Applied Soft Computing 107 (2021) 107365

Contents lists available at ScienceDirect

Applied Soft Computing

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

A decomposition-based multi-objective optimization approach for balancing the energy consumption of wireless sensor networks

Nguyen Thi Tam [a,b], Tran Huy Hung [b], Huynh Thi Thanh Binh [b,*], Le Trong Vinh [a]

[a] University of Science, Vietnam National University, Vietnam
[b] Hanoi University of Science and Technology, Vietnam

ARTICLE INFO

Article history:
Received 5 August 2020
Received in revised form 4 March 2021
Accepted 25 March 2021
Available online 8 April 2021

Keywords:
Wireless sensor networks
Network lifetime
Multi-objective evolutionary algorithm

ABSTRACT

Wireless sensor networks consist of many sensor nodes with limited resources and computing capability. Thus, managing energy consumption to prolong network lifetime is a critical issue. Several approaches have been proposed to extend the network lifetime, one of which involves deploying relay nodes to transfer data from sensors to the base station. However, the limited number of relay nodes is a challenge that often goes overlooked. This paper examines the problem of optimizing the network lifetime and the number of relay nodes in three-dimensional terrains. A novel algorithm called MOEA/D-LS is proposed with the aim of obtaining a better tradeoff between two objectives. The algorithm is a hybridization between multiobjective evolutionary algorithm based on decomposition, and a special local search to optimize the former's subproblems. Simulation results on 3D datasets show that the proposed algorithm has a significantly better performance compared with existing algorithms on all measured metrics.

1. Introduction

Wireless sensor networks (WSNs) are self-organized networks consisting of sensor nodes capable of sensing, processing, and wireless communication. These sensor nodes are generally deployed within an area of interest, and are extensively used for performing monitoring and surveillance tasks [1,2]. Depending on the application, WSNs can be designed to optimize different aspects. A range of surveys have been dedicated to different objective research domains in WSNs, such as node deployment, routing [3,4], localization [5], QoS guarantees [6], network virtualization [7], data collection [8], energy efficiency [9], and congestion control [10].

The WSNs lifetime is the time from network initialization to the instant when the network considered non-functional. Because sensor nodes have limited power and can be difficult to recharge after deployment, an energy-efficient mechanism is needed to reduce power consumption while meeting the coverage requirement. Lifetime can be indirectly optimized through energy consumption, which in turn depends on many factors, such as the receiving and transmission rate, the data sensing rate, and the nodes' placement. The wireless sensor network placement problem aims to determine nodes' positions and inter-node distance for sensor nodes to achieve coverage, connectivity, and energy efficiency.

There are three types of nodes in WSNs: sensor nodes, base stations (sinks), and relay nodes. One category of research on sensor networks has focused on base station relocation methods for maximizing the network lifetime [11–14]. This approach is based on the observation that stationary base stations establish central locations where communication activities are concentrated. This can lead to depletion of sensor battery in close proximity of the base station. Relocating the base station can thus help distribute energy consumption more evenly. Another category of research has focused on the placement of sensor nodes before deployment, with the purpose of ensuring high coverage as well as resilience to failure [15,16]. The last category is to make use of multi-hop communication for reducing overall energy consumption. A major drawback of this method is that some nodes can be overloaded and drain out their energy more quickly. Many studies have been conducted to address this problem, in which a popular approach is to deploy special relay nodes to spread out their load [17–21].

In this paper, instead of solving the single objective problems with various constraints found in previous works, we formulate a multi-objective optimization problem to obtain a better trade off among objectives.

Overview of our contributions are as follows:

- We formulate the maximum network lifetime problem in three-dimensional terrains as a multi-objective optimization problem (MOP), denoted MOO-ORP3D. The first objective

* Corresponding author.
E-mail addresses: tamnt@vnu.edu.vn (N.T. Tam), tranhuyhung1998@gmail.com (T.H. Hung), binhht@soict.hust.edu.vn (H.T.T. Binh), vinhlt@vnu.edu.vn (L.T. Vinh).

https://doi.org/10.1016/j.asoc.2021.107365
1568-4946/

is to minimize the maximum node energy consumption to prolong the network lifetime. The second objective is to minimize the number of used relay nodes.

- Inspired by the benefits of the MOEA/D framework, a novel algorithm called MOEA/D-LS is applied to the MOO-ORP3D problem. The novelties of the proposed MOEA/D-LS can be summarized as follows: *(i)* according to the problem-specific characteristics, encoding and decoding methodologies are developed to represent the solution space. Several definitions are proposed to implement objective normalization. An energy-saving procedure is designed to reduce energy consumption. Our encoding method contains the necessary information on the problem: the relay nodes used and the connection between relay nodes and sensor nodes; *(ii)* genetic operators also create the feasible children who completely inherit parents' characteristics; *(iii)* neighborhood relationship is developed to implement information sharing among subproblems to maintain the population diversity; *(iv)* a local search method is proposed to reduce the number of relays used and the maximum node energy consumption.
- The proposed algorithm is validated against existing methods including a single-objective algorithm (FCLS [22]) and the other MOEAs: the original MOEA/D [23], non-dominated sorting genetic algorithm II (NSGA-II) [24], strength Pareto evolutionary algorithm (SPEA2) [25], and multi-objective differential evolution (MODE) [26] using four commonly used metrics. Statistical tests are performed to further validate our findings. The experimental results show that MOEA/D-LS performs best in almost all datasets. In addition, we also analyze the convergence properties of MOEA/D-LS.

The rest of this paper is organized as follows. In Section 2 we summarize the related works. Section 3 describes the MOO-ORP3D. The framework of the proposed MOEA/D-LS for MOO-ORP3D is presented in Section 4. The details of our experiments is reported in Section 6. Finally, the paper is concluded in Section 7.

2. Related works

This section reviews a number of prior works on relay node placement, including single-objective problems and multi-objective problems. Many papers have studied single-objective relay node placement. The authors in [27] aimed to place a minimum number of relay nodes into a wireless sensor network while meeting connectivity and survivability requirements. They proposed approximation algorithms to solve this problem. In [28], the authors looked into a more realistic model that took into account physical constraints such as channel capacity, signal strength, and network topology. Two approximation algorithms based on independent sets and hitting sets were proposed to solve their problem. A novel energy-efficient optimal relay nodes deployment strategy is proposed in [29]. They proposed an algorithm based on artificial bee colonies to extend the lifetime by optimizing the network parameters.

Among various network performance metrics, lifetime can be considered a critical goal in network deployment. Network lifetime has been tackled at various levels: design, operation, and deployment. In [9], the authors provided a critical appraisal of state-of-the-art network lifetime maximization techniques. One of them involved using relay nodes for delivering the data from sensor nodes to base stations. This can, in particular, reduce the energy consumption of far-away sensor nodes. Inspired by this approach, some authors have studied different variations of the problem [17–19,21,30]. In [17], the issues of relay node placement and flow allocation (RNP-FA) was jointly formulated into an NP-hard integer nonlinear programming problem. They proposed a novel heuristic scheme in 3 dimensional (3D) architectures to solve RNP-FA. Their proposed algorithms attempted to increase the network lifetime by iteratively moving relay nodes to suitable locations.

Senel et al. [18] worked on establishing connected topologies in WSNs by deploying relay nodes. They proposed novel heuristics based on minimum spanning tree triangulation and Delaunay triangulation.

In [19,20,30], the authors considered the problem of deploying relay nodes to prolong network lifetime of wireless underground sensor networks under load balancing constraints. They proposed two-phase algorithms [19,30] and single-phase algorithms [20] to solve this problem. However, these works face several issues, namely: (i) the problem models do not consider the effect of the terrains, (ii) they only address the network lifetime without optimizing the number of deployed relay nodes.

WSN design is a relatively complex task, under significant influence from various performance parameters, namely quality, cost, and efficiency of real-life sensor applications. The most critical parameter is typically chosen as the optimization objective, while the rest are modeled as constraints. However, practical applications may need to optimize parameters simultaneously. Therefore, multi-objective optimization offers a natural adaptation for solving the problem. Very often, no single solution can optimize all the objectives in a MOP because these objectives are in conflict. Hence Pareto optimal solutions, which characterize optimal trade-offs among these objectives, are of practical interest. Most algorithms concerning MOP seeks an approximation of this set of Pareto-optimal or non-dominated solutions, also known as the Pareto-efficient set or Pareto set (PS) in the objective function space [31,32]. From these solutions, decision-maker can select from a diverse range of design options.

For instance, Lanza-Gutierrez et al. [33] sought to efficiently deploy energy-harvesting relay nodes in WSNs. They considered three conflicting objectives in the NP-hard Relay Node Placement Problem (RNPP): average energy cost, average sensitivity area, and network reliability. Six different multiobjective metaheuristics were proposed.

Xu et al. in [34] studied the coverage control optimization problem in WSN as a multi-objective optimization problem with three objectives, including energy consumption, coverage rate and equilibrium of energy consumption. They used two reproduction operators based on the Genetic Algorithm and Differential Evolution in MOEA/D to preserve quality individuals in each generation. The problem-specific MOEA/D for barrier coverage was presented by Zhang et al. in [35]. The authors introduced three objectives to minimize (i) total power consumption while satisfying full coverage; (ii) number of active sensor nodes to enhance reliability, and (iii) active sensor's maximum sensing range to maintain fairness. The problem is referred to as the trade-off barrier coverage problem. A multi-objective evolutionary algorithm called PS-MOEA/D is proposed to obtain the near-optimal trade-off among three objectives.

Previous works on WSN lifetime have focused on deploying relay nodes anywhere on the terrain. However, in practice, there might be some physical constraints on the placement of relay nodes. Moreover, deploying additional nodes carries intrinsic costs that have often been ignored. For these reasons, this paper considers the relay node placement for wireless sensor networks in three-dimensional terrains as a multi-objective optimization problem with two objectives: minimizing the energy consumption to prolong the network lifetime, and minimizing the number of deployed relay nodes.

Various heuristic and meta-heuristic methods have been proposed to solve similar multi-objective problems, including evolutionary algorithms. For example, SPEA2, introduced in [25], is a genetic algorithm that combines the original's strength measure

Table 1
Problem notations.

S	Set of sensors $\{s_1, s_2, \ldots, s_n\}$
L	Set of possible relay nodes $\{l_1, l_2, \ldots, l_m\}$
n	Number of sensors
m	Number of possible relay nodes
d_{ij}	3D distance between sensor s_i and relay node located l_j
et_{ij}	Energy consumption of sending data
er_j	Energy consumption of receiving, gathering and transmitting data
e_{ini}	The initial energy of a sensor and a relay
d_{jtoBS}	3D distance from l_j to base station
num_j	Number of sensor nodes connected to l_j
r_c	Communication radius of a sensor
r'_c	Communication radius of a relay

with a new truncation technique in environmental selection. Besides, there is NSGA-II [24], utilizing a non-dominated sorting algorithm and crowding distance metric. Another approach based on differential evolution and NSGA-II's multi-objective sorting method is MODE from Xue et al. [26]. Finally, we mentioned the well-known MOEA/D [23], which uses multiple weight vectors to scalarize a multi-objective problem and applies genetic operators on "neighboring" solutions.

Inspired by the multi-objective evolutionary algorithm based on decomposition, we propose a hybrid algorithm between MOEA/D and local search to solve the problem, called MOEA/D-LS. The algorithm is capable of solving an arbitrary number of scalar problems (shifting to one of two objectives) at the same time.

3. Problem statement and description

3.1. System model and assumptions

In this section, we describe the lifetime optimization problem in WSNs and its related definitions. Table 1 lists the notations used in our formulation. We assume that the sink node has unlimited energy and each sensor node has the same physical structure, meaning their initial energy, communication radius, and computing power are the same. We also assume that the sink and each sensor node are aware of their own positional information and neighboring nodes. A given node can communicate with another node iff the distance between them is within their communication radius.

We formally define MOO-ORP3D as follows:

Definition 1 (*MOO-ORP3D*). Given a wireless sensor network (WSN) consisting of a sensor set $S = \{s_1, s_2, \ldots, s_n\}$, $s_i = (x_i^s, y_i^s, h_i^s)$, and set $L = \{l_1, l_2, \ldots, l_m\}$, $l_j = (x_j^l, y_j^l, h_j^l)$ of possible relay positions. Denote (x_i^s, y_i^s, h_i^s) and (x_j^l, y_j^l, h_j^l) are the 2D coordinates and heights of s_i and l_j, respectively. The goal is to select $Z \in L$ positions at which relay nodes are deployed such that the maximum node energy consumption is minimized.

