set to 100. The max-hop constraint is set depending on the complexity of the network instance (see Table 4). Each experiment is performed over 10 independent runs with different seeds.

Since each algorithm has different crossover and mutation operators, we perform a grid search to select crossover and mutation probability. Five random instances are generated, on which the probabilities are searched independently for each algorithm. Table 5 shows the resulting settings.

Table 4 The max-hop constraint ($\hbar$) on each type of dataset. All instances are ensured to have valid solutions

	Type 1	Type 2	Type 3	Type 4
$\hbar$	6	8	12	16

Table 5 The operator's probability of each algorithm. pc is the crossover probability and pm is the mutation probability. For GPrim, pm represents a pair of energy-oriented and relay-oriented mutation probability

	HMOEA	Prufer	NetKeys	Prim	Kruskal	GPrim
pc	0.6	0.9	0.9	0.9	0.9	0.5
pm	0.4	0.5	0.1	0.5	0.5	(0.5, 0.5)

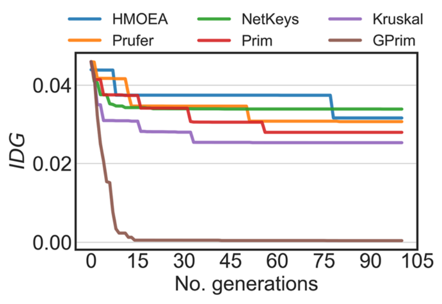
4.5.2 Scalability evaluation

This experiment aims to investigate the behavior of the algorithms on different structures of the graph, especially when the graph size is expanded. We use 12 network instances in sets $S2$ and $S3$ with different distributions and sizes to demonstrate the effectiveness of the algorithm in various topologies of the graph. Since finding the approximation PF costs a massive amount of the computation power on complex network instances, we consider the convergence of two sets (C-metric) and delta metric (Δ) instead of IGD.

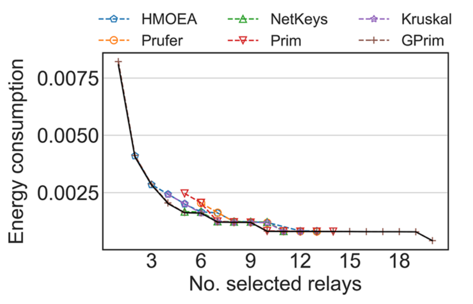
In terms of convergence, we use *box plots* similar to Zitzler and Thiele (1999) to summarize C-metric values of different runs (see Fig. 9). Results show that GPrim produces dominating PFs over other algorithms in most cases, with C values asymptotic to 1 in most test cases. Specifically, GPrim covers more than 88% on average of the fronts computed by HMOEA, while only 0.03% of which dominates the solutions of GPrim. In several instances (*NIn11, NIn12, NIn17, NIn18*), HMOEA produces some better solutions over GPrim with fewer relays due to the mechanism that decodes and evaluates each individual on different relay constraints. It diversifies the evaluated solutions across different numbers of relays. As a trade-off, HMOEA shows poor energy consumption and C-metric results, even in comparison with NetKeys, Prim, and Kruskal. Comparing GPrim and Kruskal, the PFs of GPrim still dominate the PFs of Kruskal in the test set $S3$, where the number of sensors is twice the number of potential relays. In the set $S2$, Kruskal produces some better solutions over GPrim in energy consumption with small networks (*NIn8*) or use fewer relays in more extensive networks (*NIn12*). However, GPrim still covers, on average, more than 97% of the PFs computed by Kruskal in all network instances. Figure 10 illustrates the PFs of six algorithms with a specified seed.

In terms of hypervolume (see Table 6a), we can observe that GPrim outperforms Prufer, NetKeys, Prim, and Kruskal in all instances, especially on the $S3$ set. Meanwhile,

