Two Parameter-Tuned Multi-Objective Evolutionary-Based Algorithms for Zoning Management in Marine Spatial Planning

Mohadese Basirati, Romain Billot, Patrick Meyer

▶ **To cite this version:**

Mohadese Basirati, Romain Billot, Patrick Meyer. Two Parameter-Tuned Multi-Objective Evolutionary-Based Algorithms for Zoning Management in Marine Spatial Planning. Annals of Mathematics and Artificial Intelligence, 2023, 10.1007/s10472-023-09853-2 . hal-03794648

HAL Id: hal-03794648

https://imt-atlantique.hal.science/hal-03794648

Submitted on 3 Oct 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Two Parameter-Tuned Multi-Objective Evolutionary-Based Algorithms for Zoning Management in Marine Spatial Planning

Mohadese Basirati[a,*], Romain Billot[a], Patrick Meyer[a]

[a]*IMT Atlantique, Lab-STICC, UMR CNRS 6285, Brest F-29238, France*

Abstract

Strategic spatial planning is becoming more popular around the world as a decision-making way to build a unified vision for directing the medium- to long-term development of land/marine areas. Recently, the study of marine areas in terms of spatial planning such as Marine Spatial Planning (MSP) has received much attention. One of the challenging issues in MSP is to make a balance between determining the ideal zone for a new activity while also considering the locations of existing activities. This spatial zoning problem for multi-uses with multiple objectives could be formulated as optimization models. This paper presents and compares the results of two multi-objective evolutionary-based algorithms (MOEAs), Synchronous Hypervolume-based non-dominated sorting genetic algorithm-II (SH-NSGA-II) which is an extension of NSGA-II and a memetic algorithm (MA) in which SH-NSGA-II is enhanced with a local search. These proposed algorithms are used to solve the multi-objective spatial zoning optimization problem, which seeks to maximize the zone interest value assigned to the new activity while simultaneously maximizing its spatial compactness. We introduce several innovations in these proposed algorithms to address the problem constraints and to improve the robustness of the traditional NSGA-II and MA approaches. Unlike traditional ones, a different stop condition, multiple crossover, mutation, and repairing operators, and also a local search operator are developed. A comparative study is presented between the results obtained using both algorithms. To guarantee robust results for both algorithms, their parameters are calibrated and tuned using the Multi-Response Surface Methodology (MRSM) method. The effective and non-effective components, as well as the validity of the regression models, are determined using analysis of variance (ANOVA). Although SH-NSGA-II has revealed a good efficiency, its performance is still improved using a local search scheme within SH-NSGA-II, which is specially tailored to the problem characteristics. The two methods are designed for raster data.

Keywords: Multi-Objective Spatial Zoning Optimization, Evolutionary Algorithms, SH-NSGA-II, Memetic Algorithm, Multi-Response Surface Methodology, Marine Spatial Planning, Raster

1. Introduction

Spatial management planning is a resource location-allocation strategy described as the process of locating and allocating distinct human activities or uses to the specified units of areas on the Earth's surface [13, 15].

One of the first and foremost prerequisites for properly managing spatial planning is spatial data, often known as geospatial or geographic data. That is, data about the geographic position of features and boundaries on the Earth's surface, such as natural features, land regions, ocean surfaces, and so on. Coordinates and topologies are commonly used to map and store spatial data [33, 5].

*Corresponding author.

Email addresses: mohadese.basirati@imt-atlantique.fr (Mohadese Basirati), romain.billot@imt-atlantique.fr (Romain Billot), patrick.meyer@imt-atlantique.fr (Patrick Meyer)

Land-use planning [18], biodiversity conservation planning [43], MSP [5] and military planning [27] are only a few of the decision issues in the spatial planning strategy that employ spatial data. Due to competing goals and restrictions in the mentioned problems, selecting the optimum zone(s) or area(s) for a certain purpose using geographical data may be challenging. In contrast to land-use planning, maritime activities were not handled for many years, since the sea was thought to be unbound and its resources inexhaustible [8, 5]. Today, protecting the global marine environment has become critical and the MSP strategy is a central tool for developing sustainable human activities in the ocean, taking into account the interactions between different activities and stakeholders [2].

1.1. Problem definition

In MSP, decision issues are optimization problems in which we must determine the best location and/or shape of a geographic region for a certain activity, given certain constraints. Two sub-optimization difficulties here are the appropriate location and shape of a spatial region.

Solving these two sub-optimization issues at the same time is a relatively new class of optimization problems, and there has not been much study on it in the literature. As previously stated, we focus on the zoning problem in this study, a special sub-topic of MSP in which a fixed number of human activities already exist in a certain maritime region and the best site for a new activity must be established. The area's present activities are regarded fixed and cannot be changed. The overall attractiveness of the new activity's location is determined by a map that is provided for the whole region and reflects the degree to which it is worthwhile or unattractive to complete the activity at each place in the area.

The existing activities in the area can be classified into 3 different categories :

- **shipping lanes** (known as sea lane or sea route), are navigable paths regularly used by large ships, which cannot be broken down into new activity.
- **ports** (also called harbours), The ships are situated on the edge of the ocean (on the mainland) and are used for loading and unloading cargo or passengers,
- **restricted areas** in the ocean, are considered additional activities and do not intersect with new activity. These restricted areas can, for example, represent marine protected areas, wind or tidal turbine farms, recreational areas, military areas, etc.

Meanwhile, in addition to the interest map, the other criteria that need to be considered in finding the precise zone for the new activity is the distance to the different elements of the three categories of existing activities. Therefore, they could be imposed on in terms of the constraints to the problem as the minimum and the maximum distance of the new activity to the different existing activities. In other words, the new activity should be located:

- at a minimum distance of each of the existing activities (depending on each existing activity),
- at a maximum distance of each of the existing activities (depending on each existing activity).

Finally, in our case, it is preferred that the zone be as compact as possible for two main reasons. First, an activity may also need a compact zone without holes in it; second, it also prevents potential conflicts with other new activities that may arise in the region in the future.

Therefore, the purpose of this problem is to find the optimal location for the new activity that maximizes not only its interest, but also its compactness. At the same time, it must meet the minimum and maximum distance constraints for existing activities without crossing them.

As presented in a recent article [5], Figure 1 introduces the definition of the problem in a fictive maritime area and all its distinct elements. The upper dark gray area of the picture (with the topographical isolines) shows the mainland, on which four ports are located. The lower section illustrates the marine region, where the new activity (e.g., fishing) must be situated and where many other activities already exist: numerous shipping lanes, a windmill farm (restricted area), and a protected natural area (restricted area). The interest

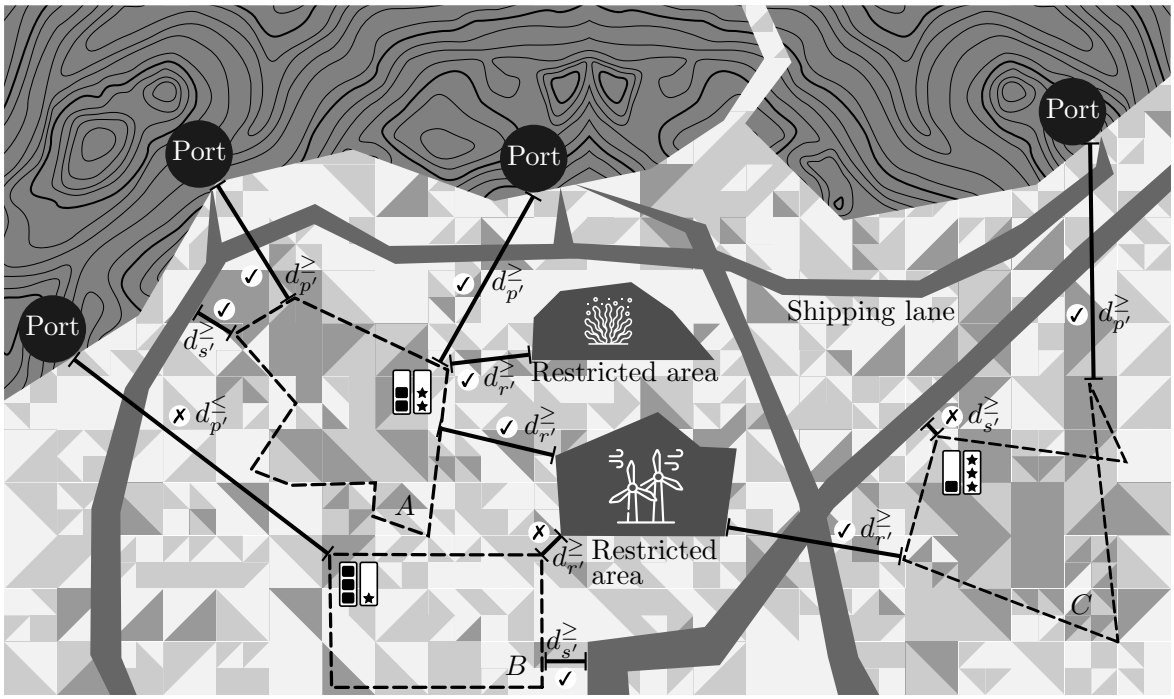

Figure 1: Problem Definition [5]

for the new activity is displayed on the background map in three shades of gray (the more interesting the area, the darker it is). In this situation, the new activity must be placed within a defined minimum distance of the shipping lanes ($(d^{\geqslant}_{s'})$), a given minimum and maximum distance of the closest port ($d^{\geqslant}_{p'}$ and $d^{\leqslant}_{p'}$), and a provided minimum distance of restricted regions ($d^{\geqslant}_{r'}$). The picture depicts three places for the new activities. A is situated in an area of the sea that is ideal for the new activity. B, on the other hand, is in a less interesting section of the marine region, but C is in a highly fascinating part of the maritime area. The star rating represents the average interest of the three areas (1 star corresponds to a low interest, and 3 stars to a high interest). Furthermore, each of these three areas has a separate compactness assessment, which is shown by squares rating box. As B is a rectangle, it is very compact, A is fairly compact, whereas C is not very compact. Choosing between these three locations merely on the basis of their compactness and interest is a challenging task, as none of them surpasses the others on either measure. Moreover, the graphic depicts some of the distance limits. As shown in Figure 1, among the three proposed zones A, is meeting all minimum and maximum distance restrictions, while is extremely compact and interesting. On the other hand, the area B ignores a limit on the maximum distance to the nearest port ($d^{\leqslant}_{p'}$) and a limit on the minimum distance to the restricted windmill farm ($d^{\geqslant}_{r'}$). While it is an utterly compact zone, but less interesting. Similarly, the region C fails the minimum distance limit to the shipping lane ($d^{\geqslant}_{s'}$), while being less compact and yet infinitely interesting. Finally, among all, it is only the area A the most preferable by considering objectives and constraints at the same time.