The energy consumption for each sensor node to send k bit data to l_j can be calculated as in Eq. (1).

$$et_{ij} = k \times (\epsilon_{elec} + \epsilon_{fs} \times d_{ij}^2) \tag{1}$$

The energy consumption for each relay node to receive, gather and transmit data from num_j sensor nodes to the base station is given in Eq. (2).

$$er_j = k \times [num_j \times (\epsilon_{elec} + \epsilon_{DA}) + \epsilon_{mp} \times d_{jtoBS}^4] \tag{2}$$

where ϵ_{elec} is the energy consumed by the transceiver circuit or receive circuit. This parameter is commonly set at $50nJ/bit$. ϵ_{fs} and ϵ_{mp} are the energy expenditures for transmitting k-bit data to achieve and acceptable bit error rate for the free space model and the multipath fading model, respectively. The values for these parameters are 10 pJ/bit/m^2 and 0.0013 pJ/bit/m^4, respectively. ϵ_{DA} is the energy consumed for aggregating data, set at 5 pJ/bit [36–39].

3.2. MOO-ORP3D formulation

Input

- A WSN as in Definition Definition 1.
- A connectivity matrix $C = (c_{ij})_{n\times m}$ between sensors and relay positions.

$$c_{ij} = \begin{cases} 1 \text{ if } d_{ij} \le \frac{r_c + r'_c}{2}, \\ 0 \text{ otherwise.} \end{cases} \tag{3}$$

Output

- $A = (a_{ij})_{n\times m}$ is a set of decision variables where $a_{ij} = 1$ iff s_i is assigned to l_j
- $Z = (z_j)_{m\times 1}$ is a set of decision variables where $z_j = 1$ iff an relay node is deployed at l_j, and $z_j = 0$ otherwise.

Constraints

- Each sensor node should be relayed by exactly one relay node

$$\sum_{j=1}^{m} a_{ij} = 1, \ \forall i = 1, \ldots, n. \tag{4}$$

- Sensors can only connect to relays within their communication range

$$a_{ij} \le c_{ij}, \forall i = 1, \ldots, n; j = 1, \ldots, m. \tag{5}$$

- The total energy consumed by each sensor or each relay cannot exceed their initial energy e_{ini}.

$$\sum_{j=1}^{m} a_{ij} * et_{ij} \le e_{ini}, \ \forall i = 1, \ldots, n. \tag{6}$$

$$er_j \le e_{ini}, \ \forall j = 1, \ldots, m. \tag{7}$$

Objectives

- Minimize the number of used relay nodes

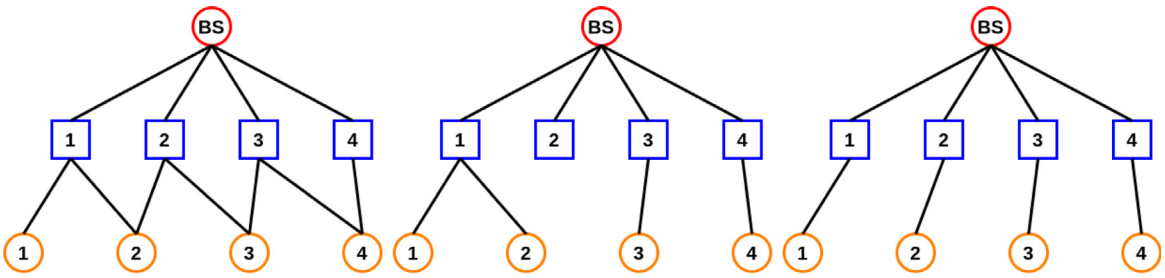
$$f_1 = \sum_{j=1}^{m} z_j \to min. \tag{8}$$

- Minimize the maximum node energy consumption

$$f_2 = \max_{i=1,\ldots,n; j=1,\ldots,m} (et_{ij}, er_j) \to min. \tag{9}$$

These objectives tend to conflict with each other and an improvement in one objective may lead to the deterioration of the other.

An example of conflict among the objectives is shown in 1. Fig. 1(a) illustrates a WSN with four sensors and four possible locations to deploy relays. We assume that if c_{ij} is 1, then d_{ij} is 1, *for* $1 \le i, j \le 4$. We also assume that $d_{jtoBS} = 1$, *for* $1 \le j \le 4$. In the first solution (Fig. 1(b)), the number of used relays is $F_1^1 = 3$, and the maximum energy consumption F_2^1 is $k \times [2 \times (\epsilon_{elec} + \epsilon_{DA}) + \epsilon_{mp}]$. In the second solution (Fig. 1(c)), the number of used relays is $F_1^2 = 4$ and maximum energy consumption F_2^2 is $k \times [\epsilon_{elec} + \epsilon_{DA} + \epsilon_{mp}]$. Thus, we have that $F_1^1 < F_1^2$ yet $F_2^1 > F_2^2$. These solutions showcased the tradeoff we intend to explore and optimize on.



(a) A WSN with 4 sensors and 4 possible locations of relays (b) A solution of MOO-ORP3D with 4 sensor and 3 relays (c) A solution of MOO-ORP3D with 4 sensors and 4 relays

Fig. 1. An example to illustrate of conflict among the objectives.

4. Multi-objective evolutionary algorithm based on decomposition for MOO-ORP3D

4.1. Objective normalization

Objective normalization has proven to be effective for MOEAs to solve multi-objective optimization problems (MOPs) with disparately scaled objectives. In this paper, we adopt a simple normalization method that transforms each objective according to the following form to solve MOPs with disparately scaled objectives:

$$\overline{f}_i(\mathbf{x}) = \frac{f_i(\mathbf{x})}{t_i^{nad}} \tag{10}$$

where $t_i^{nad} = max\{f_i(\mathbf{x})\}$ is the upper bound of ith objective of the Pareto optimal set. Additionally, all t_i^{nad} values are combined into a single point called *Nadir point*. The method to compute the Nadir point of this problem will be detailed in section below.

In this way, two objectives in Eqs. (8) and (9) can be rewritten as:

$$minimize\ \overline{F}(\mathbf{x}) = (\overline{f_1}(\mathbf{x}), \overline{f_2}(\mathbf{x})) \tag{11}$$

All objective functions after normalization will be bounded by $0 \leq \overline{f}_i(\mathbf{x}) \leq 1, i = 1, 2$.

4.2. Decomposition approach

We consider a convex combination of the different objectives. Let $\lambda = (\lambda_1, \lambda_2)^T$ be a weight vector where $0 \leq \lambda_1, \lambda_2 \leq 1$ and $\lambda_1 + \lambda_2 = 1$. The optimal solution satisfies the following scalar optimization problem:

$$minimize\ \lambda_1 * \overline{f_1}(\mathbf{x}) + \lambda_2 * \overline{f_2}(\mathbf{x}) \tag{12}$$

MOEA/D [23] initially decomposes the MOPs into N single-objective subproblems. There are several decomposition techniques, such as the weighted sum approach [40], Tchebycheff approach [40], boundary intersection approach [41], ...In this paper, we use the Tchebycheff approach to decompose the MOP into a number of sub-problems. Let $\boldsymbol{\lambda}^{\mathbf{1}}, \boldsymbol{\lambda}^{\mathbf{2}}, \ldots, \boldsymbol{\lambda}^{\mathbf{N}}$ be a set of weighted vectors for each subproblem, $\boldsymbol{\lambda}^{\mathbf{j}} = (\lambda_1^j, \lambda_2^j)^T$, *for* $j = 1, \ldots, N$. The jth single-objective optimization subproblem is defined as:

$$minimize\ g^{te}(X|\lambda^{\mathbf{j}}, t^{nad}) = \max_{1 \leq i \leq 2}\{\lambda_i^j |\overline{f}_i(\mathbf{x}) - \overline{z}_i^*|\} \tag{13}$$

Eq. (13) introduces a new factor — the reference point z^*. This point stores the best objective value achieved during the search process up to the current time. For instance, if the set $Sl = \{\mathbf{x}^{\mathbf{1}}, \mathbf{x}^{\mathbf{2}}, \ldots, \mathbf{x}^{\mathbf{K}}\}$ consists of all solutions found up to time t, then $\mathbf{z}^* = (z_1^*, z_2^*)$ in which $z_i^* = min_{1 \leq k \leq K}\{f_i(\mathbf{x}^{\mathbf{k}})\}$ for $i = 1, 2$.

$\mathbf{z}^*$ can be used in place of each subproblem's current optimal objective value. Thus, the Tchebycheff approach uses it as a base to estimate the "distance" between an individual and the optimal solutions. $\mathbf{z}^*$ is also normalized to be compatible with $\overline{f}_i(\mathbf{x})$.

4.3. Generating weight vectors

According to the MOEA/D-LS framework, N weight vectors corresponding to subproblems should be evenly spread. Moreover, as MOO-ORP3D has two objectives, each weight vector is in the form $\lambda = (\lambda_1, \lambda_2)$. For these reasons, we propose a simple formulation for λ:

$$\lambda^{\mathbf{i}} = (\frac{i-1}{N-1}, 1 - \frac{i-1}{N-1}) \tag{14}$$

The equation guarantees that corresponding elements of two consecutive vectors $\lambda^{\mathbf{i}}$ and $\lambda^{\mathbf{i+1}}$ always have the same difference $\frac{1}{N-1}$. To illustrate, if $N = 11$, the resulting weights would be:

$$\Lambda = \{(0.0, 1.0); (0.1, 0.9); (0.2, 0.8); ...; (1.0, 0.0)\}$$

Sub-problems determined by these weight vectors are equally close with each other, thus improving the similarity between adjacent solutions.

4.4. MOEA/D and MOEA/D-LS for MOO-ORP3D

We shall first describe the general framework and initial settings for MOEA/D-LS. At each iteration, consider the population with N individuals $\mathbf{x}^{\mathbf{1}}, \mathbf{x}^{\mathbf{2}}, \ldots, \mathbf{x}^{\mathbf{N}}$ corresponding to N subproblems. Additionally, the following parameters are maintained:

- $\mathbf{z}^* = (z_1^*, z_2^*)$: The reference points used in Tchebycheff equation
- *EP*: An external population containing all non-dominated solutions found during the search. It always forms a Pareto Front of the best solutions after each iteration.

Below is a detailed description of MOEA/D-LS. The primary difference to the original MOEA/D is at step 2.2 (detailed in Section 5.5). Thus we denote each algorithm as follows:

- MOEA/D: The MOEA/D algorithm similar to Algorithm 1 without step 2.2.
- MOEA/D-LS: The combination of MOEA/D and the module in Section 5.5, described in Algorithm 1.

Specific modifications over the original framework are further explained in Section 5.

5. Details of MOEA/D-LS

5.1. Solution encoding and population initialization

5.1.1. Solution encoding

Methods for encoding a solution is essential for genetic algorithms. In this paper, we use a n-dimensional vector $\mathbf{x}$ =

Algorithm 1: MOEA/D-LS framework

Input : Problem description;
N: population size and number of subproblems;
T: neighborhood size;
N weight vectors: $(\lambda_1^1, \lambda_2^1), (\lambda_1^2, \lambda_2^2), ..., (\lambda_1^N, \lambda_2^N)$
Output: EP: the external population
Step 1) Initialization
begin
1.1) Find the Nadir point $t^{nad} = (t_1^{nad}, t_2^{nad})$ (section 5.2);
1.2) Set $EP = \emptyset$;
1.3) Generate N weight vectors (section 4.3);
1.4) Build neighborhood B: Calculate the Euclidean distance between pair of weight vectors and find the T closest weight vectors to each weight vector ;
1.5) Generate a random initial population $P_0 = \mathbf{x^1}, \mathbf{x^2}, ..., \mathbf{x^N}$;
1.6) Initialize reference point $\mathbf{z}^*$ (section 5.3);
Step 2) Update
begin
for i =1 to N **do**
2.1) Genetic operators: Generate a new solution $\mathbf{y}$ using the genetic operators (section 5.4);
2.2) Improvement: Apply a scalar repair/improvement heuristic on $\mathbf{y}$ to produce $\mathbf{y}'$ (section 5.5, with λ^i as a parameter);
2.3) Update of the reference point z^*: Set $z_1^* = min\{z_1^*, f_1(\mathbf{y}')\}$ and $z_2^* = min\{z_2^*, f_2(\mathbf{y}')\}$;
2.4) Update neighboring solutions:
begin
foreach $j \in B_i$ **do**
if $g^{te}(y'|\lambda^j, \mathbf{z}^*) \leq g^{te}(x^j|\lambda^j, \mathbf{z}^*)$ **then**
Set $\mathbf{x^j} = \mathbf{y}'$
Step 2.5) Update EP;
Step 3) Stopping criteria

$(x_1, x_2, \ldots, x_n)$ to characterize each individual, where $x_i = j$ if and only if sensor node s_i is assigned to relay node at l_j.