Fig. 8 Comparison of five algorithms on *NIn1*



(a) IGD metric over generations



(b) The Pareto-fronts comparison where black line is the appoximate optimal PF

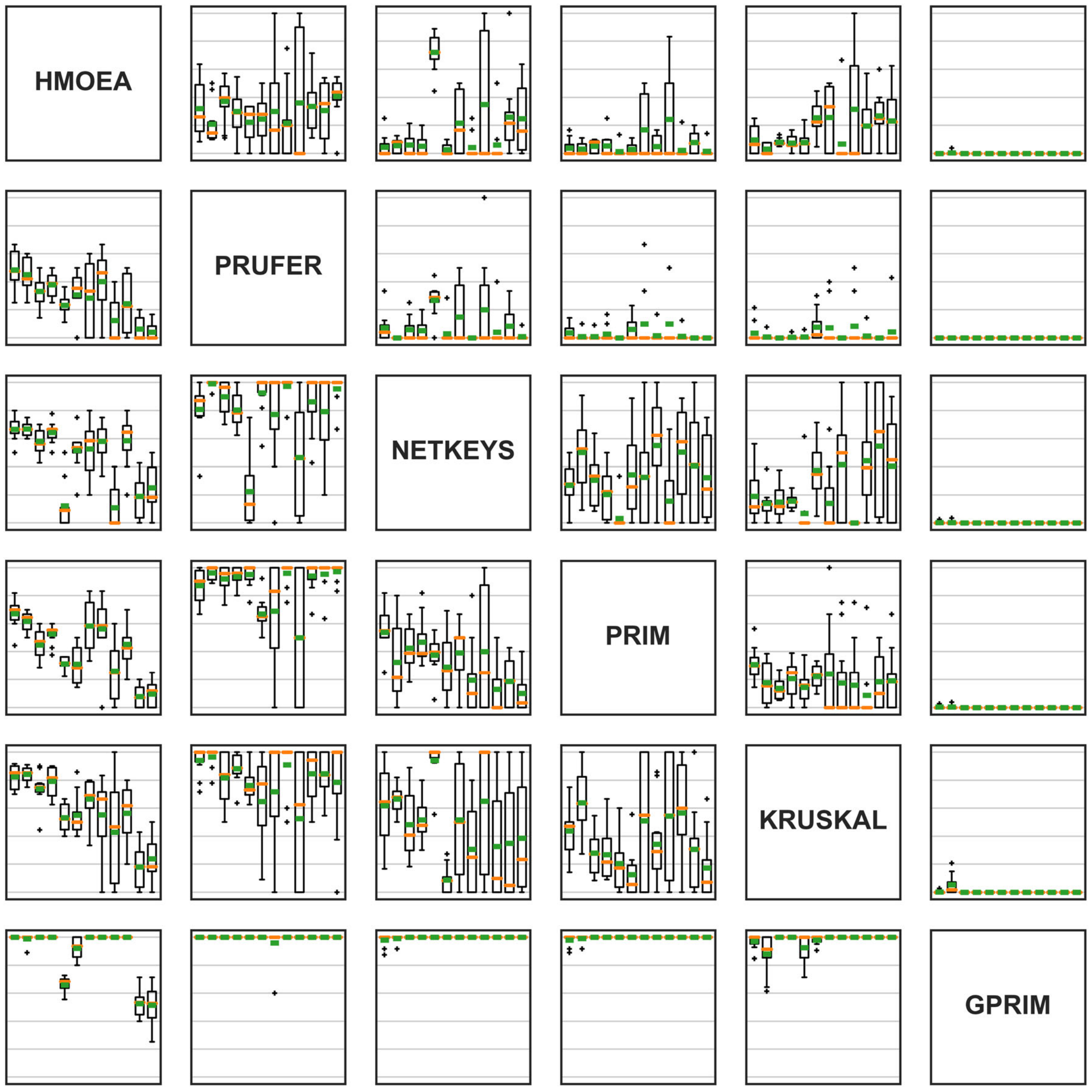


Fig. 9 Box plots for C-metric. The rectangle at row A and column B represent $C(A, B)$. Each rectangle includes 12 box plots (left to right) corresponding to 12 instances (*NIn7* to *NIn18*). C-metric values are scaled to [0,1]