However, the challenge at hand begins with geospatial data, which is data relating to or including information about places on the Earth's surface. In this study, we employ the raster data given as a regular grid of cells or pixels. A value is assigned to each pixel in a raster that reflects some unit of measurement about the underlying geographical area. The quality of raster data is mainly determined by its resolution. As a result, we assume that the interest map for the new activity is a two-dimensional matrix of uniform cells on a regular grid with n_{row} rows and n_{col} columns, yielding a total of $n_{row} \cdot n_{col} = m$ cells. Each cell in this grid is supposed to have a homogeneous interest value for the particular activity.

1.2. Related Work and knowledge gaps

The majority of spatial zoning approaches are expressed as multi-objective non-linear optimization models, which are frequently solved using stochastic search techniques, resulting in sub-optimal solutions [34]. However, Basirati et al. offered an exact mathematical zoning model for MSP as a Multi-Objective Integer Linear Program (MOILP) in a recent study. However, the main drawback of this study is that it does not solve large-scale problems due to the high computational cost and hardness limitations of MOILP[28, 5]. Therefore, to reach good enough solutions in practice, using heuristic or Meta-Heuristic (MH) algorithms are typically sufficient for the real case studies in large scale [37, 25].

Aerts et al. used a simulated annealing to solve the spatial goal programming for the land use allocation problem. The purpose of this work is to determine the multi-site allocation between different land use, which is kinda partitioning problem without considering the existing elements as constraints.

Yao et al. highlights the prominent sustainability concerns in land use planning and suggests a generalized multi-objective spatial optimization model to facilitate conventional planning. They developed an evolutionary-based algorithm to solve the land use optimization problem. One of the limitations with this work, however, is that they are focusing on the simple partitioning problem without considering the influence of the land uses on each other. Moreover, the developed heuristic algorithm is a traditional genetic algorithm (GA).

According to Stewart and Janssen, an improved land use optimization model is proposed for land use planning with a new spatial component. A GA is developed to solve the optimization problems. The context relates to interactive decision support for land use planning in which the data are stored in a vector-based GIS, which is the extended earlier work by the authors for a grid (raster) structure. However, again the weaknesses of this work concern the mathematical formulation which is nonlinear, and it is a kind of partitioning problem solved by a traditional GA which is not compared with any other algorithm.

Following the reviewed studies in this field, the main knowledge gaps addressed in this paper is designing the efficient MOEAs to be able to:

1. solve spatial zoning optimization problems which is more than a simple partitioning problem.
2. be applicable for any real big map size.
3. converging to good enough solutions in a reasonable computing time.
4. be compatible with the raster data.

1.3. Contributions of this paper

On the basis of the knowledge gaps identified previously, this article presents the following contributions.

1. **Problem resolution**: We present two different population-based MOEAs (SH-NSGA-II and MA), which are Pareto-based techniques, to address the computational hardness issue of the exact method for the large-scale spatial zoning optimization problem in MSP. Initialization, stop condition, chromosome encoding, crossover, mutation, check and repair operators, constraint management methodologies, and algorithm structure in raster data are all suggested as innovations. The proposed MOEAs are used to simultaneously optimize the interestingness and compactness objectives of the new activity's zone.
2. **Experimental validation**: MRSM for parameters tuning: we set up a design experiment (DOE) as Box-Behnken design(BBD), which implements a multi-response regression model for three different map sizes of the problem in order to determine the optimal value of the algorithm parameters. Moreover, the effectiveness of all models are validated by Analysis of Variance (ANOVA).
3. **Comparison Analysis**: To compare two MOEAs, different performance measures are indicated and calculated for better characterization of the Pareto solutions, resulting in more precise analysis between two algorithms for small- and large-scale problem. To guarantee all conclusion, the significance value of the **Wilcoxon signed-rank test (WSRT)** (paired samples) tests of all performance measures for the exact method, SH-NSGA-II, and MA in three problem size levels are calculated and compared.

The article is structured as follows. In Section 2, we proposed the SH-NSGA-II and MA algorithms for the problem at hand to find the optimal solutions. In Section 3, we describe an experimental design to tune the parameters of the proposed algorithms, while in Section 4, we propose the computational results on artificially generated synthetic instances. Conclusions are drawn in Section 6.

2. Problem Resolution

Exploration of the search space (diversification) and exploitation of the best solutions identified (intensification) are two contradicting criteria to consider while constructing a MH. In general, basic single-solution based metaheuristics (S-metaheuristics) are more exploitation-oriented, whereas basic population-based metaheuristics (P-metaheuristics) are more exploration oriented [38]. However, at each iteration of the basic steepest local search algorithm, the best adjacent solution that improves the results is selected [38]. In other words, during the search, S-metaheuristics (such as local search and simulated annealing) control and transform a single solution, whereas P-metaheuristics (such as particle swarm and evolutionary algorithms) evolve an entire population of solutions [38]. As a result, utilizing some strategies in these algorithms that enhance the underestimated local search section might be useful to empower P-metaheuristics.

2.1. Solution Encoding Schema

One of the first tasks in successfully implementing MH algorithms is to choose the solution representation. In the spatial planning, the problem solution representation could involve geographic location dimensions ("cells", "patches", or "grids").

To randomly produce feasible initial population of solutions, we must consider various constraints while solving multi-objective optimization problems (MOOPs). For the spatial zoning optimization problem, there is a set of constraints as follows:

- The size of each solution is a fixed number of cells (the required solution size).
- Intersections between the solutions and other existing activities are not permitted.
- No hole is accepted in each solution because if a zone is closed and bounded, then it is compact [3, 40].
- The layout and structure of the solution should be uninterrupted, that is, without a break in continuity to make a solution compact.

Figure 2 depicts a mapping between the space of solution and the space of encoding. on the left of Figure 2, a raster reflects a specified zone of an activity on a map. This raster-based zone includes a regular grid of cells, or pixels. Each raster cell contains a single value, and the coordinate of each raster cell relates to the center of the cell ((x_i, y_i) in Figure 2). Each cell can be defined by a cell dimension, such as the width and height of the cell. As cells in a raster are frequently square, their width and height will be the same.

In the GAs, the term "chromosome" is used to represent one single solution, while the term "population" is used for a given number of chromosomes. Therefore, the chromosome shown on the right of Figure 2 illustrate the encoded solution as a chromosome.

In Figure 2, the direction of generating a chromosome in a raster starts from the south-west of the map shown by the numbers 1 to 15 as an example (1 is located in the first cell and 15 in the last one, that is, the order of their selection).

However, our proposed algorithms lead to a Pareto front that includes multiple different chromosomes (solutions) on the map for a specific activity, one of which is illustrated in Figure 2. Therefore, if we translate and gather multiple chromosomes all together in one matrix, we will reach to the matrix of population 1 (Pop_{list}). By doing so, the solution representation in this paper is a multi-dimensional matrix ($m \times n$), in which m shows the number of rows of the matrix and n declares the number of columns. In other words, n reveals the size of each chromosome, that is, the total number of cells of its equivalent zone, while m the given population size in the proposed algorithms (N_{pop}).

$$Pop_{list} = \begin{bmatrix} (x_{11}, y_{11}) & \cdots & (x_{1j}, y_{1j}) & \cdots & (x_{1n}, y_{1n}) \\ \vdots & & \vdots & & \\ (x_{i1}, y_{i1}) & \cdots & (x_{ij}, y_{ij}) & \cdots & (x_{in}, y_{in}) \\ \vdots & & \vdots & & \\ (x_{mi}, y_{mi}) & \cdots & (x_{mj}, y_{mj}) & \cdots & (x_{mn}, y_{mn}) \end{bmatrix}_{m \times n} \quad (1)$$

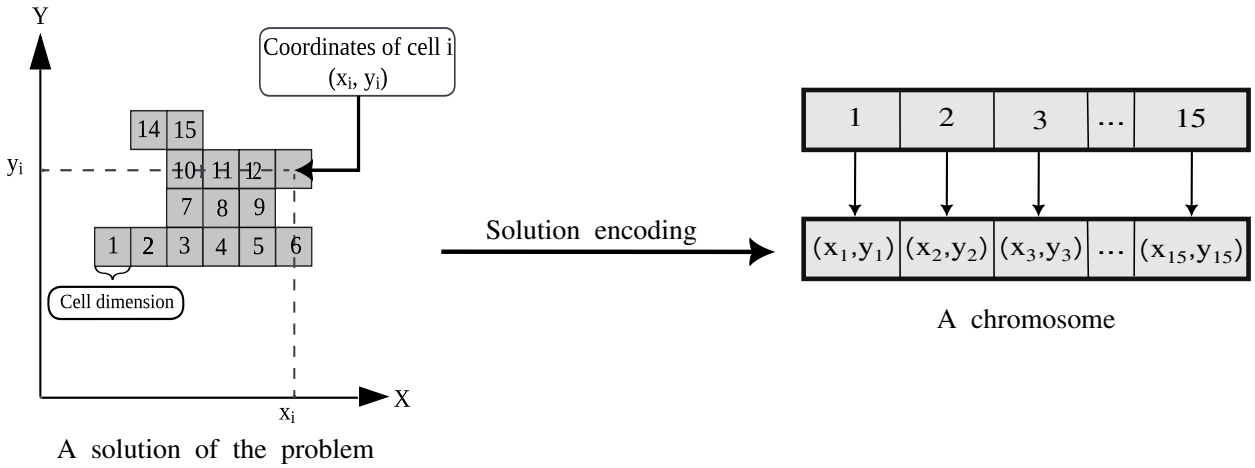

Figure 2: Mapping between the space of solution and the space of encoding

2.2. Evolutionary Multi-Objective Optimization Algorithms (SH-NSGA-II and MA)

As the computational complexity and a non-elitism approach are two major obstacles in MOOPs, MOEAs could be the efficient approaches to address them. MOEAs optimize two or more conflicting objectives by considering a collection of Pareto optimum solutions. Deb et al. proposed an efficient approach to achieve Pareto frontiers called NSGA-II. Figure 3 is a graphical depiction of NSGA-II. Figure 3 shows how the given set of five solutions (F_1-F_5) are classified into three non-dominated fronts (P_{t+1}). As can be seen on the left of Figure 3, P_t is a population that NSGA-II randomly generates with respect to population size N_p. Following that, O_t is the chosen chromosomes by the selection operator for the offspring population, with regard to a crossover rate P_c and a mutation rate P_m. As shown in the first rectangle on the left of Figure 3, NSGA-II then combines P_t and O_t to generate R_t, which it then sorts into numerous non-dominated fronts F_i based on coverage and fitness function. These sorted F_is, which are shown in the next rectangle, give rise to the next population, P_{t+1} as the last rectangle on the right, in which the best F_is form P_{t+1}. Because the size of P_{t+1} should be the same as that of P_t, all elements of F_i may not be in P_{t+1} like F_4 and F_5 marked as "Rejected" in the figure. As a result, crowding sorting is used to complete P_{t+1} by adding an incomplete front in the crowding distance technique, in which the required population is created by the top of the front elements, such as F_1, F_2, and F_3 in Figure 3, without sacrificing good solutions (elitism). NSGA-II generates O_{t+1} from P_{t+1} similarly as O_t. It then iterated the preceding processes to achieve the best Pareto solutions while keeping a stopping criterion in mind. For further information on NSGA-II, readers might consult [9, 10]. The chromosomes in P_t are then sorted into numerous fronts of non-dominated solutions.