Since each sensor node sends data to only one relay node, $\mathbf{x}$ can expand into the assignment matrix A. Conversely, a relay node is only deployed if at least one sensor connects to it. Hence, the vector Z of chosen locations is inferrable from $\mathbf{x}$. In short, $\mathbf{x}$ fully represents a solution for the problem. On the other hand, a feasible solution must still ensure that all sensors connect to relays within their communication range, which is not apparent in $\mathbf{x}$.

Fig. 2 represents a solution for an instance of MOO-ORP3D with 4 possible relay positions and 6 sensors. In this example, 3 relays are deployed at positions 1, 3, 4; their connections with 6 sensors are encoded into a 6-length vector $\mathbf{x} = (1, 3, 1, 3, 4, 3)$. As shown in the previous section, $x_2 = 3$ because sensor 1 connects with a relay at position 3. Moreover, we know only relay positions 1, 3 and 4 are chosen because $\mathbf{x}$ does not contain 2.

5.1.2. Initializing the population

The search starts by generating N random solutions for N subproblems. It is important that the initial individuals satisfy all problem constraints. Based on the encoding in 5.1.1, we propose a simple method to initialize a valid solution: for each sensor, pick a random relay node within its communication range to receive its data. These selected pairs form the required N-dimensional encoding vector while satisfying the connectivity constraint.

5.2. Finding the Nadir point

As mentioned in 4.1, the Nadir point represent the upper bounds of two objective values. In other words, the Nadir point of MOO-ORP3D is defined as $t^{nad} = (t_1^{nad}, t_2^{nad})$, in which t_1^{nad} is the maximum number of used relay nodes, and t_2^{nad} is the maximum energy a node (sensor or relay) can consume.

Since there are L possible relay positions and at most one node can be deployed at each position, $t_1^{nad} = |L|$. Meanwhile, t_2^{nad} is calculated as:

$$t_2^{nad} = \max_{i=1,...,n;j=1,...,m} (et_i^{max}, er_j^{max}) \tag{15}$$

where et_i^{max} is the maximum energy a sensor node s_i can consume. This value is equivalent to the most energy-consuming connection among those between s_i and relay nodes within its communication range. As a result:

$$et_i^{max} = \max_{j'=1,...,m;c_{ij'}=1} (et_{ij'}) \tag{16}$$

Similarly, er_j^{max} is the maximum energy consumption of a relay node deployed at position l_j. Because consumption depends on the number of connections, this upper bound is achieved when relay l_j is connected to all sensors within its communication range.

5.3. Initializing reference points

As defined in Section 4.2, the reference points $\mathbf{z}^*$ have to reflect the latest individuals in the population. Thus, they must be initialized according to the initial population $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^N}$.

$$z_i^* = \min_{1 \leq k \leq N} \{f_i(\mathbf{x^k})\}, \ i \in \{1, 2\}$$

At each proceeding iteration, $\mathbf{z}^*$ is updated whenever the search spawns a new individual y

$$z_i^* = \min\{z_i^*, f_i(\mathbf{y})\}, \ i \in \{1, 2\}$$

5.4. Genetic operators

MOEA/D-LS's genetic operators are based on the reproduction step from the MOEA/D framework. They are applied for each of the subproblem, on random individuals or pairs of individuals from its neighborhood to generate new solutions.

The crossover operator is performed on pairs of individuals with possibility p_c. For each pair, assuming x_1 is father and x_2 is mother, we use a custom relay-based crossover method: First, a child individual y is cloned from mother x_2. Then n sensors are grouped by the relay to which they are assigned in x_1. Each sensor group has a 50–50 chance to apply a modification to child y: If the corresponding relay of a group is l_j, all sensors in y which belong to that group will be reassigned to relay l_j.

Two factors motivated the design of this operator. First, it ensures the new solution y is valid, since original connections are inherited from mother x_2 and all modified ones also exist in father x_1. Secondly, the child completely inherit connections of the same relay (instead of disordered ones), which increases the likelihood of maintaining good connection groups.

Next, we apply a uniform mutation operator for $\mathbf{y}$, with the probability of each assignment is p_m. If a connection $\{s_i, l_j\}$ is chosen for mutation, we replace l_j with a new relay location l_k inside s_i's communication range. All appropriate candidates have equal chance of being selected for replacement. After the process, we obtain a new solution $\mathbf{y}$, which will be locally optimized in the next step.

The two genetic operators are illustrated in Fig. 3.

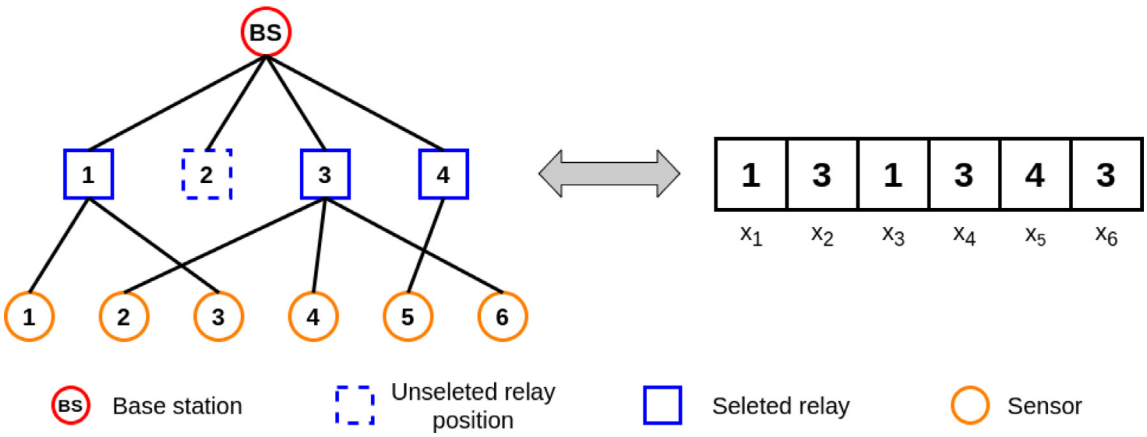


Fig. 2. An example of encoding method.

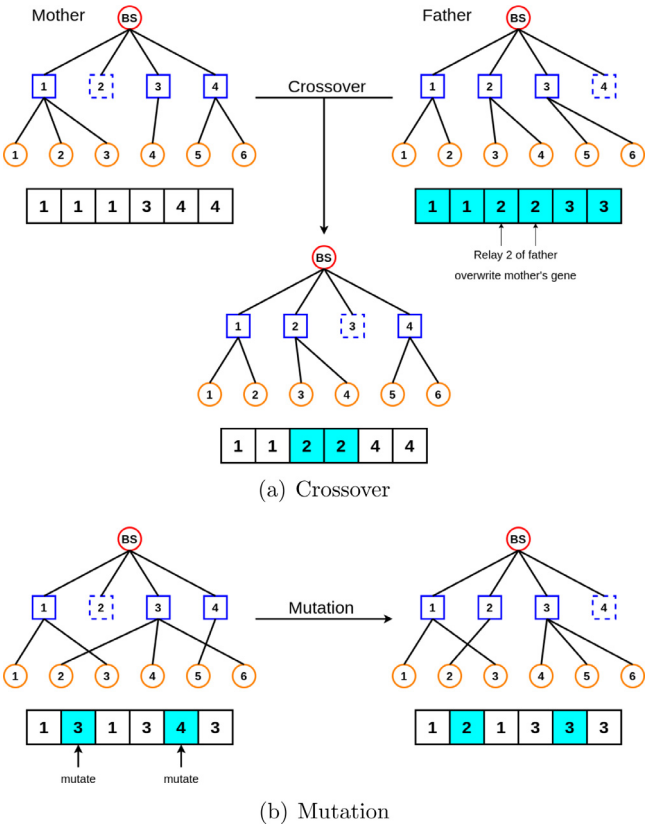


Fig. 3. Examples for genetic operations.

5.5. Local search

In the original MOEA/D, a post-search phase is used to fix invalid solutions. However, our genetic operators guarantee all generated individuals **y** are feasible. Thus instead of repairing, we propose a local search method to improve the solutions. Specifically, this phase seeks to reduce both the number of relays used and the maximum node energy consumption.

First, in order to reduce the relay count, we try to remove some relays with very few connections. For each chosen relay l_j, the algorithm searches for another chosen relay which can handle all of l_j's assigned sensors without surpassing the current maximum node energy consumption (see Fig. 4). If such a node exists, it is safe to remove l_j after reassigning its sensors to the other node. Algorithm 3 illustrates this method.

Second, we focus on the energy objective. Given an individual **y**, because **y**'s energy consumption value depends on the node which uses the most power, we refer to it as the "critical node" *crit* of **y**. Due to the energy equation and the constant values, the critical node is usually a relay node. In this case, there are two ways to improve the energy consumption:

- In order to reduce the critical node's load, find a sensor s_i assigned to l_{crit} and try to reassign it to another relay node. The alternative relay l_j must be a chosen relay and does not use as much energy as the original l_{crit} (see Fig. 5 (a)).
- If the above method does not work, try to replace l_{crit} with a substitute l_{alt}. l_{alt} is searched for in the set of unselected

relays, and must be able to handle all of l_{crit}'s current sensors with lower energy consumption. Assuming that l_alt exists, the energy objective will be improved, while the number of necessary relays remains unchanged (see Fig. 5 (b)).

We describe the above steps in Algorithm 5. It is executed until the critical node is a sensor node or no further reduction of $f_2(\mathbf{y})$ is possible. This process is detailed as a sub-module in Algorithm 4.

Combining both proposed algorithms, we acquire a final local search algorithm, detailed in Algorithm 2. It is important to note that their order also matters, since the one executed sooner will have bigger impact upon the solution. On the other hand, we know that in Algorithm 1, each new solution y is produced specially for a base ith subproblem. Thus, we use the ith corresponding weight vector to decide which sub-module is applied to y first. If the vector used for y is $\lambda^i = (\lambda_1^i, \lambda_2^i)$, the relay module has λ_1^i chance of being the first, while the energy module's probability is λ_2^i.