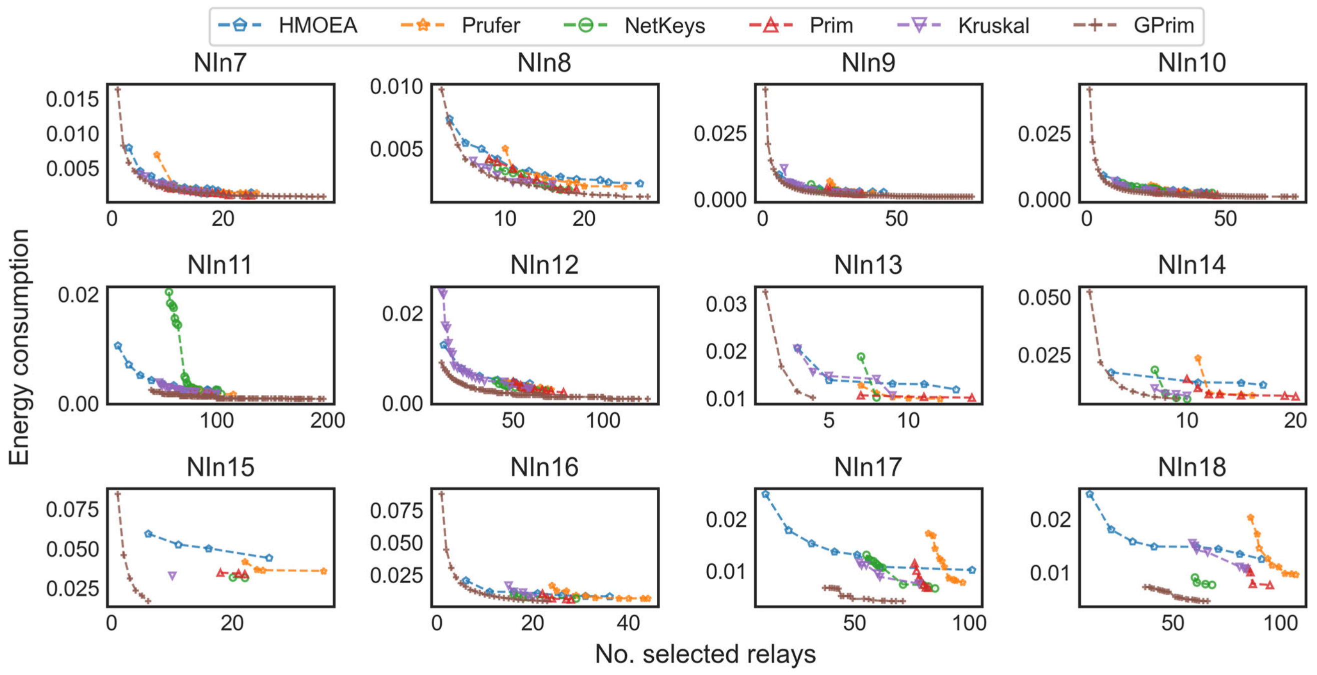


Fig. 10 The comparison of Pareto-front on test set $S2$ ($NIn7$ to $NIn12$) and $S3$ ($NIn13$ to $NIn18$)

HMOEA produces a better HV metric than GPrim in some instances ($NIn11$, $NIn17$, and $NIn18$) and worse than GPrim in the remaining instances. Note that the superiority of HMOEA in those instances comes from the better spreading of PFs regarding the relay's objective while the convergence of HMOEA is weak as mentioned in C-metric above. Prufer gives the worst result in all test cases among the remaining algorithms. The PFs obtained by NetKeys are generally better than Prim. Meanwhile, Kruskal outperforms both NetKeys and Prim. However, the results of Kruskal deteriorate in the test set $S3$.

Considering the diversity and spread of PFs, the results on Δ value (see Table 6b) shows that GPrim and HMOEA produce very diverse non-dominated networks with $\Delta = 0.840$ on average while the average diversity of the HMOEA is $\Delta = 0.845$ in which GPrim performs the best in the second test set $S2$ and HMOEA is the best in the third test set $S3$. The average Delta metric of Kruskal is $\Delta = 0.933$; Prufer is $\Delta = 0.938$; NetKeys and Prim have a very similar result $\Delta = 0.946$. Furthermore, the number of nondominated solutions obtained by GPrim (40 solutions on average) is more than four times better than others. The detail on each network instance is shown in Table 6c.

In general, the proposed approach outperforms other approaches in all evaluated aspects. HMOEA shows an advantage in the relay's objective resulting in the diversity and spread of PFs. However, as a trade-off, HMOEA provides weak performances in the convergence aspect. The poor result of the Prufer code is predictable due to problems with infeasible structures and low locality, as mentioned in Gottlieb et al. (2001); Rothlauf and Rothlauf (2006). The general phenotypic application of PrimRST is generally worse than the NetKeys approach. In comparison, Kruskal can achieve better results than NetKeys in most cases. This result is consistent with the results shown (Raidl and Julstrom 2003). However, the results of Kruskal tend to deteriorate in the third test set $S3$.

4.5.3 Evaluation on sparse solution spaces

The initialization experiment (Sect. 4.4.1) showed the relation between the max-hop constraint and the sparsity of solution space and its impact on different random-tree algorithms. Since KruskalRST tends to produce higher-diameter trees than PrimRST, it also produces more infeasible solutions when tightening the max-hop constraint. Consequently, the KruskalRST-based approach also struggles in a sparse solution space.

For example, we examine five algorithms on the same instance $NIn9$ with $\hbar = 2$ and $\hbar = 12$. The result is shown in Fig. 11. With the same settings as in 4.5.2 ($\hbar = 12$), Kruskal gives better results than both NetKeys and Prim. However, when tightening the max-hop constraint ($\hbar = 2$), the result obtained by Kruskal is worse than both NetKeys and Prim. This result is consistent with the deterioration of Kruskal in the test set $S3$ since adding more sensors in the network requires more hops to connect all sensors. Note that the best results are obtained by GPrim in both cases; the