In this paper, which is an extended version of our conference paper (LION16), we propose the SH-NSGA-II architecture for solving the spatial zoning optimization problem. Compared to the typical NSGA-II, the suggested one employs a different initialization approach, stop criterion, four crossover operators, and three mutation operators throughout the search phase [10]. Furthermore, offspring chromosomes, which are produced by the four crossovers and three mutations, may compete with parent chromosomes for survival from generation to generation. Furthermore, the proposed SH-NSGA-II includes a check and repair mechanism that prevents the search process from being trapped in local optima. In other words, the proposed SH-NSGA-II is capable of preventing repeating solutions by producing solutions with various structures and without discarding non-feasible solutions: it can fix those that require minimal changes to make them feasible/acceptable. The suggested SH-NSGA-II components will be described in depth in the following sections.

In Figure 4, the SH-NSGA-II flow chart is presented. This flowchart starts by the first randomly generated set of non-dominated population. Next, the set of initial current zones is assessed by computing their objective functions. By doing so, the main loop of the algorithm launches. One of four crossovers is chosen at random in the initial phase to create non-iterative and acceptable offspring, and this random selection is

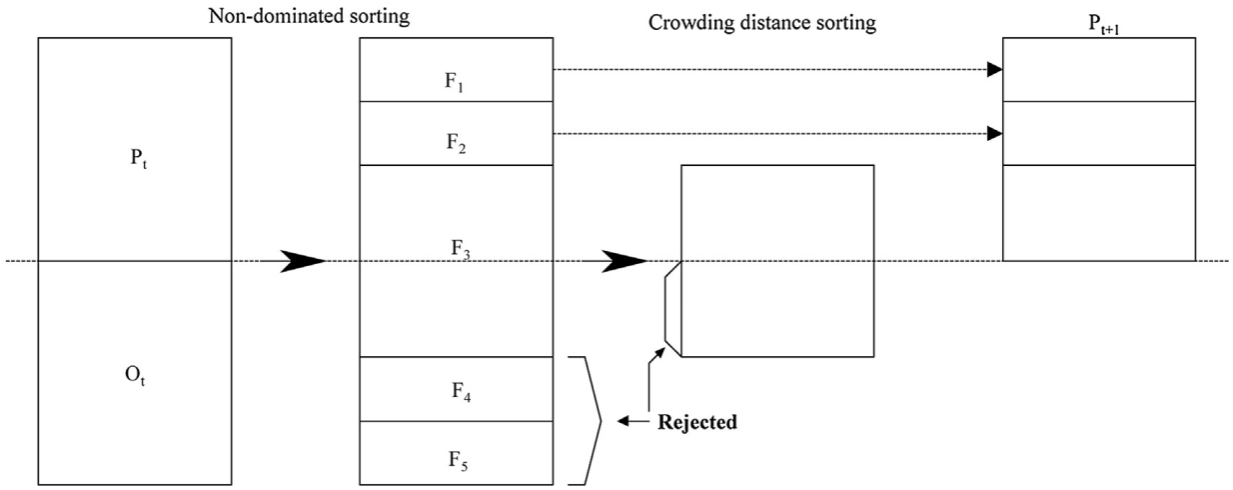

Figure 3: Graphical representation of NSGA-II [9]

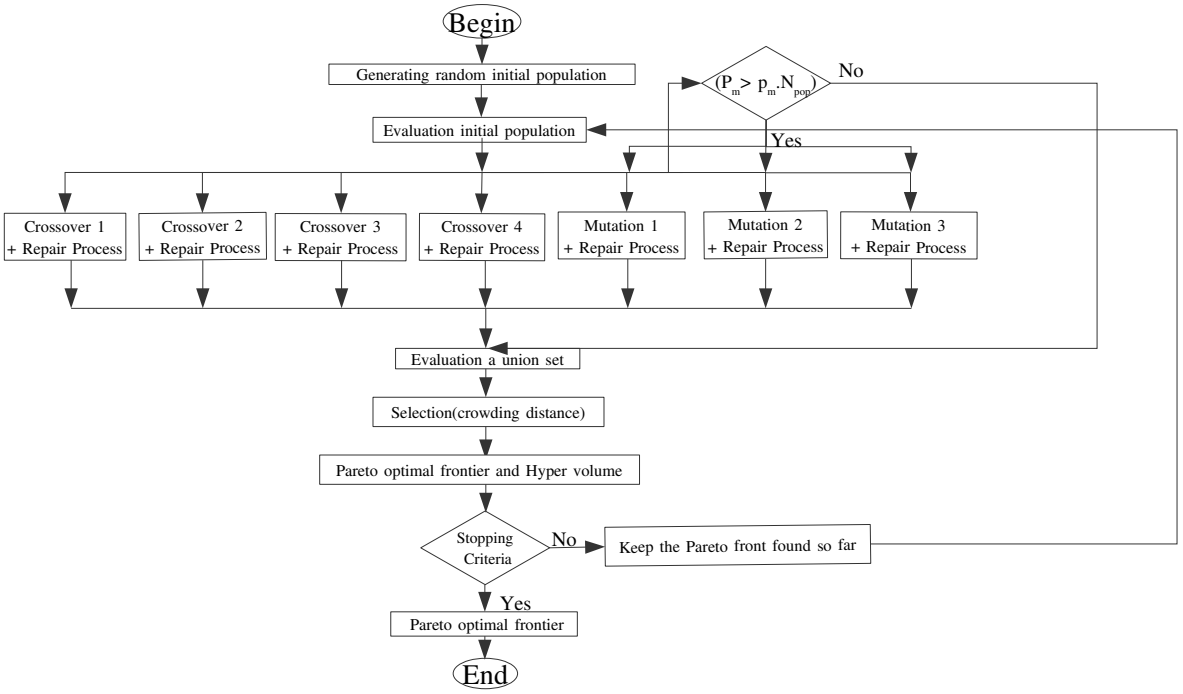

Figure 4: The flowchart of SH-NSGA-II for spatial zoning optimization problem

repeated until the crossover rate is reached. On the other hand, if the probability of mutation is satisfied, the same procedure as random selection in crossover is repeated for three mutation operators to produce offspring. Following that, all populations and the achieved offspring are grouped together to form a union set, which is then evaluated. Because the size of a union set should be the same as the beginning population, all zones may not be included in a union set at this point. As a consequence, the crowding distance approach is utilized to complete it by adding an incomplete front, in which the needed population is formed by the top of the front components without losing good solutions (elitism). Furthermore, it must attain the Pareto front in each iteration in order to compute the stop criteria, which is the number of non-improved Hypervolume

(HV) values and will be explained in detail in Subsection 2.5. As a result, if this stop requirement is not met, the final selected solution set is saved, and the next generation begins. Otherwise, the final Pareto front is stated. It should be noted that three notations p_m in Figure 4 represent the probability of mutation.

As mentioned in section 2, applying some strategy to empower the local search ability in MOEAs helps to reduce the likelihood of premature convergence. Therefore, we propose another MOEA as a hybrid SH-NSGA-II using local search (MA) to enforce and compare with the proposed SH-NSGA-II. The MA combines an evolutionary search-based optimization algorithm with a problem-specific local search to balance the exploration and exploitation of the algorithm and therefore improve the quality of the solution [12, 11]. MA has recently been shown to be useful and powerful in tackling difficult large-scale optimization issues [16, 32, 46].

The Algorithm 1 describes our proposed MA to solve the spatial zoning optimization problem. In this pseudo-code, two different colors, red and black, are used. The red lines added to SH-NSGA-II in black to make MA algorithm. Unlike traditional MA [10], the proposed one employs four crossover operators, three mutations, and a local search operator during the search process. The proposed MA added a local search strategy to SH-NSGA-II to improve its search efficiency.

As can be seen in the pseudo-code 1, the start point for both algorithms is similar, that is generating a random population with respect to the predefined population size (N_{pop}) and then it is evaluated. To begin with, all three groups of populations, including, crossover, mutation, and local search indicated by pop_{cross}, $pop_{mutation}$, pop_{local} in order, will start with initial population. Afterward, on the basis of the crossover rate, this operator begins to make a list of new offspring. In each turn, up to the crossover rate, a crossover (i) is randomly selected between four different crossover operators. In case of not being in the offspring list, the feasible generated offspring would be added. After making the offspring list of crossover, the mutation loop will be started by meeting the mutation probability. The initial population would be mated by three different mutation operators (i) which are selected randomly iteratively. Then, the output of them would be saved in the offspring list of mutation if it is not already there. Following that, the local search process would be launched by local probability. Unlike SH-NSGA-II, no binary tournament is used in the local search operator selection process. Instead, a lower selective pressure is applied to select a number of the population. By doing so, it ensures that each chromosome in the selected population gets an opportunity to pass on their genes to the next generation, promoting variety and avoiding premature convergence. Then, a union set would be updated by combining all gathered offspring from crossover, mutation, and local search operators. Next, using the crowding distance, the union set is adjusted to a certain size. Finally, the stop condition is the number of non-improved HV values (K) which will be explained in detail in Subsection 2.5, is calculated, and all population lists along with their evaluations are updated to check if to continue or stop the main loop.

2.3. Search Components

MOEAs employ different common search components (e.g., selection, variation operators (mutation and crossover), and replacement) as their major mechanism to carry out the evolutionary process. This section explains all the operators used in SH-NSGA-II and MA.

2.3.1. The Initialization Operators

The Pareto front can be generated more rapidly and produce more possible solutions with well-initialized populations, but the process is less efficient if the starting answers are poorly chosen. In spatial planning optimization issues, maps including existing activities and feasible areas to be found as the new activity should be incorporated into the iteration process, and initialization operators should generate 100% random solutions. Instead of checking distances on all possible grids, the utilized technique for generating the random population is similar to circle filling on a grid by bounding box, in which we save a lot of time by examining a much smaller region without looking at the rest of the grid. Compared to other basic algorithms, this may be able to construct compact zones with sufficient diversity that match all the restrictions listed in Section 2.1.