Algorithm 2: Local search module

Input : *sol*: A complete solution received after the previous step;
$\lambda = (\lambda_1, \lambda_2)$: The weight vector of the base subproblem;
Output: *new_sol*: The improved version of *y*
if *random* $< \lambda_1$ **then**
  new_sol ← *Relay_Count_Reduction*(*sol*); // algorithm 3
  new_sol ← *Energy_Reduction*(*sol*); // algorithm 4
else
  new_sol ← *Energy_Reduction*(*sol*); // algorithm 4
  new_sol ← *Relay_Count_Reduction*(*sol*); // algorithm 3
return *new_sol*;

Algorithm 3: Relay count reduction

Input : *S*: set of sensors;
L: set of possible relay nodes;
$C : (c_{ij})_{n\times m}$: connectivity matrix;
sol: The base solution;
Output: *new_sol*: The modified solution with reduced deployed relay count
connections ←The set of *n* sensor–relay connections in *sol*;
RNs ← $l_j \in L | Z_j(sol) = 1 \backslash \{l_{crit}\}$; // Chosen relay positions in *y* (excluding l_{crit})
foreach $l_{j^*} \in RNs$ **do**
  $children_{j^*} \leftarrow \{s_i \in S | (s_i, l_{j^*})\} \in connections$; // Sensor nodes which connect to l_{j^*} in *sol*
  foreach $l_{j^*} \in RNs$ **do**
    if $c_{ij} = 1 \forall s_i \in children_{j^*}$ **then**
      new_sol ← *sol*;
      foreach $s_i \in children_{j^*}$ **do**
        Replace connection (s_i, l_{j^*}) with (s_i, l_j) in *new_sol*;
      Unselect node l_{j^*} in *new_sol*;
      if $f_2(new\_sol) < f_2(sol)$ **then**
        sol ← *new_sol*;
        break;
return *sol*;

Algorithm 4: Energy reduction

Input : *sol*: A complete solution received after the previous step
Output: *new_sol*: The improved version of *y*
new_sol ← *sol*; *old_consumption* ← $+\infty$;
$node_{critical}$ ← The "critical node" of solution *new_sol*;
while ($f_2(new\_sol) < old\_consumption$) *and* ($node_{critical}$ *is a relay node*) **do**
  old_consumption ← $f_2(new\_sol)$;
  new_sol ← *Relay_Energy_Reduction*(*new_sol*, $node_{critical}$);
  // algorithm 5
  Update $node_{critical}$;
return *new_sol*;

Algorithm 5: One-time energy reduction

Input : *S*: set of sensors;
L: set of possible relay nodes;
$C : (c_{ij})_{n\times m}$: connectivity matrix;
sol: The base solution;
l_{crit}: The critical relay node
Output: *new_sol*: The modified solution which reduced l_{crit}'s consumption
connections ←The set of *n* sensor–relay connections in *sol*;
RNs ← $l_j \in L | Z_j(sol) = 1 \backslash \{l_{crit}\}$; // Chosen relay positions in *y* (excluding l_{crit})
$children_{crit} \leftarrow \{s_i \in S | (s_i, l_{crit})\} \in connections$; // Sensor nodes which connect to l_{crit} in *sol*
// Move one sensor connection from l_{crit} to another relay
foreach $s_i \in children_{crit}$ **do**
  foreach $l_j \in RNs$ **do**
    if $c_{ij} = 1$ **then**
      new_sol ← *sol*;
      Replace connection (s_i, l_{crit}) with (s_i, l_j) in *new_sol*;
      if $f_2(new\_sol) < f_2(sol)$ **then**
        return *new_sol*;
// Replace l_{crit} with a better energy-efficient relay node
foreach $l_j \in L \backslash RNs$ **do**
  if $c_{ij} = 1 \forall s_i \in children_{crit}$ **then**
    new_sol ← *sol*;
    Select node l_j in *new_sol*;
    foreach $s_i \in children_{crit}$ **do**
      Replace connection (s_i, l_{crit}) with (s_i, l_j) in *new_sol*;
    Unselect node l_{crit} in *new_sol*;
    if $f_2(new\_sol) < f_2(sol)$ **then**
      return *new_sol*;
return *sol*;

6. Experimental results

6.1. Experimental settings

Our test data is based on Vietnam's geographical data, as used in [39]. The dataset contains ten 3D terrain maps (T1-T10) of various morphologies. Table 2 and Fig. 6 give a brief description of these terrains. 3D terrains are defined according to the Digital Elevation Model (DEM) standard. Parameters are set as in Table 3. Algorithms are executed on a single PC (Intel Xeon x5450 CPU

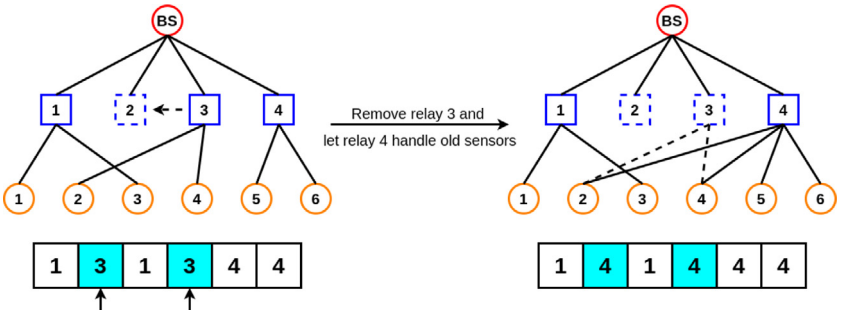
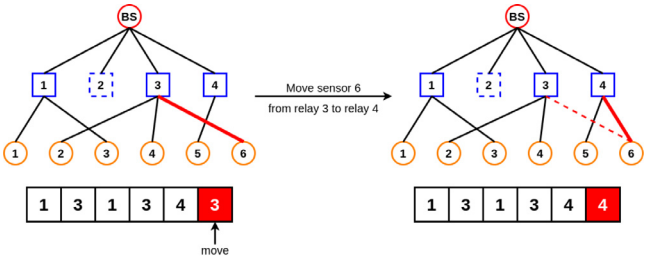
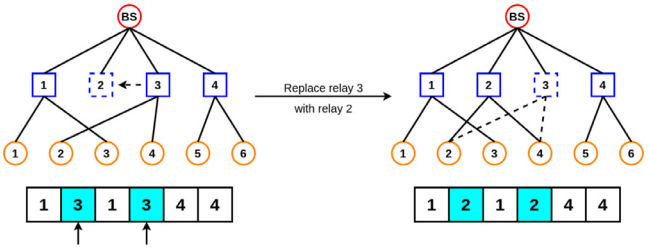


Fig. 4. Relay count reduction.



(a) Move one sensor out of critical relay



(b) Replace critical relay

Fig. 5. Relay energy reduction.

3 GHz, 16 GB RAM). All algorithms were implemented in the Python programming language. Each test result is measured over 30 independent runs.

6.2. Problem instances

Due to the fact that there are no available set of instances of the problems, ten datasets (100 instances) are created and categorized into two scenarios. Scenario 1 is a network with 40 sensor nodes and 40 relay locations an area of 200m × 200m. The scenario consists of five datasets (**Type 1s** to **Type 5s**). In scenario 2, 100 sensor nodes and 100 relay locations are distributed over an area of 500m × 500m. Datasets from **Type 1l** to **Type 5l** are included in scenario 2. Details of datasets are described as follows:

- **Type 1s** (*ga*25*s*_1 to *ga*25*s*_10) and **Type 1l** (*ga*25*l*_1 to *ga*25*l*_10): Nodes (sensor nodes and relay nodes) are generated on small terrains and large terrains from T1 to T10 according to a Gamma distribution shown in Fig. 7. The nodes' communication range is set at 25 m.
- **Type 2s** (*no*25*s*_1 to *no*25*s*_10) and **Type 2l** (*no*25*l*_1 to *no*25*l*_10): Nodes are generated on small terrains and large terrains from T1 to T10 according to Gaussian distribution shown in Fig. 7. The nodes' communication range is set at 25 m.
- **Type 3s** (*uu*25*s*_1 to *uu*25*s*_10) and **Type 3l** (*uu*25*l*_1 to *uu*25*l*_10): Similar to Type 1 and Type 2, however nodes are generated according to Uniform distribution.
- **Type 4s** (*uu*30*s*_1 to *uu*30*s*_10) and **Type 4l** (*uu*30*l*_1 to *uu*30*l*_10): Nodes are generated on small terrains and large terrains from T1 to T10 according to a Uniform distribution. The communication range is set at 30 m.
- **Type 5s** (*uu*45*s*_1 to *uu*45*s*_10) and **Type 5l** (*uu*45*l*_1 to *uu*45*l*_10): Similar to Type 4, but with a communication range of 45 m.

6.3. Evaluation metrics

In multi-objective optimization, the performance of an algorithm is determined both in terms of the convergence and the diversity of the obtained front. In this paper, we use some popular measurements as follows:

- *Coverage-metric* [24,42] The C(A,B) metric, which is usually considered as a MOEA's quality metric, evaluates the ration of the non-dominated solutions in an algorithm A's Pareto front dominated by the non-dominated solutions in an algorithm B's Pareto front, divided by the total number of non-dominated solutions obtained by algorithm A.

$$\mathcal{C}(A,B) = \frac{|x \in B | \exists y \in A : y \text{ dominates } x|}{|B|} \tag{17}$$

$\mathcal{C}(A,B)$ is not necessarily equal to $1-\mathcal{C}(B,A)$. If $\mathcal{C}(A,B)=1$, all solutions in B are dominated by at least one solution in A. $\mathcal{C}(A,B)=0$ implies that no solution in B is dominated by a solution in A.
For convenience, in this paper we define a new difference-in-coverage metric:

$$\delta(A,B) = \mathcal{C}(A,B) - \mathcal{C}(B,A) \tag{18}$$

This metric is always in the range of $[-1,1]$. $\delta(A,B) = 1$ means A completely dominates B. In the opposite case, $\delta(A,B) = -1$.
- *Δ-metric* [42,43]: the Δ value of a set of candidate solutions A is calculated as follows:

$$\Delta(A) = \frac{\sum_{j=1}^{|A|}(d_j - \bar{d})}{|A|\bar{d}} \tag{19}$$

where d_j is the distance between consecutive points and $\bar{d}$ is the averaged value of all those distances. $\Delta(A)=0$ means a uniform spread of solutions in the objective space, therefore a lower $\Delta(A)$ is preferable.
- $\mathcal{NDS}(Non-dominatedsolutions)-metric$ [42] is the number of non-dominated solutions in set A, i.e.

$$\mathcal{NDS}(A) = |A| \tag{20}$$

Table 2
Terrain morphologies.

Terrain	Location	Morphology
T1	Ho Chi Minh	City with variable building heights, no hills or rivers
T2	Vung Tau	City with mostly medium-height buildings, gentle hills, and part ocean
T3	Phu Quoc	An island with low rolling hills
T4	Dong Thap	Plains with sparse buildings, along many rivers and canals, no hills.
T5	Vinh Long	Plains with many rivers and canals, no hills.
T6	Lam Dong	Highlands area, many mountains and hills.
T7	Cao Nguyen	Highlands with high mountains and ascending hils
T8	Da Nang	City with sparse buildings and mountains bordering the sea.
T9	Ha Noi	City with many buildings and large number of lakes.
T10	Ha Long	Coastal area with many small islands of varying heights

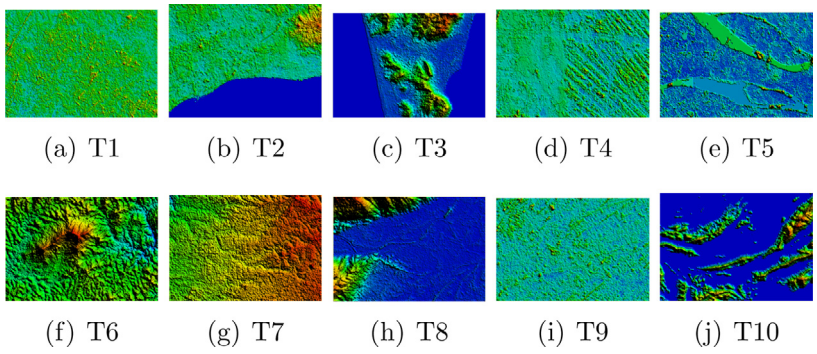


Fig. 6. Height heatmaps for terrains in the dataset.

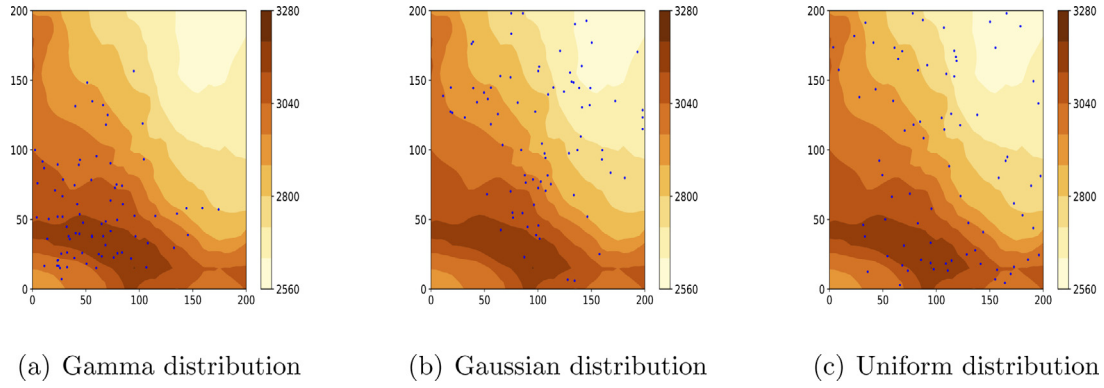


Fig. 7. The different distributions of sensors investigated in our experiments.

In these cases, it is more desirable to obtain a high number of $\mathcal{NDS}(A)$ in order to provide an adequate number of Pareto optimal choices.