Table 6 Performance of competing algorithms on testsets *S*2 and *S*3

Instance	HMOEA	Prufer	NetKeys	Prim	Kruskal	GPrim
(a) Hypervolume (HV)						
NIn7	0.879 ± 0.001	0.763 ± 0.013	0.774 ± 0.029	0.765 ± 0.024	0.858 ± 0.015	**0.930** ± 0.001
NIn8	0.854 ± 0.002	0.716 ± 0.021	0.771 ± 0.015	0.740 ± 0.026	0.836 ± 0.024	**0.907** ± 0.005
NIn9	0.909 ± 0.001	0.737 ± 0.015	0.788 ± 0.020	0.733 ± 0.013	0.890 ± 0.019	**0.964** ± 0.006
NIn10	0.910 ± 0.001	0.750 ± 0.011	0.806 ± 0.024	0.753 ± 0.006	0.897 ± 0.022	**0.967** ± 0.000
NIn11	**0.930** ± 0.000	0.602 ± 0.015	0.709 ± 0.025	0.609 ± 0.007	0.764 ± 0.028	0.780 ± 0.017
NIn12	0.922 ± 0.001	0.762 ± 0.011	0.797 ± 0.023	0.738 ± 0.008	0.905 ± 0.016	**0.941** ± 0.021
NIn13	0.788 ± 0.007	0.688 ± 0.018	0.735 ± 0.021	0.740 ± 0.014	0.765 ± 0.019	**0.843** ± 0.001
NIn14	0.751 ± 0.019	0.649 ± 0.022	0.733 ± 0.020	0.725 ± 0.025	0.741 ± 0.015	**0.885** ± 0.006
NIn15	0.741 ± 0.025	0.608 ± 0.038	0.621 ± 0.030	0.671 ± 0.016	0.719 ± 0.046	**0.899** ± 0.001
NIn16	0.896 ± 0.002	0.744 ± 0.010	0.810 ± 0.009	0.762 ± 0.018	0.825 ± 0.016	**0.959** ± 0.001
NIn17	**0.915** ± 0.002	0.575 ± 0.010	0.708 ± 0.032	0.604 ± 0.008	0.700 ± 0.024	0.814 ± 0.015
NIn18	**0.914** ± 0.002	0.556 ± 0.009	0.706 ± 0.018	0.577 ± 0.017	0.697 ± 0.026	0.805 ± 0.011
(b) Delta-metric (Δ)						
NIn7	0.775 ± 0.028	0.878 ± 0.019	0.892 ± 0.023	0.868 ± 0.020	0.829 ± 0.012	**0.599** ± 0.002
NIn8	0.776 ± 0.031	0.870 ± 0.030	0.873 ± 0.020	0.870 ± 0.019	0.842 ± 0.037	**0.699** ± 0.048
NIn9	0.826 ± 0.017	0.937 ± 0.017	0.941 ± 0.015	0.945 ± 0.017	0.923 ± 0.025	**0.758** ± 0.032
NIn10	0.825 ± 0.022	0.940 ± 0.016	0.944 ± 0.015	0.945 ± 0.010	0.917 ± 0.028	**0.779** ± 0.015
NIn11	0.799 ± 0.017	0.962 ± 0.012	0.948 ± 0.026	0.975 ± 0.016	0.955 ± 0.020	**0.762** ± 0.010
NIn12	0.885 ± 0.028	0.956 ± 0.009	0.974 ± 0.013	0.971 ± 0.009	0.936 ± 0.013	**0.785** ± 0.012
NIn13	**0.882** ± 0.028	0.940 ± 0.040	0.947 ± 0.025	0.953 ± 0.031	0.950 ± 0.033	0.935 ± 0.004
NIn14	**0.874** ± 0.048	0.936 ± 0.034	0.947 ± 0.020	0.921 ± 0.036	0.940 ± 0.035	0.947 ± 0.027
NIn15	**0.896** ± 0.036	0.960 ± 0.022	0.975 ± 0.033	0.969 ± 0.025	0.979 ± 0.023	0.960 ± 0.006
NIn16	**0.851** ± 0.033	0.945 ± 0.019	0.985 ± 0.012	0.967 ± 0.013	0.970 ± 0.012	0.966 ± 0.012
NIn17	**0.878** ± 0.019	0.964 ± 0.009	0.991 ± 0.022	0.983 ± 0.008	0.968 ± 0.016	0.946 ± 0.008
NIn18	**0.877** ± 0.047	0.963 ± 0.012	0.982 ± 0.016	0.987 ± 0.010	0.993 ± 0.011	0.945 ± 0.007
(c) Cardinality ($ONVG$)						
NIn7	10.30 ± 1.100	10.00 ± 1.483	9.60 ± 1.428	11.00 ± 2.236	14.00 ± 1.095	**38.00** ± 0.000
NIn8	9.50 ± 1.285	9.80 ± 1.887	10.80 ± 1.327	10.00 ± 1.183	12.90 ± 2.948	**27.70** ± 0.900
NIn9	8.50 ± 1.025	10.90 ± 2.343	11.80 ± 2.040	10.90 ± 1.221	15.20 ± 2.561	**71.80** ± 3.572
NIn10	8.80 ± 0.872	10.90 ± 1.972	11.10 ± 2.948	10.30 ± 1.487	17.70 ± 2.934	**69.60** ± 3.323
NIn11	10.40 ± 0.917	9.70 ± 1.900	17.50 ± 3.170	9.80 ± 3.400	16.30 ± 2.452	**98.60** ± 1.114
NIn12	6.80 ± 0.872	11.70 ± 1.187	11.00 ± 1.789	10.30 ± 2.193	19.20 ± 2.441	**94.60** ± 3.980
NIn13	**5.00** ± 1.000	3.10 ± 1.640	2.50 ± 0.671	2.40 ± 1.020	2.40 ± 1.020	4.00 ± 0.000
NIn14	5.10 ± 1.446	4.30 ± 1.487	4.30 ± 1.100	6.20 ± 1.778	3.20 ± 0.748	**9.10** ± 0.539
NIn15	3.80 ± 1.536	3.70 ± 1.005	1.90 ± 0.539	3.00 ± 1.414	2.00 ± 0.894	**6.00** ± 0.000
NIn16	7.00 ± 1.732	8.00 ± 1.183	5.70 ± 1.952	6.50 ± 1.910	6.00 ± 1.732	**24.30** ± 1.900
NIn17	6.20 ± 0.872	9.80 ± 1.990	10.10 ± 3.448	5.00 ± 1.897	6.80 ± 2.600	**23.20** ± 2.891
NIn18	6.50 ± 1.500	8.00 ± 2.191	10.40 ± 2.905	4.00 ± 1.844	6.20 ± 2.522	**22.30** ± 3.257