The following is a summary of the steps in this algorithm:

Algorithm 1 Pseudo-code MA

```
procedure MA(N_pop, R_c, R_m, P_m, R_l, P_l)
    pop ← random_population(N_pop)                          ▷ Create a random initial population set
    F ← evaluation_Fitness(pop)                             ▷ Evaluate the initial population
    pop_cross, pop_mutation, pop_local ← pop
    F_cross, F_mutation, F_local ← F
    while not stop criterion(counter < K) do
        while not crossover rate do
            C_i ← crossover_i(pop_cross, F_cross)           ▷ For i^th crossover of 4
            Offspring_cross_i ← non_repeat(C_i)             ▷ Clearing and Collecting offspring
        end while
        if mutation probability then
            while not mutation rate do
                M_i ← mutation_i(pop_mutation, F_mutation)  ▷ For i^th mutation of 3
                Offspring_mutation_i ← non_repeat(M_i)      ▷ Clearing and Collecting offspring
            end while
        end if
        if local probability then
            Offspring_local ← local_search(pop_local, F_local)
        end if
        pop ← insert(pop, pop_cross, pop_mutation, pop_local)   ▷ Create a union set
        F ← evaluation_Fitness(pop)                         ▷ Evaluate union set
        F_pareto, N_pareto ← pareto_front_finding(F, N_pop) ▷ Finding the pareto front
        hv ← HV(F_pareto, N_pareto)                         ▷ Calculating HV
        counter ← count(hv_list)                            ▷ Counting the non-improved HV value
        pop ← selection(N_pop, pop, F)                      ▷ Do selection among union set
        F ← evaluation_Fitness(pop)
        Updating pop_cross, pop_mutation, pop_local, F_cross, F_mutation, and F_local
    end while
    Reporting the final Pareto front based on the Crowding Distance
end procedure
```

1. The square bounding box must be defined first
2. Getting all the cells in this box together
3. Choosing cells that match all the following criteria
    - Starting with radius 1, being within the circle (if it is less or equal than the radius then mark it)
    - Being feasible
    - Being non-repetitive
4. Step 3 is repeated until the upper-bound solution size is reached, and the radius is increased to the maximum preset radius in each iteration (8)

As can be seen in Figure 5, it is a sample of circular fill of radius 3.5 with a bounding box. The values shown in Figure 5 are the centroid distances of the grid from the center yellow point. The bounding box is defined by the four bounds; left, right, bottom, and top. Having gathered all feasible cells within this rectangle, the checking approach begins. Here, we select the number of cells with respect to the solution size starting with radius 1. Among all the inside gathered (x, y)s, those are marked that not only meet the maximum radius distance restriction but also have yet-to-be selected (as shown in dark green). In order to increase the variety structure of the generated zones, another algorithm is used in which instead of filling the circle, the square surrounded this circle (the bounding box) is considered.

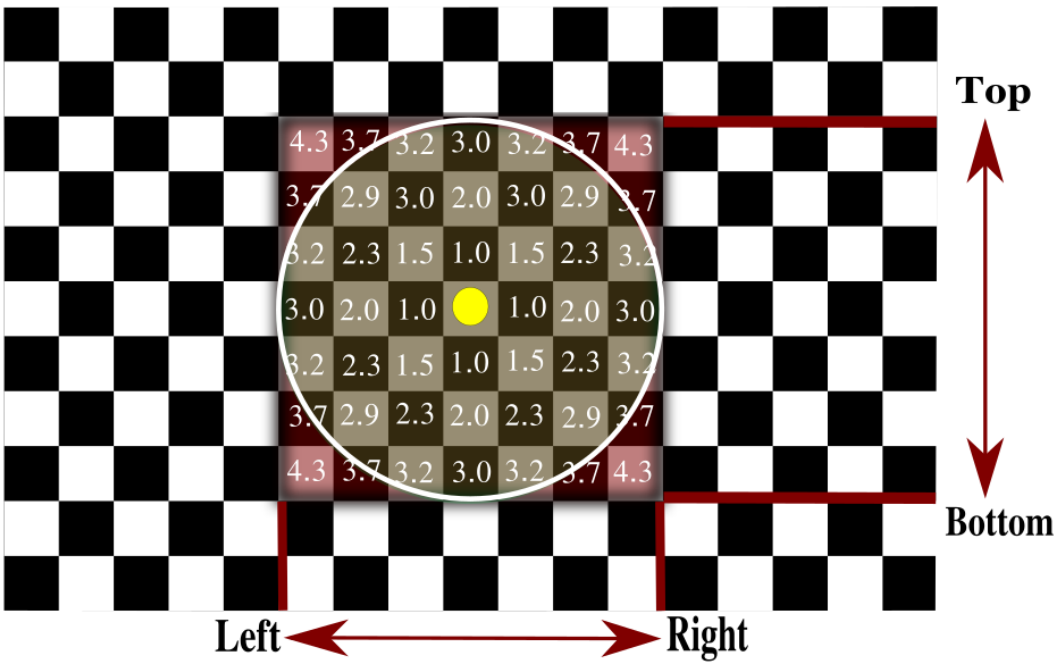

Figure 5: Random population using circle filling on a grid by bounding box

2.3.2. Crossover Operators

Two proposed MOEAs utilize four crossovers, namely Crossover-1 (2.3.2.1), Crossover-2 (2.3.2.2), Crossover-3 (2.3.2.3), and Crossover-4 (2.3.2.4), which are explained in details in this section. They are applied to three separate sections of the chromosome, to thoroughly investigate the search space of the problem. We must choose two parents as inputs for this operator for each crossover. As a result, binary tournament selection is chosen as a selection method to pick two selective parents [10]. The chromosomes of both parents are then encoded and sorted according to x-coordinate/y-coordinate.

Having selected a random crossover $\mathfrak{i}$ as mentioned on line 8 of the pseudo-code 1, the parent populations go through the tournament selection and two of them are selected. The selected parents are mated by the

selected crossover operator and generate maximum two offspring after passing the check and repair operator. The parents of the remained offspring are deleted from the parent population list. The updated parent list will be returned to the crossover loop, and this action is iterative until the crossover rate is reached.

2.3.2.1. Single-Point Vertical Cutting Crossover (Crossover-1).

In this crossover, two parents representing two zones, together with their encoded chromosomes, are represented in purple and yellow in Figure 6. Then, as shown in Figure 6, a cut-point cell is chosen at random along the length of each chromosome. The next step is to create the center cell of these two cut-point cells, which is labeled "**C**" in red. As a result, each parent is split into three sections: before cut-point, cut-point, and after cut-point.

This split is done vertically since both parents are ordered based on the x-coordinate to begin with, which is why this crossover is named "**Single-Point Vertical Cutting Crossover**". After locating the middle cell, the other two parts are vertically swapped and transformed to the new center point, i.e., the "**left-hand**" side of "**Parent-1**" and the "**right-hand**" side of "**Parent-2**"are shifted to the middle cell that forms "**Offspring-1**".

The "**left-hand**" side of "**Parent-2**" and the "**right-hand**" side of "**Parent-1**" are substituted with the identical middle cell, resulting in another offspring, "**Offspring-2**".

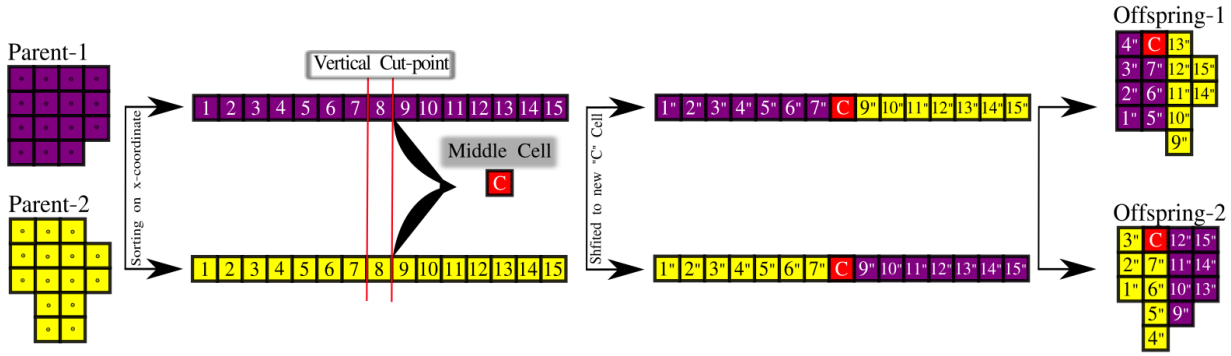

Figure 6: Single-point vertical cutting crossover (Crossover-1)

Figure 6 shows different numbers in the offspring chromosomes than those of parents because the coordinates of parent cells are changed by the new center point to other locations in the map. The output of each crossover is two offspring (children). By implementing this crossover leading to replacement through the map, some offspring may become infeasible due to the violation of certain constraints of the spatial zoning optimization problem. Therefore, it is needed to check and repair all offspring chromosomes to ensure their feasibility. This check and repair operator is explained in detail in Section 2.3.4. It is noted that four crossovers are not related, and they usually have different inputs but always unique outputs.

2.3.2.2. Single-Point Horizontal Cutting Crossover (Crossover-2).

The cutting direction is switched from vertical to horizontal, unlike Crossover-1. To put it another way, the y-coordinate is used to arrange two parent chromosomes. As shown in Figure 7, it should be noted that the random cut point in this example is the cell number (5). The graphical representation of this crossover is shown in Figure 7. Next, the middle cell of these two cut-point cells (number 5) is made, which is called "**C**" in red. Each parent is divided into three parts; before cut-point, cut-point, and after cut-point, respectively. As both parents are sorted based on y-coordinate, this division is done horizontally, that is the reason why this crossover is called "**Single-Point Horizontal Cutting Crossover**". Having found the middle cell, the other two parts are horizontally swapped and transformed to the new center point, that is, "**bottom**" of "**Parent-1**" and "**top**" of "**Parent-2**" are shifted to the middle cell that forms the "**Offspring-1**". On the other side, "**bottom**" of "**Parent-2**" and "**top**" of "**Parent-1**" are replaced to the same middle cell making another offspring, "**Offspring-2**".

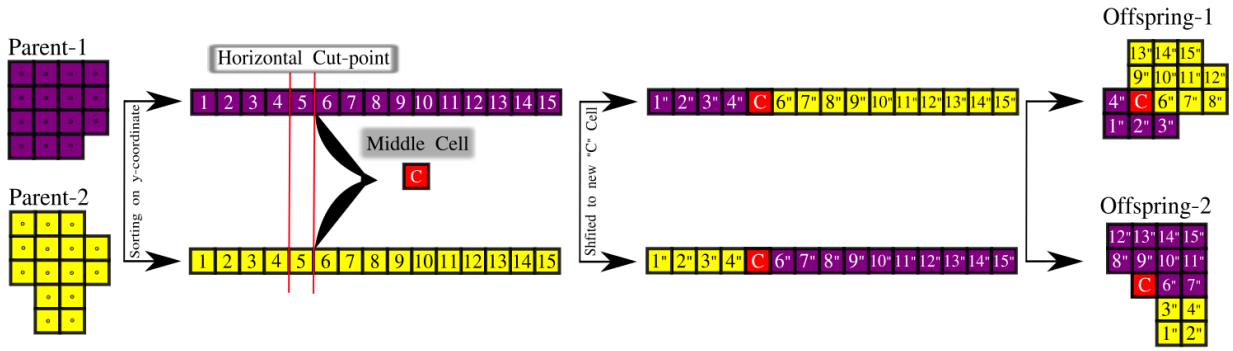

Figure 7: Single-point horizontal cutting crossover (Crossover-2)

2.3.2.3. Semi-Proportional Vertical Single-Point Cutting Crossover (Crossover-3).

The structure (shape and size) and position of each zone are the two fundamental concerns in the spatial zoning optimization problem, as far as can be deduced. That is why the fundamental goal of the two first crossovers is to create well-diversified offspring in the parent neighborhoods from a structural standpoint. However, the goal of the following two crossovers is to produce well-diversified offspring in close proximity to the parents. Therefore, Crossover-3 and Crossover-4 are built to focus on positioning the new activity on the highly interesting zones. Therefore, the cutting type in Crossover-3 and -4 is the same as Crossover-1 (vertical single-point cutting).