- *Hypervolume metric* [43]: The hypervolume indicator is a popular metric used to evaluate both the convergence and diversity of Pareto fronts. It is the volume of the space in the objective space dominated by the front approximation and delimited from above by the Nadir point (described in Section 5.2), as shown in Fig. 8.
  We also define the improvement count $\mathcal{S}$ as the number of instances for which an algorithm outperforms the other aspect of hypervolume metric:

$$\mathcal{S}(A,B) = \sum_{i=1}^{no.instances} (HV(A,r) > HV(B,r)) \tag{21}$$

6.4. Experiment results

To evaluate the performance of MOEA/D-LS in solving the MOO-ORP3D problem, we performed the following experiments:

Table 3
Parameters.

Parameter	Value
Max number of generations	200
Population size	40
Number of subproblem	40
Neighborhood size	3
Crossover rate	0.9
Mutation rate	0.01

- Experiment 1: we compare the proposed algorithm with the original MOEA/D [23] to evaluate the effect of the improvement phase on the performance of the proposed algorithm (MOEA/D-LS).
- Experiment 2: we compare the proposed MOEA/D-LS with several multiobjective evolutionary algorithms on different metrics as mentioned in Section 6.3:
  - NSGA-II [24] selects individuals according to Pareto dominance relation and reproduces offsprings iteratively. Specifically, the new population is sorted by

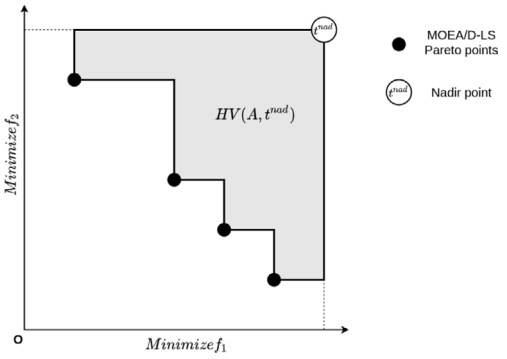


Fig. 8. Illustration of the hypervolume indicator for a 2-objective problem.

ranking according to the relationship of dominance, and the solutions are widespread by using crowding distance.

- SPEA2 [25] is an elitist MOEA. This algorithm used new archive truncation methods to guarantee the preservation of boundary solutions.
- MODE [26] is unlike some metaheuristic techniques such as genetic algorithms and evolutionary strategies, where perturbation occurs in accordance with a random quantity. MODE uses weighted differences between solution vectors to perturb the population.

Note that we are using the same representation and reproduction mechanisms (crossover, mutation, and solution selection) for MOEA/D-LS and four above MOEAs.

- Experiment 3: compare the convergence rate of MOEA/D-LS, MOEA/D to other MOEAs.
- Experiment 4: compare MOEA/D-LS to the other MOEAs on two objectives (the number of deployed relay nodes and energy consumption).
- Experiment 5: compare between MOEA/D-LS and single-objective algorithm FCLS [22].

6.4.1. Evaluate the effect of local search

Our first experiment focuses on the effectiveness of the local search phase (Section 5.5). We compare the results of MOEA/D and MOEA/D-LS. Each instance is run 10 times. The Pareto fronts of all runs are measured and averaged using Δ-metric. The same process is applied to the improvement count $\mathcal{S}$ metric, and the average hypervolume values of one instance for both algorithms are used for comparison. Additionally, for each instance, we pair Pareto fronts of all runs from MOEA/D and MOEA/D-LS and report the mean δ-metric for all pairs.

The results in Table 4 shows MOEA/D-LS greatly outperforming MOEA/D. All δ values approximately 1.00, meaning MOEA/D-LS's fronts dominate MOEA/D's counterparts in almost every case. MOEA/D-LS's Pareto fronts are better in almost all instances, for both diversity and optimal, demonstrating the effect of Algorithm 2.

Tables 5 and 7 also shows significant improvement in Pareto diversity and density. The smaller the Δ value, the more uniform the Pareto front. In Table 5, Δ decrease from 0.17 of MOEA/D to 0.13 of MOEA/D-LS for small datasets, and from 0.22 to 0.12 for large ones. Besides, the average *NDS* of MOEA/D-LS is much higher (10.5 for 200 × 200 and 21.0 for 500 × 500, respectively).

In addition, the difference in results between the two methods is higher in large scenario (500 × 500 datasets), which means the local search module has bigger impact on the shape of Pareto fronts when solving larger instances.

Table 4
Comparison δ-metric between MOEA/D-LS (A) and other MOEAs.

Datasets	δ(A, MOEA/D)	δ(A, NSGA-II)	δ(A, SPEA2)	δ(A, MODE)
200 m × 200 m	1.00	0.99	1.00	1.00
500 m × 500 m	1.00	1.00	1.00	1.00

Table 5
Δ-metric for MOEA/D-LS and other MOEAs. Italic values indicate the best result for each dataset.

Datasets		Δ(MOEA/D-LS)	Δ(MOEA/D)	Δ(NSGA-II)	Δ (SPEA2)	Δ(MODE)
200 m × 200 m	Type 1s	*0.14*	0.18	0.22	0.20	0.22
	Type 2s	*0.09*	0.14	0.14	0.15	0.21
	Type 3s	*0.16*	0.19	0.18	0.17	0.25
	Type 4s	*0.15*	0.18	0.22	0.21	0.25
	Type 5s	*0.13*	0.16	*0.13*	*0.13*	0.25
	Average	**0.13**	**0.17**	**0.18**	**0.17**	**0.24**
500 m × 500 m	Type 1l	*0.13*	0.21	0.19	0.20	0.19
	Type 2l	*0.10*	0.22	0.19	0.20	0.24
	Type 3l	*0.15*	0.25	0.19	0.21	0.24
	Type 4l	*0.06*	0.22	0.14	0.17	0.18
	Type 5l	*0.15*	0.22	0.12	0.16	0.09
	Average	**0.12**	**0.22**	**0.17**	**0.19**	**0.19**

Table 6
Comparison $\mathcal{S}$-metric between MOEA/D-LS (A) and other MOEAs.

Datasets	$\mathcal{S}$ (A, MOEA/D)	$\mathcal{S}$ (A, NSGA-II)	$\mathcal{S}$ (A, SPEA2)	$\mathcal{S}$ (A, MODE)
200 m × 200 m	50	50	50	50
500 m × 500 m	50	50	50	50

6.4.2. Comparison with other MOEAs

Table 4 shows the comparison of δ-metric ($\mathcal{C}$-metric) for different datasets. We can see that δ-metric for MOEA/D-LS outperforms other MOEAs all cases. The non-dominated solutions in MOEA/D-LS's Pareto front dominate approximately 100% of those in MOEA/D, NSGA-II, SPEA2, and MODE's Pareto front, on average.

Moreover, MOEA/D-LS gives a better Δ average with 0.13, compared to 0.17, 0.18, 0.17, and 0.24 obtained by MOEA/D, NSGA-II, SPEA2, and MODE, on terrains of size 200m × 200m. On the larger 500m × 500m terrains, the figures are 0.12, 0.22, 0.17, 0.19, and 0.19 for MOEA/D-LS, MOEA/D, NSGA-II, SPEA2, and MODE.

From Table 6, we see that the proposed algorithm also outperforms MOEA/D, NSGA-II, SPEA2, and MODE in all instances (100/100). MOEA/D-LS shows remarkable advantage in δ-metric, Δ-metric, and hypervolume metric. This further highlights MOEA/D-LS's stability from changing problem requirements (different node distribution, different sensing radius).

Table 7 compares the $\mathcal{NDS}$-metric of our proposed algorithm with other MOEAs in each scenario. Our experiments show that all $\mathcal{NDS}$ values of MOEA/D-LS are better than or equal to those of the remaining algorithms. For example, the average values of $\mathcal{NDS}$-metric of MOEA/D-LS are 10.5 and 21.0 for scenario 1 and scenario 2, respectively. These figures are 6.4 and 11.9 for NSGA-II, 6.5 and 10.8 for SPEA2, 3.2 and 3.7 for MODE, respectively. With regard to the NDS values in Table 7, MOEA/D-LS performs the best, having a clear advantage over other MOEAs on all the test cases. Specially, all the MOEA/D-LS obtain better results on Gaussian test cases rather than Uniform and Gamma.

In order to verify the effectiveness of MOEA/D-LS, statistical analysis is necessary. We again choose the hypervolume metric as the measurement value because it represents both the optimization level and diversity of Pareto fronts. In addition, results do not follow statistical distributions, so our statistics use non-parametric tests.

First, a Friedman Test is carried out to test the following null hypothesis H_0: "there are no differences between the results

Table 7
Comparison $\mathcal{NDS}$-metric between MOEA/D-LS and other MOEAs.

Datasets		MOEA/D-LS	MOEA/D	NSGA-II	SPEA2	MODE
200 m × 200 m	Type 1s	*9.4*	6.3	5.3	5.6	3.1
	Type 2s	*12.8*	8.8	8.8	8.4	4.5
	Type 3s	*8.4*	6.6	5.6	5.9	3.1
	Type 4s	*7.7*	5.9	4.4	4.9	2.4
	Type 5s	*14.5*	8.0	7.8	7.9	3.1
	Average	***10.5***	**7.1**	**6.4**	**6.5**	**3.2**
500 m × 500 m	Type 1l	*24.6*	10.0	13.4	12.1	4.0
	Type 2l	*30.8*	12.5	17.3	15.3	4.7
	Type 3l	*26.4*	11.1	15.3	13.6	4.4
	Type 4l	*18.1*	9.0	10.0	9.3	3.4
	Type 5l	*5.0*	3.7	3.3	3.5	1.7
	Average	***21.0***	**9.3**	**11.9**	**10.8**	**3.7**

Table 8
Friedman Test for MOEA/D-LS and other MOEAs.

		MOEA/D-LS	NSGA-II	SPEA2	MODE
200 m × 200 m	Mean	*0.7320*	0.6231	0.6268	0.5420
	Std. Derivation	0.0708	0.0531	0.0539	*0.0492*
	Mean Rank	*4.00*	2.30	2.70	1.00
	Number of instances	50			
	χ^2	137.40			
	df	3			
	p	0.000			
500 m × 500 m	Mean	*0.7076*	0.5367	0.5263	0.4796
	Std. Derivation	0.2089	0.1543	0.1528	*0.1432*
	Mean Rank	*4.00*	2.98	2.02	1.00
	Number of instances	50			
	χ^2	148.824			
	df	3			
	p	0.000			

of the mean hypervolume values obtained by the algorithms". Table 8 gives details of the statistic test, including the following values: The mean ranks among the group of algorithms (the higher the better), the chi-square value χ^2, degree of freedom *df* and the asymptotic significance *p*-value. *p*-values in both scenarios are approximately 0.000, which rejects H_0 (because $p < 0.05$) and indicates that there are notable differences between the algorithms' results.

The above test also pinpoints MOEA/D-LS as having the highest mean rank among four methods. As a result, we run post-hoc tests to verify MOEA/D-LS's performance. Wilcoxon Signed Rank Tests are performed on three different pairs: MOEA/D-LS-NSGA-II, MOEA/D-LS-SPEA2 and MOEA/D-LS-MODE. Additionally, multiple comparisons require a Bonferroni adjustment applied to their test results. Dividing the initial significance level 0.05 by the number of comparison tests, we acquire a new significance level of $0.05/3 = 0.017$. Summaries of the tests are detailed in Table 9. Again, MOEA/D-LS has higher results with a *p*-value of 0.000 (< 0.0017) in all three tests. In conclusion, we assert that MOEA/D-LS's performance is better than three counterparts.

6.4.3. Evaluate the convergence rate

This experiment aims to evaluate the algorithms' convergence properties. We use the optimal Pareto front of a test instance as the measurement baseline. Although the optimal fronts in these experiments are unknown, previous sections indicate MOEA/D-LS is most likely to achieve a near-optimal front. As a consequence, we use MOEA/D-LS's results as a replacement for the true optimal front, as follows:

- For each instance, we run MOEA/D-LS up to 100 generations and call the final Pareto front *base_Pareto* of that instance.
- To evaluate the performance of a method on a particular instance, we run that method for 20 generations. The achieved result is compared with the *base_Pareto*, as reflected by the "convergence ratio" (*CR*):

$$CR(A) = HV(A, t^{nad})/HV(base\_Pareto, t^{nad}) \quad (22)$$

where A is the resulting front after 20 iterations, and *base_Pareto* was defined above. $CR = 1$ means front A almost coincides with the base front (assuming that A does not surpass the optimal).