Each table shows the results on one metric and bold values indicate the best value

gap to other algorithms increases when the feasible solution space becomes smaller.

4.5.4 Evaluation on dense network

We vary the communication range from 25 to 40 to examine the impact of the density of the graph on six algorithms. Figure 12 shows the comparison of hypervolume, Delta-metric, and $ONVG$ with different communication radius on the

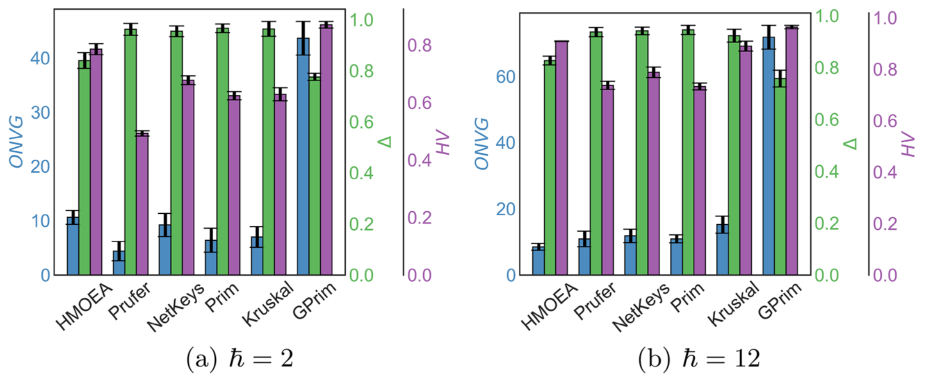


Fig. 11 Performance of competing algorithms in different $\hbar$ on network instance *NIn9*

Table 7 Complexity comparison for each algorithm

	Decoding	Crossover	Mutation	Repair
Prufer	$O(n)$	$O(n)$	$O(n)$	$O(m)$
NetKeys	$O(m \log m)$	$O(m)$	$O(m)$	–
Prim	$O(n)$	$O(n)$	$O(n)$	–
Kruskal	$O(n)$	$O(n)$	$O(n)$	–
HMOEA	$O(n^2)$	$O(n)$	$O(1)$	–
GPrim	$O(n)$	$O(n\hbar)$	$O(max(n\hbar, m))$	–

network instance *NIn9*. The ranking of the algorithms is similar to that of the previous experiments. However, we can observe the different tendencies of algorithms when increasing the radius. Prufer, NetKeys, Prim, and Kruskal deteriorate their results when increasing the communication range, while GPrim and HMOEA tend to produce better results with the dense graph.

4.5.5 Complexity evaluation

We summarize the analysis of the complexity of the five algorithms in Table 7. The integer search operators on Prufer code only take $O(n)$ in each move. However, the repair process of removing invalid edges takes $O(m)$ in the worst case. Both Prim and Kruskal only take $O(n)$ in crossover and mutation operator and require no repair process. NetKeys encodes all possible edges and requires a sorting algorithm; thus, it costs $O(m \log m)$ in decoding and $O(m)$ in both crossover and mutation. In HMOEA, the decoder uses a CBTC-based heuristic which requires $O(n^2)$ to decode a chromosome for a given relay constraint, while the reproduction costs $O(n)$ for a recombination and $O(1)$ for a mutation. The complexity of GPrim is the same as described in Sect. 3.

Table 8 shows the average running time of five algorithms in three test sets. The running time of GPrim is an acceptable tradeoff considering its various improvements. The fastest algorithm is Prim in most cases. Prufer gives the fastest result in some cases but is often much slower as infeasible structures occur more frequently. Kruskal has the same theoretical complexity as Prim, but requires higher programming constants. Meanwhile, NetKeys and, especially, HMOEA requires much computation in large and dense networks.