However, after selecting the cutting cells at random in the parents, the new rule is used to locate the cell "**C**" among the offspring. The method for locating the cell "**C**" is based on the first objective function, which is the interestingness value of parents.

However, after randomly selecting the cutting cells in the parents, the new rule is applied to find the cell "**C**" in the offspring. The approach to find the cell "**C**" is based on the first objective function, namely, the interestingness value of parents.

The difference between the initial objective function values of two parent chromosomes is calculated using the indicator "**proportion**" in Crossover-3. Following the proportional value, three situations are considered:

1. **Zero proportion**: When both parents' first objective function values are equivalent, the middle cell in the distance between the parents' selected cutting cells ($1/2 \times A$) is picked as the new "C" cell. The cells of the offspring are created by modifying the parent chromosomes, depending on the sort of cutting used here, which is vertical.
2. **Negative proportion**: When the initial objective function value of "**Parent-1**" is smaller than "**Parent-2**", a new "C" cell is drawn toward the "**Parent-2**" placed in the distance ($2/3 \times A$) from the "**Parent-1**", resulting in one of the offspring. The other offspring, on the other hand, is formed in the middle distance as previously. The rest of the procedure is identical to that of a zero proportion.
3. **Positive proportion**: The direction of the movement of the "C" cell is exactly the opposite of the negative proportion. Because the objective value of "**Parent-1**" is greater than that of the other. As a result, one of the children is drawn to the first parent, while the other stays in the center. The rest of the procedure remains unchanged.

For example, in Figure 8, the negative proportion is shown. As this crossover is called semi-proportional single-point vertical cutting crossover, the first offspring stays in the middle, and the other goes toward the parent with a higher first objective function (interestingness value).

2.3.2.4. Full-Proportional Vertical Single-Point Cutting Crossover (Crossover-4).

The only difference between Crossover-3 and -4 is that in case of positive and negative proportions, both offspring intend to get closer toward the parent with higher objective function. Therefore, we could call

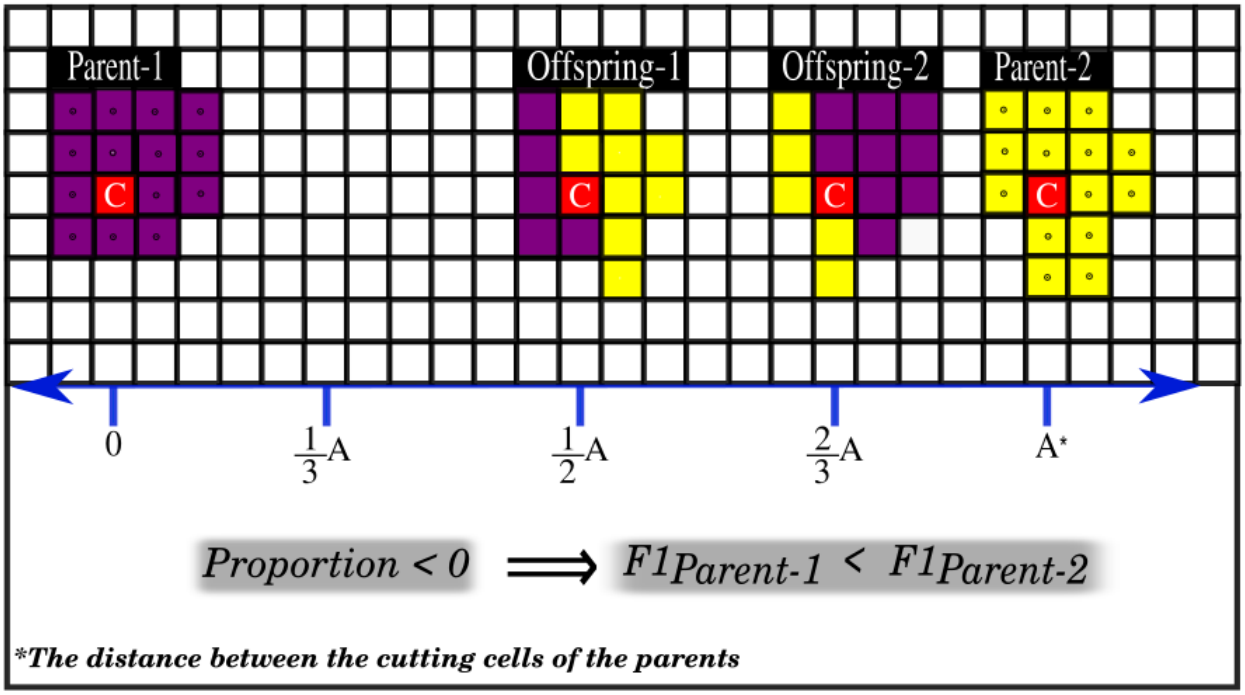

Figure 8: Semi-proportional vertical single-point cutting crossover Crossover-3

it as full-proportional vertical single-point cutting crossover. All 4 crossovers are implemented iteratively through a loop. In each iteration, the check and repair operator checks the feasibility of the offspring (explained in Section 2.3.4). If each offspring is validated, it will be added to the list of offspring. This insertion will continue until the crossover rate is reached.

2.3.3. Mutation Operators

Vertical and horizontal reconfiguration of the solutions are examined to employ well-diversified solutions around the parent chromosomes, as demonstrated in the crossover operators. Three mutation operators, on the other hand, are utilized to better explore the problem's search space. After a certain number of iterations within the main loop of the suggested MOEAs, the mutation operators will randomly be chosen. The proposed MOEAs employ three mutation operators, namely Mutation-1, Mutation-2, and Mutation-3. These operators start to be applied to each chromosome. The mutations' structure is based on a mix of four-directional motions (right, left, up, and down) and rotational symmetry of the solution across solution space (90°, 180°, 270°). Each mutation has two parent chromosomes as inputs, just like the crossover operators in Section 2.3.2 (binary tournament selection). Each chromosome is shifted to the other side of the search space in Mutation-1, with each parent's moving step between the list of four directions chosen at random. On the one hand, each gen coordinate (x, y) in each parent chromosome should be rotated counterclockwise by a specified angle around a given origin in Mutation-2, in addition to the 4-direction movement. In Mutation-3, however, the rotation process is the only factor that affects each parent chromosome. There could only be two offspring chromosomes after confirming the feasibility and correcting the result of each mutation.

2.3.4. Check and Repair Operators

Some challenges may arise throughout the solution development process. One challenge is that new solutions must be generated inside the possible solution space, not outside of it or in conflict with existing activity. Another challenge is recreating the solutions in a compact manner (*i.e.* without any hole). Two distinct check and repair operators, "**check-and-repair**" and "**compacity-improver**", are employed to fix

these difficulties. Three potential situations for the first issue and one scenario for the second issue might occur as follows:

- Scenarios pertaining to the first problem are as follows:
  1. The chromosomes are not included inside the solution space (map).
  2. The chromosomes are located within the solution space, but they overlap with the activities that already exist.
- Scenarios related to the second issue:
  1. Holes or discontinuities can be seen in the chromosomes.

The following are some possible solutions for each scenario.

- The following is the recommended solution for the first issue, which uses the "**check-and-repair**" operator:
  1. A random population generator generates and replaces a completely new chromosome.
  2. Counting the number of overlapping cells; if there are less than 5, the search procedure continues to look for alternatives in nearby cells while maintaining or expanding the compacity; otherwise, the cell is completely deleted. The overlapping cell's feasible and non-iterative 4-direction ((-1, 0), (0, 1), (0, -1), (1, 0) neighbors are collected.
- The following is the proposed solution for the second issue, which employs the "**compacity-improver**" operator:
  1. The "0-1" solution matrix is bound by the number "2".
  2. Examining the rows and columns for any "1" or solitary "0" completely encircled by "1".
  3. Removing the zero detected rows / columns and replacing the single encircled element "0" in the outer layer of the matrix with one of the possible elements "1".

The solutions mentioned for "**check-and-repair**" are coded as explained in pseudo-code 2 to repair infeasible chromosomes to meet all the constraints and requirements of the spatial zoning optimization problem. In pseudo-code 2, the first proposed solution is coded in red and the second one in blue. In line 16 of this Pseudo-code, all feasible and non-iterative 4-direction ((-1, 0), (0, 1), (0, -1), (1, 0)) neighbors of the overlapped cell are gathered.

Figure 9 is a given example to better understanding how the "**compacity-improver**" operator works. In Figure 9, one achieved solution with one hole and interruptions is shown in the shape "1". The "**compacity-improver**" first bounds the matrix of solution with value "2". Next, the operator starts detecting the rows and columns without "1" like the green row in matrix "2". After deleting all zero rows and columns, the initial solution turned into shape "3" with a hole. Afterwards, the operator makes a list of zeros with 2×1 or 4×1 connected components in their neighbors. Like blue zero in matrix "4" which is surrounded by 4×1 in red. Next, the zero is changed to 1 in orange as shown in matrix "5" and from the outer layer, one of the ones is changed to 0 in purple. Finally, the repaired shape is "6" which is well-compact.

All crossover, mutation, and local search operators include these two repairing operators in their bodies.

2.3.5. Local Search Operator

In addition to the crossovers and mutations which are used in both MOEAs, the suggested MA utilizes a local search operator that performs a tiny modification on a given solution to extensively search the neighborhood of that solution and boost the spatial zoning optimization problem convergence speed to the optimality. Using tournament selection, the parent chromosomes of the local search are chosen from the population with the size of the local search rate, and the offspring chromosomes are provided following the repair process.