Note that this formula is only used for one method and one instance at a time. For all algorithms, each instance is run 10 times, then the mean of 10 *CR* values are calculated. Finally, the average *CR* of all instances in a dataset which corresponds to each algorithm is represented in Table 10.

Table 10 shows MOEA/D-LS has the fastest convergence rate, with a *CR* value of approximately 1.This convergence is much better than other MOEAs' partial Pareto, as NSGA-II only achieves on average $CR = 0.85$ for small instances and $CR = 0.76$ for large sets.

Finally, illustrations of the convergence process of MOEAs are presented in Figs. 9 and 10. Each front is represented by a distinct line connecting all of its solutions.

Our experiments lead to the conclusion that most MOEAs converge after around 200 generations. The only exception is MOEA/D-LS, whose fronts do not have significant improvements after 10 generations. Consequently, in these instances, we run each instance for 200 generations, and draw a Pareto front every 40 iterations (except MOEA/D-LS which is run up to only 10 first generations and drawn after 2 consecutive iterations). Due to genetic algorithm (GA)'s convergence properties, all algorithms excluding MOEA/D-LS converge gradually and improve the fronts significantly. However, MOEA/D-LS not only converges much faster in Figs. 9 and 10, but also has wider and better optimized final solutions in Fig. 11.

From these results, some important factors that contribute to the convergence of MOEA/D-LS can be observed. First of all, the effectiveness of the optimization module, which has been proved in the previous section, is the main reason behind the fast convergence rate. Since the two parts of this module focus on two objectives, both objective values of MOEA/D-LS are better than others' ones, as shown in Fig. 11. However, this also leads to conflicts because two objectives oppose each other. In Figs. 9(a) and 10(a), MOEA/D-LS can still maintain the diversity of its fronts, thanks to the selection mechanism of the original MOEA/D which preserves the best solutions for different weight vectors. The way we order two sub-modules of the local search algorithm based on weight vectors also ensures individuals which shift heavily to one objective have a better chance to extend extreme points of the Pareto front. On the other hand, the frequency of using local search can affect the convergence speed. After some experiments, we concluded that most non-optimized individuals are immediately discarded by optimized ones. Therefore, it is best to apply the local search module to all new solutions. The population size is also a noticeable factor. More subproblems mean better results but higher time complexity.

6.4.4. Evaluation on two objectives

We also define the improvement count $\mathcal{S}_1$ and $\mathcal{S}_2$ as the number of instances for which an algorithm outperforms the other on the energy consumption objective and the used relays objective, respectively.

$$\mathcal{S}_1(A, B) = \sum_{i=1}^{no.instances} (\overline{f_1^B} - \overline{f_1^A}) > 0 \quad (23)$$

$$\mathcal{S}_2(A, B) = \sum_{i=1}^{no.instances} (\overline{f_2^B} - \overline{f_2^A}) > 0 \quad (24)$$

Table 9
Wilcoxon Tests for comparisons between MOEA/D-LS and other MOEAs.

		NSGA-II		SPEA2		MODE	
		Negative	Positive	Negative	Positive	Negative	Positive
200 m × 200 m	Number of instances	50	0	50	0	50	0
	Mean Rank	25.50	0.00	25.50	0.00	25.50	0.00
	Sum of Ranks	1275	0	1275	0	1275	0
	Z	−6.514		−6.514		−6.514	
	(based on)	(positive ranks)		(positive ranks)		(positive ranks)	
	p (2-tailed)	0.000		0.000		0.000	
500 m × 500 m	Number of instances	50	0	50	0	50	0
	Mean Rank	25.50	0.00	25.50	0.00	25.50	0.00
	Sum of Ranks	1275	0	1275	0	1275	0
	Z	−6.514		−6.514		−6.514	
	(based on)	(positive ranks)		(positive ranks)		(positive ranks)	
	p (2-tailed)	0.000		0.000		0.000	

*Negative rank: former algorithm's hypervolume < MOEA/D-LS's.
*Positive rank: former algorithm's hypervolume > MOEA/D-LS's.

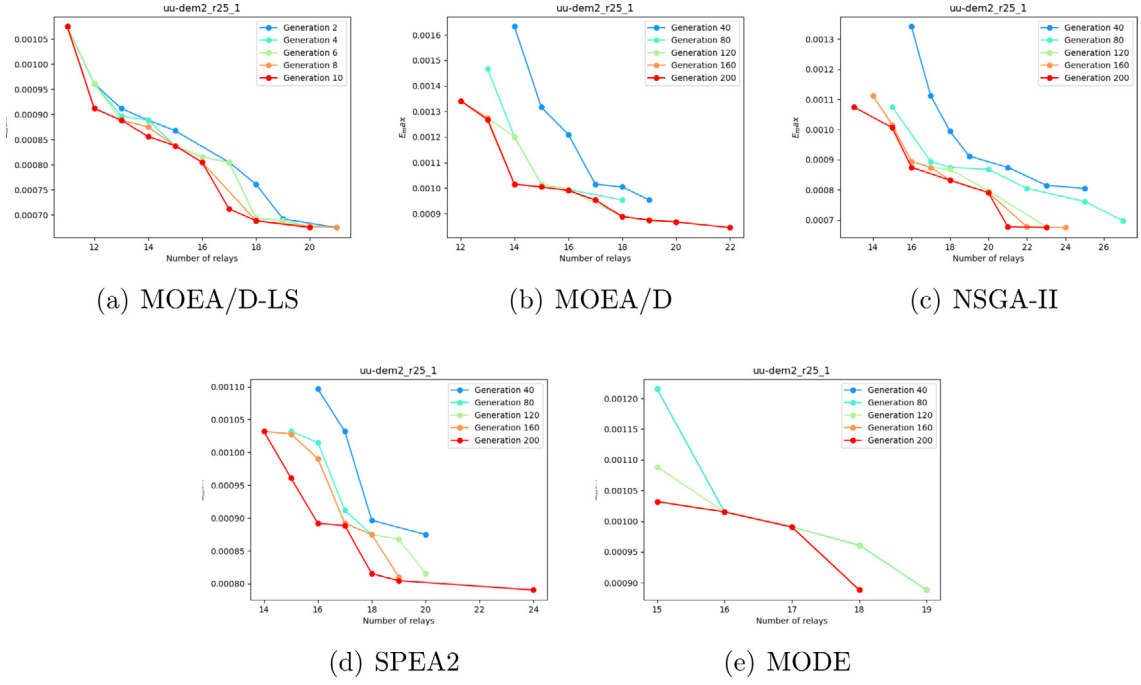


Fig. 9. Experiment 3: Convergence on test *uu25s_2* of Type 3s.

where $\overline{f_1^X}$ is the minimally used relay nodes obtained by an algorithm X, and $\overline{f_2^X}$ is the minimal energy consumption obtained by an algorithm X.

Table 11 summarizes the number of instances on which MOEA/D-LS outperforms the other MOEAs in both objectives (the number of relay nodes and energy consumption).

First, we compare the number of deployed relay nodes of MOEA/D-LS and NSGA-II with local search (NSGA-II-LS) in Figs. 12 and 13. It is obvious that the proposed MOEA/D-LS outperforms traditional MOEA/D, NSGA-II, SPEA2, and MODE for the considered scenarios.

In Fig. 12, we can see that MOEA/D-LS tends to perform well regardless of node distribution and communication range in both scenarios (small terrains and large terrains). Specifically, the average MOEA/D-LS result with Gamma distribution (Type 1s) uses 13 relays, while those of MOEA/D, NSGA-II, SPEA2, and MODE are 13, 14, 14, and 16, respectively. Average results for Gaussian and Uniform distribution (Type 2s and Type 3s) are 14 and 11 compare with 16 and 14 for the best remaining algorithms.

Table 10
CR for MOEA/D-LS and other MOEAs. Italic values indicate the best result for each dataset.

Datasets		MOEA/D-LS	MOEA/D	NSGA-II	SPEA2	MODE
200 m × 200 m	Type 1s	*0.99*	0.78	0.86	0.76	0.76
	Type 2s	*0.99*	0.77	0.85	0.75	0.74
	Type 3s	*0.99*	0.76	0.87	0.74	0.75
	Type 4s	*1.00*	0.75	0.86	0.73	0.74
	Type 5s	*1.00*	0.71	0.83	0.70	0.71
	Average	**0.99**	**0.75**	**0.85**	**0.74**	**0.74**
500 m × 500 m	Type 1l	*0.99*	0.67	0.76	0.65	0.67
	Type 2l	*0.99*	0.66	0.75	0.64	0.66
	Type 3l	*0.99*	0.68	0.77	0.66	0.68
	Type 4l	*0.99*	0.68	0.78	0.67	0.69
	Type 5l	*1.00*	0.62	0.75	0.62	0.65
	Average	**0.99**	**0.66**	**0.76**	**0.65**	**0.67**

MOEA/D-LS results in Scenario 2 from Fig. 13 are 26, 30, and 30 for Type 1l, Type 2l, Type 3l, respectively. For MOEA/D these figures are 41, 45, and 43. This indicates that our proposed algorithm works well in all distributions.

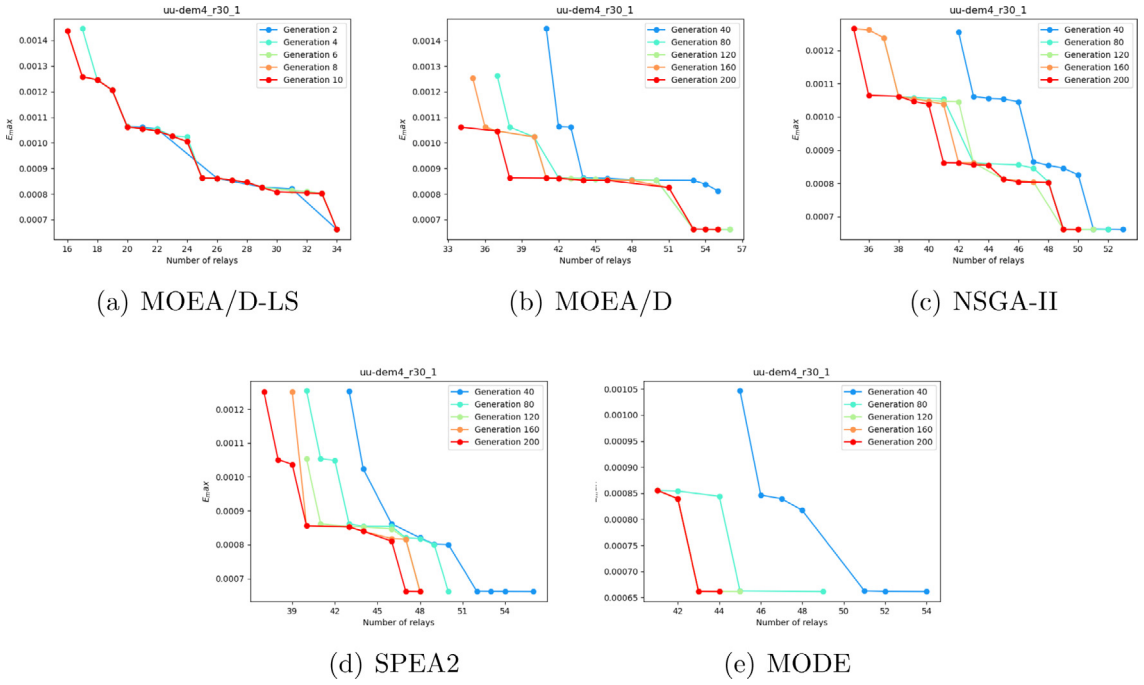


Fig. 10. Experiment 3: Convergence on test *uu*30*l*_4 of Type 3l.

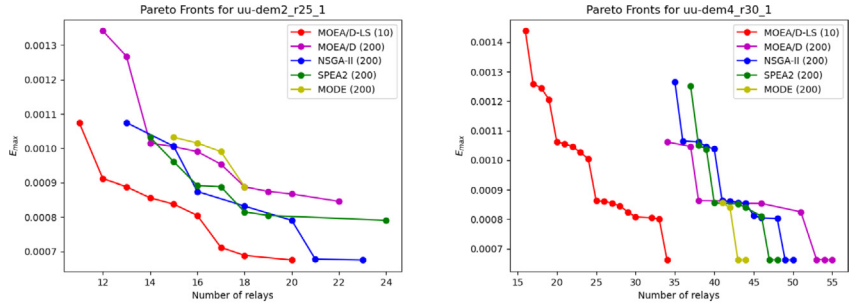


Fig. 11. Experiment 3: Comparison between last generation fronts of MOEAs. (Numbers of generations are put in parentheses right after algorithms' labels).