5 Conclusions and future works

This paper has made several contributions to reducing nodes' energy consumption and prolonging the wireless sensor networks' lifetime. Firstly, we introduced the NEBP with two objectives: minimizing the number of relay nodes and maximizing the network lifetime. We considered the limiting number of communication hops for each sensor node toward the base station. Secondly, a phenotype-based multi-objective evolutionary algorithm is developed to simultaneously tackle two objectives of the NEBP problem. We proposed local heuristics aiding the initialization, crossover, and mutations to enhance the convergence speed of the algorithm in an appropriate computation. Extensive experiments are conducted comparing the proposed method to other standard encoding methods. The simulation results indicate that the proposed method dramatically improves all measured metrics with a relatively reasonable time tradeoff.

In the scope of this paper, we assumed that all representations are integrated with the NSGA-ii structure. However, hybridizing different encoding methods with different MOO algorithms may give us additional insight into the effect of representation on each MOO algorithm. Moreover, these results show prospects for phenotype-based approaches, which may prove promising for similar problems. In future works, we plan to find more optimized algorithms for the problem and expand the model to cope with mobile sensors. In addition, we intend to consider the same problem for the heterogeneous wireless sensor networks instead of the homogeneous WSNs.

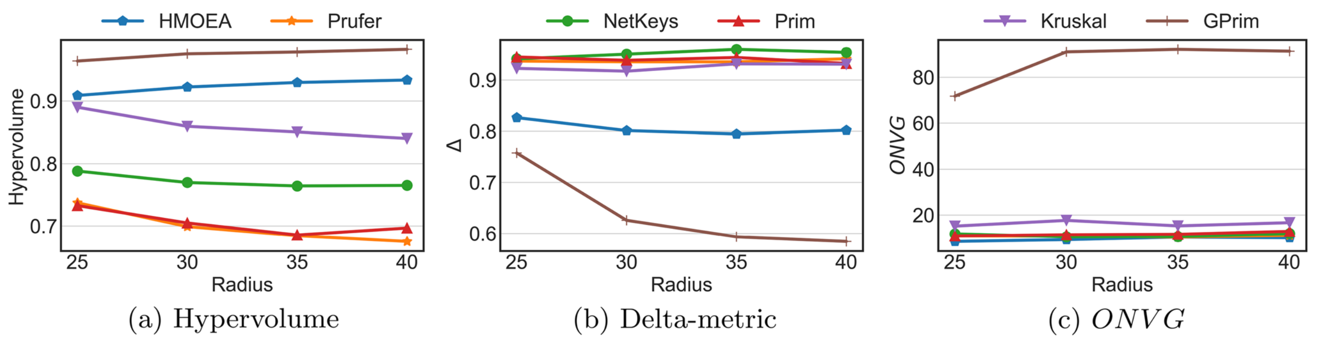


Fig. 12 Comparison of hypervolume, delta-metric and $ONVG$ with different communication radius on network instance *NIn9*

Table 8 Average algorithm running time (in s)

Instance	HMOEA-mean	Prufer	NetKeys	Prim	Kruskal	GPrim
NIn1	35.1	19.1	25.0	**17.2**	21.8	23.9
NIn2	39.2	19.3	26.6	**17.2**	21.8	24.0
NIn3	39.6	27.3	34.3	**19.8**	29.8	26.1
NIn4	39.6	28.4	34.5	**19.8**	30.0	26.0
NIn5	39.6	27.9	34.5	**19.8**	30.5	26.2
NIn6	37.1	28.0	34.3	**19.9**	30.5	25.8
NIn7	85.9	**27.9**	39.5	29.7	34.5	42.2
NIn8	36.5	**22.9**	24.0	28.9	32.1	40.6
NIn9	342.2	**63.0**	129.3	68.2	84.4	98.8
NIn10	293.8	**59.0**	114.4	70.8	81.6	98.2
NIn11	3921.3	235.8	1151.3	**195.0**	338.0	249.0
NIn12	1060.3	**131.5**	379.2	158.3	199.4	215.1
NIn13	117.9	**36.0**	55.9	43.1	44.5	76.3
NIn14	101.3	**37.5**	50.5	41.9	46.5	73.6
NIn15	365.8	110.1	154.9	**106.9**	119.8	227.8
NIn16	332.6	125.8	134.4	**106.3**	115.9	178.2
NIn17	4303.1	594.6	1386.9	**287.2**	436.8	459.8
NIn18	4427.1	639.5	1493.7	**297.5**	463.0	478.2

Funding This research was funded by the Office of Naval Research under grant number N62909-23-1-2018 for Huynh Thi Thanh Binh. Nguyen Thi Tam was funded by the Postdoctoral Scholarship Programme of Vingroup Innovation Foundation (VINIF), code VINIF. 2022.STS.06.

Data availability Enquiries about data availability should be directed to the authors.

Declarations

Conflicts of interest The authors have not disclosed any competing interests.