Algorithm 2 Pseudo-code Check-and-Repair

```
procedure CHECK-AND-REPAIR(pop, area, feasible_cells)
    counter ← 0
    remove_list ← ∅
    check_list ← ∅
    for i in pop do
        for j in i do
            if j not in area then
                i ← random_population(N_pop)                ▷ Replacing the new chromosome
            else if j not in feasible_cells then
                counter ← counter + 1
                check_list ← j
            end if
        end for
        if 1 ⩽ counter ⩽ 4 then
            for k in check_list do
                neighbour ← get_adjacent(N_pop)
                if neighbour ≠ ∅ then
                    One of neighbor which leads to a more compact solution is chosen
                else
                    remove_list ← i
                end if
            end for
        else
            remove_list ← i
        end if
    end for
    pop ← delete(pop, remove_list)          ▷ Deleting the remove_list from the population list
end procedure
```

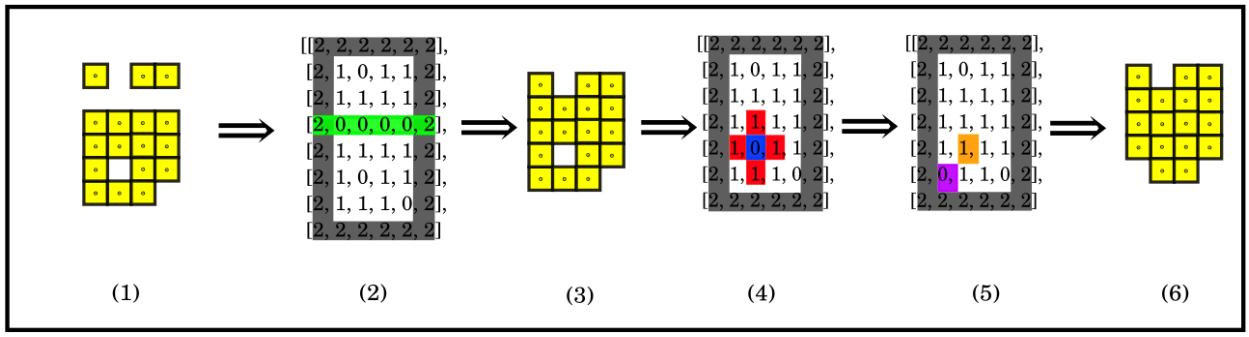

Figure 9: Compacity improver operator

This operator focuses on improving the interestingness values of the cells whose values are 2 units lower than the maximum value. Therefore, it starts to search for the best replacements in the neighbors of these cells. In this replacing process, the priority is on the higher value neighbors. However, it is noted that only if the 4-direction neighbors of the considered parent's gen are not in the parent chromosome, this replacement would be done. In other words, removing the selected gene bounded by other cells in the parent chromosome would result in the non-compacity and hole in the offspring chromosome. Next, among these neighbors, one of them is randomly selected as the replacement. The offspring chromosome of the local search may be infeasible because some constraints can be violated after employing the local search operator. Therefore, the local search requires employing the repair process.

2.4. Evaluation and Selection Operators

Two different objective functions are used to calculate the fitness value of each chromosome (solution quality). The first objective function is obtained by adding the interest values of the zone cells, and the compactness value is determined using the "**Normalized Discrete Compactness (NDC)**" metric, proposed by [42].

Each time, the population is classified into distinct non-dominance levels through the selection procedure. The fitness of any solution is equal to its level of non-dominance ("1" will be ascribed to the first non-dominated front). This procedure is for the minimization problems, but otherwise the maximization problems could be altered to minimization by multiplying by "-1".

This approach allows for simultaneous non-dominated sorting and filling of population steps based on crowding distance until the population size requirement is reached. As a consequence, each time, a non-dominated front finding operator was used to determine if the acquired solution could be included in the Pareto set. Otherwise, there is no reason to continue sorting. If the number of identified solutions exceeds the population size, the excess will be removed using the crowding-distance metric from the previous front that could not be fully accommodated.

When two solutions are compared, the crowded comparison operator gives the tournament winner. The winner is determined by two factors in the population: the non-dominance ranking r_i and the local crowding distance d_i. The search space surrounding a i^{th} solution in its front (marked with solid circles) that is not filled by any other solution in the population is measured by the crowding distance attribute of that solution. The perimeter of the cuboid produced by employing the nearest neighbors as vertices (solid circles $(i-1)$ and $(i+1)$) is estimated as d_i, shown in Figure 10 by the dashed box (called the crowding distance). The binary crowding tournament selection operator, which is based on r_i and d_i, operates as follows: A solution i wins a tournament over a solution j if any of the following criteria are true:

1. If $r_i < r_j$, the chosen solution is on a better non-dominated front.
2. If $r_i = r_j$ and $d_i > d_j$ (When both solutions are on the same front and the criteria above cannot be met, this is used; in this situation, the solution that is located in a less congested region and has a bigger d_i , wins).

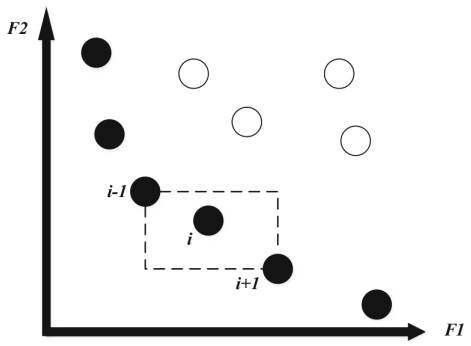

Figure 10: Crowding-distance calculation. Points marked in filled circles are solutions of the same non-dominated front [4]

2.5. Stop Condition of MOEAs

Different termination criteria may be utilized in MOEAs, including 1) a predetermined amount of iterations, and 2) convergence to a solution of a specified quality [38, 35]. We created a novel stop criterion that may alleviate some shortcomings of duplicate generations while also lowering the ratio of solution quality to processing time. This condition is used to halt the operation after the algorithm has completed a particular number of iterations without improving. It is based on the HV value, MOEAs Pareto set diversity, and convergence control measure, over a set of iterations. For multi-objective issues, HV is a well-known performance metric. It adheres to the Pareto principle and is based on the volume difference between a predetermined reference point and the solution offered. As a result, the HV necessitates the establishment of a reference point that is greater than the Pareto front's maximum value named by r in figure 11 [7]. It determines the area / volume dominated by the set of solutions provided in relation to a reference point [31, 17].

The figure 11 depicts a two-objective example, in which the area dominated by a set of points ($p^{(1)}$,$p^{(2)}$, and $p^{(3)}$) is shown in gray. Whereas the goal with this metric is to increase the distance to the reference point, to maximize its performance. The more HV value, the less distance to the Pareto front is.

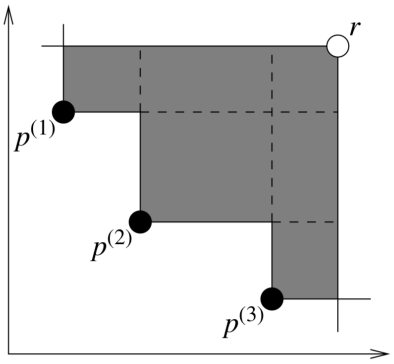

Figure 11: HV indicator for a non-dominated approximation set of solutions [14]

This metric is also used for the stop criterion in the proposed MOEAs algorithms. That is, at the end of each iteration in both algorithms, the HV of the optimal Pareto front is calculated and compared with that of the previous iteration. Then, the number of non-improved HV values in each iteration is determined. If this number violates the predefined maximum bound, the main loop of each MOEA will be stopped and the optimal Pareto front will be returned. The approach to define this upper bound is explained in Subsection 2.5.

3. Tuning parameters

To be able to compare the performance of the two MOEAs, it is needed to tune their parameters. However, before that, a DOE approach is used to explore the impacts of the components. Next, the MOEAs parameters, which have a substantial influence on the quality of the solution explored and then optimized by using the response surface methodology (RSM). RSM contributes to the improvement and optimization of processes by creating an analytical link between the input and result variables in experiments. However, most previous RSM-based solutions focused on single-response problems, with multi-response scenarios receiving less attention [39]. Therefore, in this section the appropriate tuning approach for our problem is explained (3.1) and the final obtained values of tuned parameters are reported (3.2).

3.1. Multi-Response Surface Methodology (MRSM) Optimization

According to researches, the optimal factor settings for one performance feature are not always consistent with those for other performance qualities. Finding compromising circumstances in input variables that are moderately favorable to all responses might be addressed in more general situations [24]. More details on RSM related design and optimization of response surfaces are given in [23] and [30].

However, in order to determine the substantial factors, MRSM developed a special type of fractional factorial experiments (FFEs) to reduce the large number of experiments required in it [22]. BBD with one central point is used to run the experiments, even though the responses may have curvature over the search ranges of the factors [30]. There are $k = 4$ factors for SH-NSGA-II and $k = 6$ factors for MA, each with three levels, i.e. low, medium, and high, and each signed by -1, 0, and $+1$, respectively. The data generation parameters of the spatial zoning optimization problem are summarized in Table 1. We vary these problem-specific parameters in three different levels (low, medium, and high) for three different map sizes (55×55, 300×300, 1000×1000) as shown in Table 1. The total number of artificial datasets for each map size that have been generated are $3^4 = 81$. Among 81 datasets, 8 of them are randomly selected to be used in all evaluations for each map size. Moreover, the anonymous link where anyone with the link can view all coded algorithms along with all used data is provided in the repository [3]. Using the data in Table 1, the coded MOEAs were executed based on the BBD for four factors in three levels with a center point for SH-NSGA-II and six factors in three levels with a center point for MA, shown in Table 2.

Table 1: Data generation parameters

Parameter names	Possible values	Description
n_{row}	55, 300, 1000	Number of rows of the raster grid
n_{col}	55, 300, 1000	Number of columns of the raster grid
n_p	6, 8, 10	Number of ports
n_s	6, 7, 8	Number of shipping lanes
n_a	3, 4, 5	Number of protected area
n_w	2, 3, 4	Number of windmill farms

Multi-response optimization problems (MROPs) have been examined from a variety of perspectives and can be divided into three groups, Desirability viewpoints, Priority based methods, and Loss function [20]: The third category of MRSM (Loss function) achieves a balance between resilience and optimization for multiple response issues by incorporating some well-established methods in it, such as GA, Artificial Neural Network (ANN), Taguchi loss function, and desirability function. So, we create a hybrid technique that uses the Taguchi method's loss function to compact and calculate multi-responses. In this paper, all response values (multi-objectives) of MROP are aggregated and converted to a single one using the Taguchi loss function.

In the Taguchi method's loss function, there are two sorts of components: noise factors N and controllable factors S. In this study, the signal-to-noise ratio (S/N) is used to examine the findings, as MOEAs have

Table 2: Search range of algorithm parameters

Algorithm	Actual Values	Coded Values	Low(-1)	Medium(0)	High(+1)
SH-NSGA-II	Population_size (N_{pop})	x_1	100	150	200
	Crossover_rate (R_c)	x_2	0,4	0,6	0,8
	Mutation_rate (R_m)	x_3	0,1	0,4	0,7
	Mutation_probability (P_m)	x_4	0,25	0,5	0,75
MA	Population_size (N_{pop})	x_1	100	150	200
	Crossover_rate (R_c)	x_2	0,4	0,6	0,8
	Mutation_rate (R_m)	x_3	0,1	0,4	0,7
	Mutation_probability (P_m)	x_4	0,25	0,5	0,75
	Local_rate (R_l)	x_5	0,1	0,5	0,8
	Local_probability (P_l)	x_6	0,45	0,6	0,8

numerous runs to acquire better answers. Under varying noise situations, the signal-to-noise ratio evaluates how the response fluctuates in relation to the goal value.

Three metrics are used to assess MOEAs in this article: HV, number of Pareto solutions (NPS), and best solution (Best Sol). NPS metric presents the number of Pareto optimal solutions that are obtained by each algorithm. For each set of solutions, the values of both objective functions, each weighted 0.5, are put together to identify the best solution. After that, Best Sol is picked as the best answer, the maximum (similar to how the simple additive weighting algorithm (SAWA) in multi-criteria decision-making (MCDM) handles [45]. Because the goal is to maximize efficiency, the higher the HV, NPS, and Best Sol values, the better.