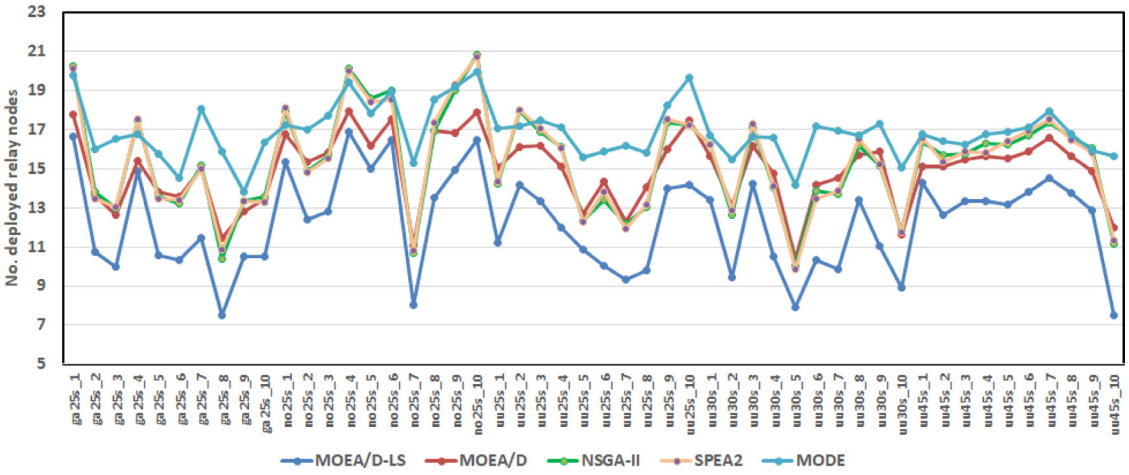


Fig. 12. The number of deployed relay nodes in Scenario 1 (Type 1s to Type 5s).

We investigate our algorithm's performance when applied to two different sensor communication radii: 30m and 45m. Type 4s, Type 4l (radius 30m), Type 5s and Type 5l (radius 45m) are used in this experiment.

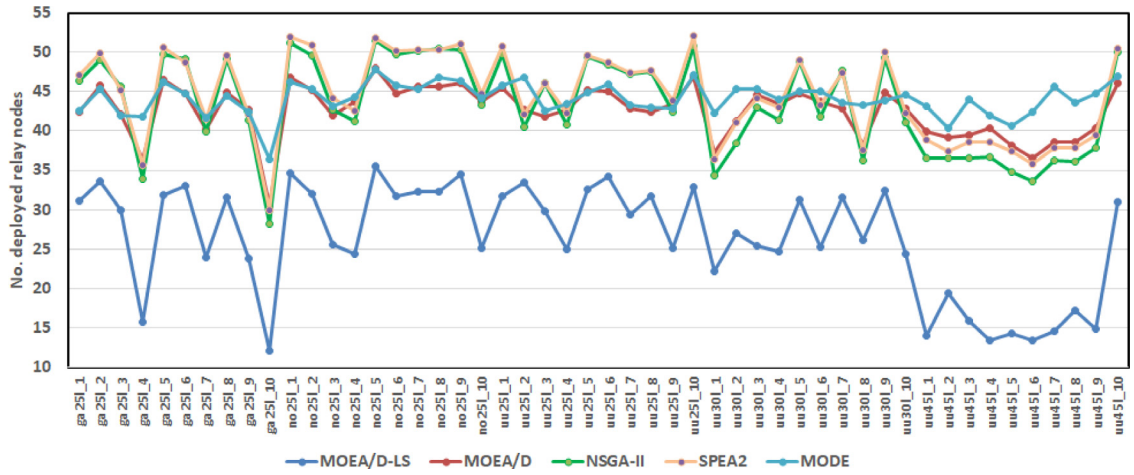


Fig. 13. The number of deployed relay nodes in Scenario 2 (Type 1l to Type 5l).

Table 11
Summary of the number of instances for which MOEA/D-LS outperform the other MOEAs.

Datasets		MOEAs								No. instances
		MOEA/D		NSGA-II		SPEA2		MODE		
		$\mathcal{S}_2$	$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_1$	
200 m × 200 m	Type 1s	2	10	0	10	0	10	0	10	10
	Type 2s	8	10	0	10	1	10	0	10	10
	Type 3s	6	10	0	10	1	10	5	10	10
	Type 4s	9	10	1	10	1	10	8	10	10
	Type 5s	9	10	1	10	1	10	9	10	10
500 m × 500 m	Type 1l	1	10	1	10	1	10	0	10	10
	Type 2l	0	10	0	10	0	10	0	10	10
	Type 3l	4	10	3	10	4	10	0	10	10
	Type 4l	4	10	4	10	5	10	0	10	10
	Type 5l	3	10	3	10	3	10	0	10	10

From Fig. 12, we can see that the average number of relay nodes for 30m radius (Type 4s) for our proposed algorithm and NSGA-II (the best of the remaining algorithms) are 10 and 14 for small terrains (scenario 1). In Type 5s, these figures have not changed significantly: 12 for MOEA/D-LS and 15 for NSGA-II.

In scenario 2, the results of MOEA/D-LS and NSGA-II on Type 4l are 27 and 42, respectively. As the radius increases, the communication becomes denser, allowing more optimal connections, leading to more optimal results. The number of relay nodes decreases to 16 for MOEA/D-LS and 37 for NSGA-II with a 45m radius (Type 5l).

Second, we compare the algorithms in terms of energy consumption. The different MOEAs all have approximately the same energy consumption. In Table 12, we report the energy consumption from MOEAs, where the italic values indicate the best results for each instance. We also compute the minimum energy consumption from non-dominated solutions obtained in each instance. It can be seen that the results from MOEA/D-LS are better than in Type 3l, Type 4l, Type 5l. In scenario 2, MOEA/D-LS improves energy consumption average over MOEA/D, NSGA-II, SPEA2, and MODE by approximately 1.14%, 0.897%, 0.37%, and 3.01%, respectively. In addition, we can see that the efficiency of MOEA/D-LS becomes more pronounced when working with a large terrains and larger communication radii.

Table 12
Comparison of energy consumption (mJ) of different algorithms on Scenario 2. Italic values indicate the best results for each instance.

Datasets		MOEA/D-LS	MOEA/D	NSGA-II	SPEA2	MODE
200 m × 200 m	Type 1s	0.952	0.915	0.848	0.865	*0.834*
	Type 2s	0.797	0.819	*0.738*	0.751	0.747
	Type 3s	1.091	1.105	*1.018*	1.031	1.083
	Type 4s	1.148	1.160	*1.081*	1.112	1.137
	Type 5s	4.383	4.432	*4.316*	4.337	4.413
	Average	**1.674**	**1.686**	***1.600***	**1.619**	1.643
500 m × 500 m	Type 1l	2.799	2.839	2.713	2.733	*2.680*
	Type 2l	0.840	0.858	0.737	0.756	*0.705*
	Type 3l	*7.373*	7.614	7.594	7.498	7.464
	Type 4l	*13.692*	13.791	13.708	13.742	13.945
	Type 5l	*34.779*	35.071	34.950	35.291	36.540
	Average	***11.897***	**12.035**	**11.940**	**12.004**	**12.267**

Table 13
Comparison of $\mathcal{S}$-metric between MOEA/D-LS and FCLS.

Datasets		$\mathcal{S}$(MOEA/D-LS, FCLS)
200 m × 200 m	Type 1s	3
	Type 2s	6
	Type 3s	9
	Type 4s	4
	Type 5s	4
	Average	**5.2**
500 m × 500 m	Type 1l	6
	Type 2l	4
	Type 3l	6
	Type 4l	5
	Type 5l	9
	Average	**6**

6.4.5. Comparison with single-objective FCLS

In our final experiment, MOEA/D-LS is compared with FCLS - another proposed method for the same model. The main difference is that FCLS to convert MOO-ORP3D to a single objective problem by weight vector. Therefore, we apply methods described in [44] to compare the single-objective FCLS with the multi-objective MOEA/D-LS.

Among three variants of single-objective algorithm introduced in [44], we choose SOGA-3 for FCLS. With a prespecified $d = 4$, this method creates 5 weight vectors: (0.0, 1.0), (0.25, 0.75), (0.5, 0.5), (0.75, 0.25), (1.0, 0.0). Five different vectors used for FCLS may return five distinct solutions, which will be assembled to form a single Pareto front. We compare this Pareto front to MOEA/D-LS in Fig. 14.

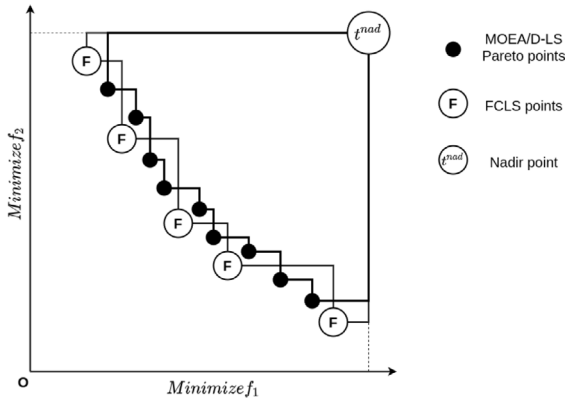


Fig. 14. This is the first figure.

Table 13 shows the improvement count $\mathcal{S}$ between FCLS and MOEA/D-LS achieved after calculating hypervolume metric. However, it can be seen that the improvement count in both scenarios are close to average (5.2 for the 200m × 200m datasets and 6.0 for the 500m × 500m ones). This means the results of two algorithms have no significant difference.

On the other hand, FCLS requires 5 consecutive runs to acquire an equivalent Pareto front to one single run of MOEA/D-LS, meaning MOEA/D-LS still has an advantage in terms of computational load.

6.5. Complexity analysis

6.5.1. Time complexity of the local search module

The local search module in Section 5.5 is used to optimize every new individual y after the reproduction step. Each run has two main steps regardless of order:

- Relay count reduction (Algorithm 3): We define $R = \{r_1, r_2, \ldots\}$ as the set of chosen relay nodes in y ($r_j \in L$), and n'_{r_j} as the number of sensors that have connections with r_j. For each chosen node r_j, it takes $|R| - 1$ iterations or less to loop through other ones, multiplied by n'_{r_j} to check if an other relay is qualified to handle r_j's connections. As a result, the cost to try removing r_j is $O(|R|^2 \times n'_{r_j})$, and the cost for all nodes is $O(\sum_{r_j \in R}(|R|^2 \times n'_{r_j}))$. Interestingly, the total number of connections equals the number of sensors, so $\sum_{r_j \in R}(n'_{r_j}) = n$. Therefore, the complexity of this step is: $O(\sum_{r_j \in R}(|R|^2 \times N_{r_j})) = O(|R|^2 \times \sum_{r_j \in R}(n'_{r_j})) = O(|R|^2 n) = O(m^2 n)$.
- Energy reduction (Algorithm 4): This step is actually repeating iterations of applying two small techniques on an energy-consuming relay r_{crit}: disconnect it with a sensor (by switching that sensor's link to another relay), or directly exchange it with a better unselected relay.
  - In the former case, it requires at most $n'_{r_{crit}}$ loops to iterate through all child sensors of r_{crit}, and then find an alternative relay for each sensor among $|R| - 1$ remaining ones. The complexity of this step is $O(|R| \times n'_{r_{crit}})$.
  - The latter also has two nested loops, one to find a relay among $m - |R|$ unselected nodes, and the other to verify if that node can connect with all of $n'_{r_{crit}}$ sensors. Thus, it takes $O((m - |R|) \times n'_{r_{crit}})$ time for each run.

  As a consequence, each step which consists of both runs in $O(m \times n'_{r_{crit}}) = O(m \times n)$ time. The tricky measurement here is how many time those two steps are executed. This attribute is mostly affected by the dense of valid connections in an instance. It would be safer to specify a parameter T - the maximum number of times Algorithm 5 is applied, which limits the computation cost to $O(Tmn)$. In practice, this module actually run pretty fast even without adding any extra stopping criteria.