References

Abdulkarem M, Samsudin K, Rokhani FZ, A Rasid MF (2020) Wireless sensor network for structural health monitoring: a contemporary review of technologies, challenges, and future direction. Struct Health Monit 19(3):693–735

Audet C, Bigeon J, Cartier D, Le Digabel S, Salomon L (2020) Performance indicators in multiobjective optimization. Eur J Oper Res

Bagaa M, Chelli A, Djenouri D, Taleb T, Balasingham I, Kansanen K (2017) Optimal placement of relay nodes over limited positions in wireless sensor networks. IEEE Trans Wireless Commun 16(4):2205–2219

Bhattacharya A, Kumar A (2014) A shortest path tree based algorithm for relay placement in a wireless sensor network and its performance analysis. Comput Netw 71:48–62

Bui N (2020) A Python framework for genetic-based algorithms. Zenodo. https://doi.org/10.5281/zenodo.4159410

Chan L, Chavez KG, Rudolph H, Hourani A (2020) Hierarchical routing protocols for wireless sensor network: a compressive survey. Wireless Netw 26(5):3291–3314

Cheng R, Jin Y, Narukawa K, Sendhoff B (2015) A multiobjective evolutionary algorithm using gaussian process-based inverse modeling. IEEE Trans Evol Comput 19(6):838–856

Coello CAC, Cortés NC (2005) Solving multiobjective optimization problems using an artificial immune system. Genet Progr Evolv Mach 6(2):163–190

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

Dey N, Ashour AS, Shi F, Fong SJ, Sherratt RS (2017) Developing residential wireless sensor networks for ECG healthcare monitoring. IEEE Trans Consum Electron 63(4):442–449

Farsi M, Elhosseini MA, Badawy M, Ali HA, Eldin HZ (2019) Deployment techniques in wireless sensor networks, coverage and connectivity: A survey. Ieee Access 7:28940–28954

Florinsky I (2016) Digital terrain analysis in soil science and geology. Academic Press, Cambridge

Gawade RD, Nalbalwar SL (2016) A centralized energy efficient distance based routing protocol for wireless sensor networks. J Sens. https://doi.org/10.1155/2016/8313986

Gottlieb J, Julstrom BA, Raidl GR, Rothlauf F (2001) Prüfer numbers: A poor representation of spanning trees for evolutionary search. In: Proceedings of the genetic and evolutionary computation conference, vol 343, p 350 . Citeseer

Guleria K, Verma AK (2019) Comprehensive review for energy efficient hierarchical routing protocols on wireless sensor networks. Wireless Netw 25(3):1159–1183

Haenggi M, Puccinelli D (2005) Routing in ad hoc networks: a case for long hops. IEEE Commun Mag 43(10):93–101

Hai DT, Le Vinh T et al (2017) Novel fuzzy clustering scheme for 3D wireless sensor networks. Appl Soft Comput 54:141–149

Hanh NT, Binh HTT, Van Son N, Lan PN (2019) Minimal node placement for ensuring target coverage with network connectivity and fault tolerance constraints in wireless sensor networks. In: 2019 IEEE congress on evolutionary computation (CEC), pp 2923–2930. IEEE

Hao X, Liu J (2017) A multiagent evolutionary algorithm with direct and indirect combined representation for constraint satisfaction problems. Soft Comput 21(3):781–793

Hwang FK, Richards DS (1992) Steiner tree problems. Networks 22(1):55–89

Konstantinidis A, Yang K (2011) Multi-objective energy-efficient dense deployment in wireless sensor networks using a hybrid problem-specific moea/d. Appl Soft Comput 11(6):4117–4134

Kumar S, Agrawal R (2021) A comprehensive survey on meta-heuristic-based energy minimization routing techniques for wireless sensor network: classification and challenges. J Supercomput, pp 1–52

Lee S, Younis M, Lee M (2015) Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance. Ad Hoc Netw 24:1–19

Li Y (2001) An effective implementation of a direct spanning tree representation in gas. In: Workshops on applications of evolutionary computation, pp 11–19. Springer

Liang W, Ma C, Zheng M, Luo L (2019) Relay node placement in wireless sensor networks: from theory to practice. IEEE Trans Mob Comput 20(4):1602–1613

Lin D, Wang Q, Min W, Xu J, Zhang Z (2020) A survey on energy-efficient strategies in static wireless sensor networks. ACM Trans Sens Netw 17(1):1–48

Lloyd EL, Xue G (2006) Relay node placement in wireless sensor networks. IEEE Trans Comput 56(1):134–138

Lombardo L, Corbellini S, Parvis M, Elsayed A, Angelini E, Grassini S (2017) Wireless sensor network for distributed environmental monitoring. IEEE Trans Instrum Meas 67(5):1214–1222

Ma C, Liang W, Zheng M, Sharif H (2015) A connectivity-aware approximation algorithm for relay node placement in wireless sensor networks. IEEE Sens J 16(2):515–528

Ma C, Liang W, Zheng M (2017) Delay constrained relay node placement in two-tiered wireless sensor networks: a set-covering-based algorithm. J Netw Comput Appl 93:76–90

Ma C, Liang W, Zheng M, Yang B (2018) Relay node placement in wireless sensor networks with respect to delay and reliability requirements. IEEE Syst J 13(3):2570–2581