There are four different formulations to calculate signal-to-noise ratios[19]. Following the objective of our experiment, we selected the first type in which for the signal-to-noise ratio, the larger is better, whose aim is to get the maximum S/N determined in Eq.2:

$$\frac{S}{N} = -10\log(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{sum_i^2}) \tag{2}$$

Where sum_i is the response in the Taguchi method, and n is the number of replications ($n = 3$). S/N is the MRSM response. Since the largest S/N value corresponds to the optimal combination of parameter values, this response should be maximized. A regression equation can be used to determine the relevance of individual process factors and their interactions. It calculates the relationship between the response and the parameters of the input process.

To estimate the response functions, the developed algorithms and experimental DOE tests are programmed in Python 3.8 and *R* version 4.1.2, respectively. All experimental tests are done on an OpenStack virtual machine running Linux/Ubuntu 20.04.1 LTS with 20 VCPU, 10 GB disk, and 30 GB RAM.

Then, the response function is estimated and optimized using MRSM. Furthermore, for each MOEA in each size, the design should fit the second-order regression model (a quadratic model), that is, the one containing squared terms, product of two factors, linear terms, and an intercept. To find the subset of variables in the dataset resulting in the best performing model, that is, a model that lowers prediction error, the feature selection technique is applied to iteratively add and remove predictors in the predictive model.

In this article, we employed a feature selection strategy for all regression models that combined stepwise regression and cross-validation to produce the highest performing model. All final models are solved using the coded parameters, and the algorithm finds the best combination of parameters (a stationary point in the original units).

F-value and p-value (significant probability value less than *0*.05) were used to determine the statistical significance. The degree of fit of the polynomial model was assessed using the determination of coefficients R-squared, modified R-squared, p-value, and the acceptable stationary point in the original units, all based on the ANOVA findings. Finally, the best combination is found, which is the stationary point based on the

response surface model that was produced.

By means of best fitted regression models, the interactions of important factors on the response surface were evaluated. Finally, the optimal combination is obtained, which is the stationary point based on the generated response surface model. One caution about DOE and MRSM is related to the extrapolation of the stationary point, that is, whether this point is outside the experimental space or not? In other words, since a quadratic model will always show an optimum point, its accuracy depends on the accuracy of the model. Therefore, to validate the best MRSM model, it is necessary to make a balance between the accuracy of optimal point and regression model at the same time. Figure 12 represents an example of response surface 3D plots between population size and mutation rate, population size and mutation probability, mutation rate and crossover rate, and mutation probability and crossover rate for the SH-NSGA-II algorithm for small size maps. The graphs show the significant impact on S/N.

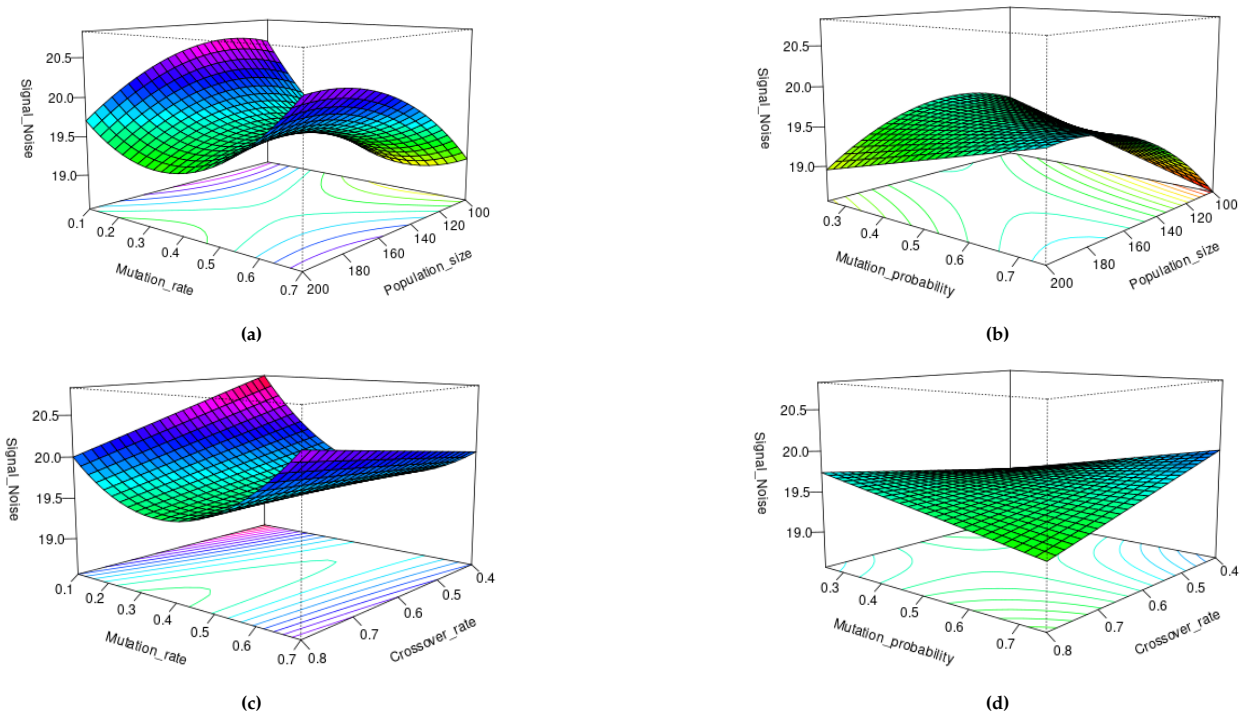

Figure 12: The response surface 3D plots of interaction of important factors on S/N: (a) interaction of population size and mutation rate; (b) interaction of population size and mutation probability; (c) interaction of mutation rate and crossover rate; (d) interaction of mutation probability and crossover rate

From Figure 12a, the contour plot shows that the mutation rate around 0.4 and population size between 150 and 160 led to the best value for S/N. By doing so for all other three plots, we can easily figure out that the obtained stationary point is driven correctly, and it is compatible with and in the range of all interactions between important factors.

3.2. Final Tuned Parameters

The optimal values of the parameters related to the stationary point are tuned as shown in Table 3 after finding the best fitted MRSM regression model for three distinct map sizes for each MOEA. The parameter setting may vary depending on the magnitude of the challenge in terms of the size of the map. As a result, we fit a distinct MRSM model to each map size, resulting in various adjusted parameters.

As we divided our spatial zoning optimization problem into three different size levels, for each of them, this stop criterion should be defined. To discover the ideal value for these criteria, we run 8 various problems

Table 3: Tuned parameters for SH-NSGA-II and MA

Solving Methodologies	Parameters	Size		
		Small	Medium	Large
SH-NSGA-II	N_{pop}	157	179	150
	R_c	0,6	0,67	0,69
	R_m	0,44	0,43	0,46
	P_m	0,5	0,5	0,5
MA	N_{pop}	150	163	124
	R_c	0,55	0,6	0,6
	R_m	0,4	0,5429	0,4
	P_m	0,5	0,5	0,583
	R_l	0,776	0,586	0,4
	P_l	0,667	0,625	0,64

with a set number of repetitions 3000 to examine the trend of HV value for estimate for each size of both methods. The average number of repetitions without improvement for each size is then used as a stop condition.

Table. 4 declares the stop condition value for the MOEAs in each size.

Table 4: Stopping Condition for SH-NSGA-II and MA

Solving Methodologies	Size		
	Small	Medium	Large
SH-NSGA-II	600	600	600
MA	400	500	500

4. Experimental Validation

Once the parameters have been tuned, the goal is to compare the performances of SH-NSGA-II vs. MA. To evaluate both methods, 8 different randomly selected datasets are used, as presented in table 1. Then, to evaluate a much more robust comparison, each instance is implemented 30 times and each reported value is the median of 30 runs of each problem with its respective method. Before presenting the results, the selected performance metrics are presented in the subsection 4.1.

4.1. Performance Measures

The fact that the result of the optimization process is a set of solutions representing an approximation of the Pareto front, rather than a single solution, is a major challenge in multi-objective optimization assessment. Because of the conflicting nature of the Pareto set solutions, we need to utilize certain performance measurements to evaluate the given methods [6]. We cannot tell if the algorithm has converged to the exact optimum until we know the Pareto-front. However, we can see when the algorithm has made the most progress during optimization and, as a result, whether the number of iterations should be reduced or increased. Additionally, the measurements allow the two algorithms to be compared against each other. To assess the performance of different multi-objective MHs, non-dominated sets of solutions must be compared [38]. Although various measures for non-dominated sets have been proposed, there is no universally acknowledged performance evaluation standard. To categorize quality indicators, many attributes can be employed.

The quality indicators are listed in the order in which they meet two separate performance goals: 1) convergence to the ideal Pareto front and 2) diversity of alternatives along the front. They are usually based on metrics of cardinality, distance, or volume. Cardinality-based indicators necessitate a limited approximation of the Pareto set. In general, distance-based indicators are sensitive to the scope of the objectives. As a result, all objective magnitudes must be standardized. It should be noted that relying solely on one quality indicator is invariably insufficient. From each class of measurements, at least one indicator must be chosen. Therefore, in addition to the metrics described in Section 3.1, the iteration number and three other performance metrics are used to evaluate and compare both MOEA algorithms. *NPS*, *BestSol*, and *HV* are the diversity-based, convergence-based, and hybrid categories of quality indicators, respectively.

The higher the three other performance criteria, the greater the quality of the solution we have. ***Mean Ideal Distance (MID):***
This measure depicts the proximity of the Pareto solution to the ideal point (0, 0), which is a convergence-based indicator as given in Eq. 3:

$$\mathrm{MID} = \frac{\sum_{i=1}^{n} c_i}{n} \tag{3}$$

where n is the number of the non-dominated set and $c_i = \sqrt{{f_{1i}}^2 + {f_{2i}}^2}$, and f_{1i}, f_{2i} are the value of the non-dominated solution of i^{th} for the first and second objective functions, respectively.

Spread of non-dominance solution (SNS):
A diversity-based metric that analyzes the uniformity of the generated solution distribution in terms of dispersion and extension is the spread of a non-dominance solution. The formula for this indicator may be found in Eq. 4.

$$\mathrm{SNS} = \sqrt{\frac{\sum_{i=1}^{n} (\mathrm{MID} - c_i)^2}{n-1}} \tag{4}$$

The rate of achievement to two objectives simultaneously (RAS):
The balance in reaching to objective functions is another convergence-based quality metrics. In the following Eq. 5 $F_i = \min(f_{1i}, f_{2i})$.

$$\mathrm{RAS} = \frac{\sum_{i=1}^{n} |\frac{f_{1i} - F_i}{F_i}| + |\frac{f_{2i} - F_i}{F_i}|}{n} \tag{5}$$

5. Computational Results

Therefore, this section investigates the effectiveness of the proposed MA algorithm using defined indicators. To analyze these two MOEAs with respect to each indicator, we did the **Wilcoxon signed-rank test (WSRT)** (paired samples) tests to check the null hypothesis that MA algorithm works better than SH-NSGA-II with respect to each indicator. These conclusions are supported by significant Wilcoxon tests ($p - value < 0.05$).