In conclusion, the module runs in $O(m^2 n + Tmn) = O(mn(m + T))$ time for each individual y. Since each generation produces N new solutions, the total complexity of local search in each generation is $O(Nmn(m + T))$.

Table 14
Computational costs of MOEAs for each generation.

Algorithm	Complexity
MOEA/D	$O(Nn + NK)$
MOEA/D-LS	$O(Nn + NK + Nmn(m + T))$
NSGA-II	$O(Nn + N^2)$
SPEA2	$O(Nn + N^2 log(N))$
MODE	$O(Nn + N^2)$

6.5.2. Computational costs of MOEAs

It should be noted that each operation (genetic operator, MODE's reproduction procedure or fitness calculation) in this problem has $O(n)$ complexity. Since all MOEAs use the same population size $N = 40$, the time complexity of the reproduction step is $O(Nn)$. Their performances differ from each other due to custom steps:

- MOEA/D's population update step has a $O(NK)$ complexity.
- The fast non-dominated sorting algorithm used in NSGA-II and MODE costs $O(N^2)$.
- The truncation procedure of SPEA2 runs in $O(N^2 log(N))$ time.
- The local search module's cost is $O(Nmn(m + T))$, as mentioned before.

Because MOO-ORP3D only has two objectives, this constant is skipped. Table 14 shows MOEAs' run time for each generation.

The results show that MOEA/D-LS's optimization module has significant higher complexity than MOEAs. This is an expected tradeoff for improved Pareto fronts. On the other hand, since it is a local search algorithm, its performance in practice is better than theory-based evaluation. Moreover, MOEA/D-LS converges much faster than other MOEAs, which also contribute to its reasonable balance between complexity and optimization.

7. Conclusions

A multi-objective optimization algorithm aims at producing solutions as close as possible to the Pareto-Optimal Set and as diverse as possible in the obtained non-dominated set. This diverse set of optimal solutions expresses trade-offs between different objectives. In this paper, we propose the MOEA/D-LS algorithm to maximize the network lifetime with a trade-off between the number of used relay nodes and energy consumption. A multi-objective fitness function is derived to model the two conflicting objectives, while Pareto dominance is employed to select the local and global best solution for each individual. The MOEA/D-LS algorithm has been proposed based on the well-known MOEA/D algorithm. In MOEA/D-LS, solution encoding ensures that all generated individuals are feasible. Genetic operators also create the children who completely inherit parents' characteristics. To improve solutions, a local search method is proposed to reduce the number of relays used and the maximum node energy consumption. The algorithm was put through rigorous simulations and

evaluations. Our results demonstrate that MOEA/D-LS outperforms existing methods in several critical performance metrics. In our future work, we plan to apply learning strategies to further improve the performance of our proposed algorithms. In addition, we consider balancing the energy consumption problem of WSN within more complex real-world scenarios such as those with mobility nodes in the networks. Furthermore, we also intend to extend the same for the heterogeneous wireless sensor networks in contrast to the proposed scheme which works only for homogeneous WSNs.

CRediT authorship contribution statement

Nguyen Thi Tam: Methodology, Algorithms, Coding, Writing - original draft. **Tran Huy Hung:** Algorithms, Coding, Experiment, Writing - original draft. **Huynh Thi Thanh Binh:** Methodology, Algorithms, Writing - original draft, Supervision. **Le Trong Vinh:** Methodology, Algorithms, Writing - original draft, Supervision.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Nguyen Thi Tam, Tran Huy Hung, Huynh Thi Thanh Binh, Le Trong Vinh

Acknowledgments

Nguyen Thi Tam was funded by Vingroup Joint Stock Company, Vietnam and supported by the Domestic Master/PhD Scholarship Programme of Vingroup Innovation Foundation (VINIF), Vingroup Big Data Institute (VINBIGDATA), code VINIF.2020.TS.105.

References

[1] D. Bruckner, C. Picus, R. Velik, W. Herzner, G. Zucker, Hierarchical semantic processing architecture for smart sensors in surveillance networks, IEEE Trans. Ind. Inf. 8 (2) (2012) 291–301.
[2] J. Yick, B. Mukherjee, D. Ghosal, Wireless sensor network survey, Comput. Netw. 52 (12) (2008) 2292–2330.
[3] H. Mostafaei, Energy-efficient algorithm for reliable routing of wireless sensor networks, IEEE Trans. Ind. Electron. 66 (7) (2018) 5567–5575.
[4] S.P. Singh, S. Sharma, A survey on cluster based routing protocols in wireless sensor networks, Procedia Comput. Sci. 45 (2015) 687–695.
[5] A. Paul, T. Sato, Localization in wireless sensor networks: a survey on algorithms, measurement techniques, applications and challenges, J. Sensor Actuator Netw. 6 (4) (2017) 24.
[6] N. Yessad, M. Omar, A. Tari, A. Bouabdallah, QoS-based routing in Wireless Body Area Networks: A survey and taxonomy, Computing 100 (3) (2018) 245–275.
[7] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow, P. Polakos, Wireless sensor network virtualization: A survey, IEEE Commun. Surv. Tutor. 18 (1) (2015) 553–576.
[8] G. Dhand, S. Tyagi, Data aggregation techniques in WSN: Survey, Procedia Comput. Sci. 92 (2016) 378–384.
[9] H. Yetgin, K.T.K. Cheung, M. El-Hajjar, L.H. Hanzo, A survey of network lifetime maximization techniques in wireless sensor networks, IEEE Commun. Surv. Tutor. 19 (2) (2017) 828–854.
[10] A. Ghaffari, Congestion control mechanisms in wireless sensor networks: A survey, J. Netw. Comput. Appl. 52 (2015) 101–115.
[11] O. Cayirpunar, E. Kadioglu-Urtis, B. Tavli, Optimal base station mobility patterns for wireless sensor network lifetime maximization, IEEE Sens. J. 15 (11) (2015) 6592–6603.
[12] O. Moh'd Alia, Dynamic relocation of mobile base station in wireless sensor networks using a cluster-based harmony search algorithm, Inform. Sci. 385 (2017) 76–95.
[13] N.T. Tung, D.H. Ly, H.T.T. Binh, Maximizing the lifetime of wireless sensor networks with the base station location, in: International Conference on Nature of Computation and Communication, Springer, 2014, pp. 108–116.
[14] N.T. Tung, H.T.T. Binh, Base station location-aware optimization model of the lifetime of wireless sensor networks, Mob. Netw. Appl. 21 (1) (2016) 10–17.
[15] H.Z. Abidin, N.M. Din, N.A.M. Radzi, Z.I. Rizman, A review on sensor node placement techniques in wireless sensor networks, Int. J. Adv. Sci. Eng. Inf. Technol. 7 (1) (2017) 190–197.
[16] K.Y. Bendigeri, J.D. Mallapur, Multiple node placement strategy for efficient routing in wireless sensor networks, Wirel. Sensor Netw. 7 (08) (2015) 101.
[17] L. Liu, M. Ma, C. Liu, Y. Shu, Optimal relay node placement and flow allocation in underwater acoustic sensor networks, IEEE Trans. Commun. 65 (5) (2017) 2141–2152.
[18] F. Senel, M. Younis, Novel relay node placement algorithms for establishing connected topologies, J. Netw. Comput. Appl. 70 (2016) 114–130.
[19] N.T. Tam, H.T.T. Binh, D.A. Dung, P.N. Lan, B. Yuan, X. Yao, et al., A hybrid clustering and evolutionary approach for wireless underground sensor network lifetime maximization, Inform. Sci. 504 (2019) 372–393.
[20] N.T. Tam, H.T.T. Binh, T.H. Hung, D.A. Dung, et al., Prolong the network lifetime of wireless underground sensor networks by optimal relay node placement, in: International Conference on the Applications of Evolutionary Computation (Part of EvoStar), Springer, 2019, pp. 439–453.
[21] Y. Xu, Y. Xiao, Q. Sun, A swarm-based meta-heuristic for relay nodes placement in wireless sensor networks, Int. J. Innov. Comput. Inf. Control 15 (2) (2019) 551–567.
[22] N.T. Tam, H.T.T. Binh, V.T. Dat, P.N. Lan, et al., Towards optimal wireless sensor network lifetime in three dimensional terrains using relay placement metaheuristics, Knowl.-Based Syst. 206 (2020) 106407.
[23] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[24] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[25] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the Strength Pareto Evolutionary Algorithm, Vol. 103, TIK-Report, Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische, 2001.
[26] F. Xue, A.C. Sanderson, R.J. Graves, Pareto-based multi-objective differential evolution, in: The 2003 Congress on Evolutionary Computation, 2003, Vol. 2, CEC'03, IEEE, 2003, pp. 862–869.
[27] S. Misra, S.D. Hong, G. Xue, J. Tang, Constrained relay node placement in wireless sensor networks: Formulation and approximations, IEEE/ACM Trans. Netw. 18 (2) (2009) 434–447.
[28] W. Zhang, S. Bai, G. Xue, J. Tang, C. Wang, Darp: Distance-aware relay placement in wimax mesh networks, in: 2011 Proceedings IEEE INFOCOM, IEEE, 2011, pp. 2060–2068.
[29] H.A. Hashim, B.O. Ayinde, M.A. Abido, Optimal placement of relay nodes in wireless sensor network using artificial bee colony algorithm, J. Netw. Comput. Appl. 64 (2016) 239–248.
[30] B. Yuan, H. Chen, X. Yao, Optimal relay placement for lifetime maximization in wireless underground sensor networks, Inform. Sci. 418 (2017) 463–479.
[31] W. Peng, Q. Zhang, H. Li, Comparison between MOEA/D and NSGA-II on the multi-objective travelling salesman problem, in: Multi-Objective Memetic Algorithms, Springer, 2009, pp. 309–324.
[32] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P.N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: A survey of the state of the art, Swarm Evol. Comput. 1 (1) (2011) 32–49.
[33] J.M. Lanza-Gutierrez, J.A. Gomez-Pulido, Assuming multiobjective metaheuristics to solve a three-objective optimisation problem for Relay Node deployment in Wireless Sensor Networks, Appl. Soft Comput. 30 (2015) 675–687.
[34] Y. Xu, O. Ding, R. Qu, K. Li, Hybrid multi-objective evolutionary algorithms based on decomposition for wireless sensor network coverage optimization, Appl. Soft Comput. 68 (2018) 268–282.
[35] X. Zhang, Y. Zhou, Q. Zhang, V.C. Lee, M. Li, Problem specific MOEA/D for barrier coverage with wireless sensors, IEEE Trans. Cybern. 47 (11) (2016) 3854–3865.
[36] M. Baghouri, A. Hajraoui, S. Chakkor, Low energy adaptive clustering hierarchy for three-dimensional wireless sensor network, Recent Adv. Commun. (2015) 214–218.
[37] R.D. Gawade, S.L. Nalbalwar, A centralized energy efficient distance based routing protocol for wireless sensor networks, J. Sensors 2016 (2016).
[38] D.T. Hai, T. Le Vinh, et al., Novel fuzzy clustering scheme for 3D wireless sensor networks, Appl. Soft Comput. 54 (2017) 141–149.

[39] N.T. Tam, D.T. Hai, et al., Improving lifetime and network connections of 3D wireless sensor networks based on fuzzy clustering and particle swarm optimization, Wirel. Netw. 24 (5) (2018) 1477–1490.
[40] K. Miettinen, Nonlinear Multiobjective Optimization, Vol. 12, Springer Science & Business Media, 2012.
[41] I. Das, J. Dennis, Normal-Boundary Intersection: A new method for generating Pareto-optimal points in multieriteria optimization problems, SIAM J. Optim. (1996).
[42] M. Laszczyk, P.B. Myszkowski, Survey of quality measures for multi-objective optimization. Construction of complementary set of multi-objective quality measures, Swarm Evol. Comput. 48 (2019) 109–133.
[43] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Trans. Evol. Comput. 3 (4) (1999) 257–271.
[44] H. Ishibuchi, Y. Nojima, T. Doi, Comparison between single-objective and multi-objective genetic algorithms: Performance comparison and performance measures, in: 2006 IEEE International Conference on Evolutionary Computation, IEEE, 2006, pp. 1143–1150.