Misra S, Hong SD, Xue G, Tang J (2009) Constrained relay node placement in wireless sensor networks: formulation and approximations. IEEE/ACM Trans Netw 18(2):434–447

Muduli L, Mishra DP, Jana PK (2018) Application of wireless sensor network for environmental monitoring in underground coal mines: a systematic review. J Netw Comput Appl 106:48–67

Nayyar A, Le D-N, Nguyen NG (2018) Advances in swarm intelligence for optimizing problems in computer science. CRC Press, Boca Raton

Palmer CC, Kershenbaum A (1994) Representing trees in genetic algorithms. In: Proceedings of the First IEEE conference on evolutionary computation. IEEE World Congress on Computational Intelligence, pp 379–384. IEEE

Prakash VP, Patvardhan C, Srivastav A (2020) A novel hybrid multi-objective evolutionary algorithm for the bi-objective minimum diameter-cost spanning tree (bi-mdcst) problem. Eng Appl Artif Intell 87:103237

Priyadarshi R, Gupta B, Anurag A (2020) Deployment techniques in wireless sensor networks: a survey, classification, challenges, and future research issues. J Supercomput 76(9):7333–7373

Prüfer H (1918) Neuer beweis eines satzes über permutationen. Arch Math Phys 27(1918):742–744

Pule M, Yahya A, Chuma J (2017) Wireless sensor networks: a survey on monitoring water quality. J Appl Res Technol 15(6):562–570

Raidl GR (2000) An efficient evolutionary algorithm for the degree-constrained minimum spanning tree problem. In: Proceedings of the 2000 congress on evolutionary computation. CEC00 (Cat. No. 00TH8512), vol 1, pp 104–111. IEEE

Raidl GR, Julstrom BA (2003) Edge sets: an effective evolutionary coding of spanning trees. IEEE Trans Evol Comput 7(3):225–239

Ramson SJ, Moni DJ (2017) Applications of wireless sensor networks-a survey. In: 2017 international conference on innovations in electrical, electronics, instrumentation and media technology (ICEEIMT), pp 325–329. IEEE

Rashid B, Rehmani MH (2016) Applications of wireless sensor networks for urban areas: a survey. J Netw Comput Appl 60:192–219

Riquelme N, Von Lücken C, Baran B (2015) Performance metrics in multi-objective optimization. In: 2015 Latin American computing conference (CLEI), pp 1–11. IEEE

Rothlauf F, Rothlauf F (2006) Representations for genetic and evolutionary algorithms. Springer, Berlin

Rothlauf F, Goldberg DE, Heinzl A (2002) Network random keys-a tree representation scheme for genetic and evolutionary algorithms. Evol Comput 10(1):75–97

Schott JR (1995) Fault tolerant design using single and multicriteria genetic algorithm optimization. Technical report, Air Force Inst of Tech Wright-Patterson AFB OH

Sheikhi H, Hoseini M, Sabaei M (2021) k-connected relay node deployment in heterogeneous wireless sensor networks. Wireless Pers Commun 120(4):3277–3292

Singh S, Malik A, Singh PK et al (2021) A threshold-based energy efficient military surveillance system using heterogeneous wireless sensor networks. Soft Comput, 1–14

Tam NT, Hai DT et al (2018) Improving lifetime and network connections of 3D wireless sensor networks based on fuzzy clustering and particle swarm optimization. Wireless Netw 24(5):1477–1490

Tam NT, Binh HTT, Dat VT, Lan PN et al (2020) Towards optimal wireless sensor network lifetime in three dimensional terrains using relay placement metaheuristics. Knowl-Based Syst 206:106407

Tam NT, Hung TH, Binh HTT et al (2021) A decomposition-based multi-objective optimization approach for balancing the energy consumption of wireless sensor networks. Appl Soft Comput 107:107365

Tam NT, Dat VT, Lan PN, Binh HTT, Swami A et al (2021) Multifactorial evolutionary optimization to maximize lifetime of wireless sensor network. Inf Sci 576:355–373

Van Der Walt S, Colbert SC, Varoquaux G (2011) The numpy array: a structure for efficient numerical computation. Comput Sci Eng 13(2):22–30

Verma A, Ranga V, Angra S (2015) Relay node placement techniques in wireless sensor networks. In: 2015 international conference on green computing and internet of things (ICGCIoT), pp 1384–1389. IEEE

Wu Y, Liu W (2013) Routing protocol based on genetic algorithm for energy harvesting-wireless sensor networks. IET Wirel Sens Syst 3(2):112–118

Yetgin H, Cheung KTK, El-Hajjar M, Hanzo LH (2017) A survey of network lifetime maximization techniques in wireless sensor networks. IEEE Commun Surv Tutor 19(2):828–854

Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans Evol Comput 3(4):257–271

Zitzler E, Laumanns M, Thiele L (2001) Spea2: Improving the strength pareto evolutionary algorithm. TIK-report **103**

Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms-a comparative case study. In: International conference on parallel problem solving from nature, pp 292–301. Springer