The WSRT tests the null hypothesis that two related paired samples come from the same distribution. In particular, it tests whether the distribution of the differences $(x - y)$ is symmetric about zero. It is a non-parametric version of the paired T-test.

To prove the validity of the algorithms, we need to show the gap between the optimal and MOEAs solutions. As the solutions for the small size from the exact method are available, we could do the validation for this size. Table 5 shows the performance indicators with respect to the exact methods and two MOEAs. As shown in Table 5 according to each measure, although both MOEAs achieved promising values and pretty well close to exact solutions, MA has less gap than SH-NSGA-II with exact and optimal solutions in small size. In figures 13, the differences between all three methods are shown.

Table 5: The comparison between MOEAs and the optimal solution

Problems	Map Size	HV			Best Sol			MID			SNS			RAS		
		Exact	SH-NSGA-II	MA	Exact	SH-NSGA-II	MA	Exact	SH-NSGA-II	MA	Exact	SH-NSGA-II	MA	Exact	SH-NSGA-II	MA
1	**Small**	5,866	5,5173	5,7448	0,563	0,5315	0,5462	78,859	76,0040	76,4613	3,038	2,215	2,984	0,995	0,9903	0,9913
2		6,310	5,7088	6,3004	0,557	0,5210	0,5570	85,126	79,5036	83,0024	4,936	1,730	4,455	0,996	0,9909	0,9940
3		6,313	5,6567	6,3004	0,558	0,5205	0,5570	84,002	79,0040	83,1690	6,725	1,368	4,627	0,995	0,9904	0,9940
4		6,310	5,7088	6,3034	0,557	0,5214	0,5570	85,112	79,1463	83,0024	5,039	1,891	4,553	0,996	0,9904	0,9940
5		6,316	5,7119	6,3004	0,557	0,5220	0,5570	85,126	79,3371	83,3357	4,814	2,060	4,455	0,996	0,9901	0,9940
6		6,313	5,7149	6,3004	0,557	0,5205	0,5570	85,751	79,3374	83,3357	5,158	1,869	4,502	0,996	0,9904	0,9940
7		6,310	5,6299	6,3004	0,558	0,5205	0,5570	85,223	79,0040	83,0023	5,606	2,060	4,231	0,996	0,9901	0,9939
8		6,310	5,7149	6,3004	0,557	0,5181	0,5570	85,112	79,3368	83,0024	5,314	2,079	4,293	0,996	0,9901	0,9939
Average		**6,2556**	**5,6704**	**6,2313**	**0,5580**	**0,5219**	**0,5557**	**84,2891**	**78,8342**	**82,2889**	**5,0789**	**1,9090**	**4,2625**	**0,9958**	**0,9903**	**0,9936**

Finally, in addition to small size, for the other two problem sizes, the results of both MOEAs with respect to the performance metrics are shown in Table 6. For each set of problems, seven performance metrics are investigated which show execution improvement through MA algorithm. MA outperforms SH-NSGA-II by taking all mentioned metrics into account.

Table 6: The Result of MOEAs

Problems	Map Size	HV		Best Sol		MID		SNS		RAS	
		SH-NSGA-II	MA	SH-NSGA-II	MA	SH-NSGA-II	MA	SH-NSGA-II	MA	SH-NSGA-II	MA
1	Medium	5,7088	6,1241	0,5163	0,5441	78,4038	81,6018	0,8989	2,6722	0,9901	0,9946
2		5,7088	5,8651	0,5196	0,5330	77,8921	78,2774	1,5506	2,1082	0,9901	0,9897
3		5,7088	6,1287	0,5172	0,5450	78,3789	81,2689	0,8721	2,9073	0,9901	0,9937
4		5,7057	6,1333	0,5154	0,5450	78,4323	81,0321	0,8692	2,9057	0,9901	0,9932
5		5,7057	6,1272	0,5154	0,5450	78,4038	81,4758	0,9147	2,8806	0,9901	0,9940
6		5,8636	5,8513	0,5356	0,5255	78,4328	77,3799	2,5409	1,6422	0,9898	0,9888
7		5,7057	6,1303	0,5154	0,5450	78,3675	81,2245	0,8647	2,9648	0,9899	0,9938
8		5,6451	6,1015	0,5163	0,5436	78,3789	80,7083	0,8421	2,8663	0,9899	0,9934
Average		**5,7190**	**6,0577**	**0,5189**	**0,5408**	**78,3363**	**80,3711**	**1,1691**	**2,6184**	**0,9900**	**0,9927**
1	Large	5,8958	6,1704	0,5129	0,5215	80,6714	82,8363	1,2377	2,0048	0,9901	0,9911
2		5,9640	6,1187	0,5145	0,5201	80,6714	81,2543	1,4120	1,3426	0,9897	0,9902
3		5,8881	6,0367	0,5168	0,5198	80,2547	81,2951	1,1655	1,5255	0,9895	0,9905
4		5,8881	6,2283	0,5155	0,5364	80,6714	83,1792	1,2235	2,6565	0,9894	0,9911
5		5,8138	6,1126	0,5054	0,5254	80,6463	82,2531	0,4633	1,9388	0,9888	0,9914
6		5,8850	6,0065	0,5129	0,5211	80,3623	80,8038	1,0260	1,7866	0,9895	0,9910
7		5,8942	6,3204	0,5103	0,5570	81,0039	81,8038	1,0945	2,7866	0,9900	0,9940
8		5,8881	5,9352	0,5103	0,5235	80,5050	80,9838	0,8421	1,7966	0,9897	0,9906
Average		**5,8896**	**6,1161**	**0,5123**	**0,5281**	**80,5983**	**81,8012**	**1,0581**	**1,9798**	**0,9896**	**0,9912**

Additionally, for small, medium, and large scale problems, with respect to the computational time MA is almost 50%, 29%, and 17% faster than the extended SH-NSGA-II with higher solution convergence quality.

When a MOEA is used to solve a problem, it generates a set of Pareto solutions from which a decision maker (DM) can choose the best one. Because this decision is similar to those made in multi-attribute decision-making (MADM) problems, one of the methods used in MADM is the technique for order preference, which is similar to an ideal solution (TOPSIS) [21], the fuzzy hierarchical TOPSIS [41], simple additive weighting (SAW) [29], the fuzzy SAW [26], and the linear programming technique for multidimensional analysis.

6. Conclusion

The spatial zoning optimization problem, which can be found in a variety of domains, including MSP, is defined in this paper. Two novel P-metaheuristics methods based on GA have been developed to solve

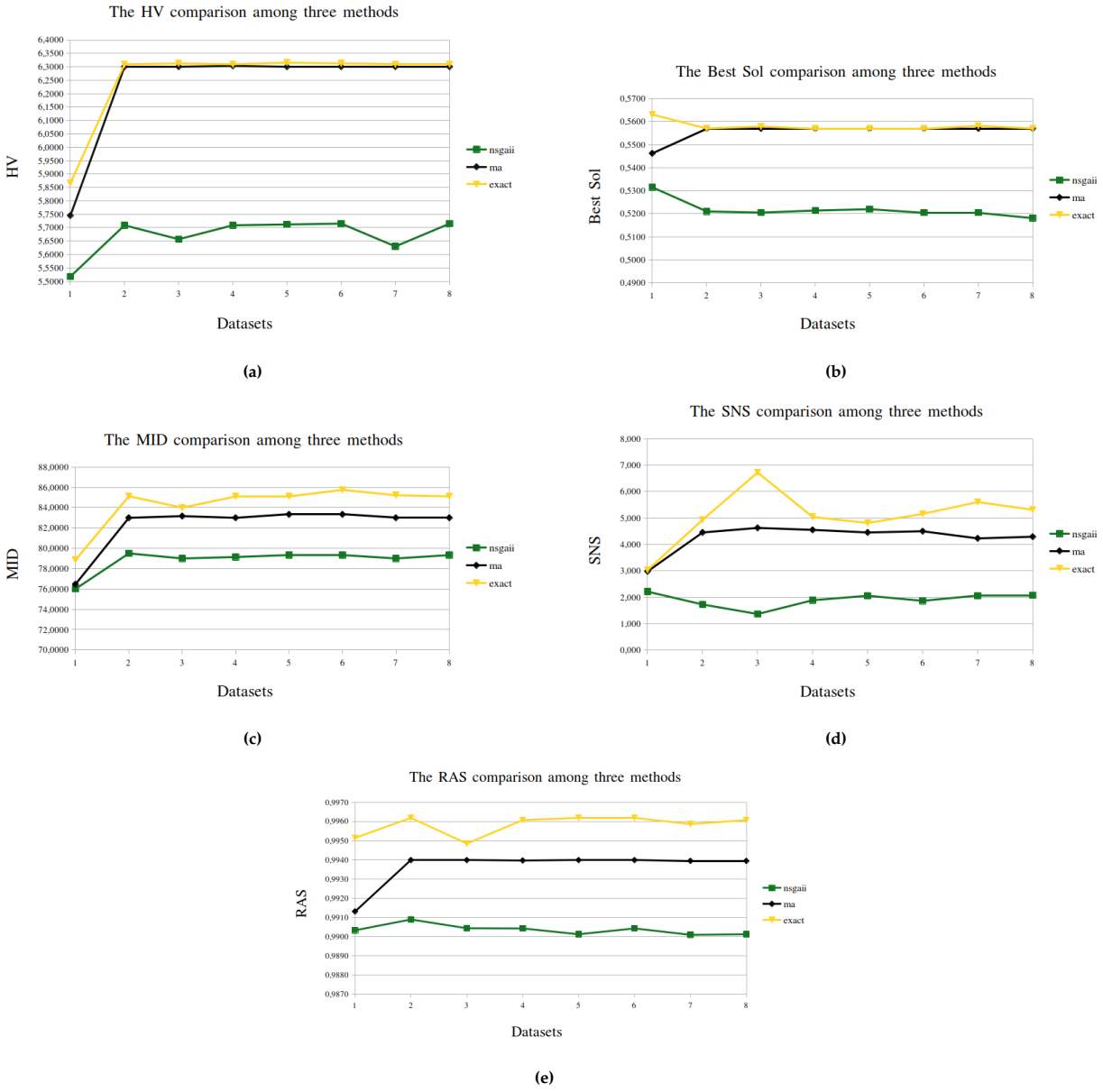

Figure 13: The comparison among MOEAs algorithms and exact optimal solutions for small size

this problem (SH-NSGA-II and MA). To demonstrate the outperformance of the proposed MOEAs, 24 test cases with 30 times replications were used. The results show that, on average, the proposed MA provided better solutions in less computational time, and that, when compared to SH-NSGA-II, the proposed MA has better consistency. Finally, a set of (24×30) Wilcoxon Signed-Rank tests revealed that the proposed MA outperforms the SH-NSGA-II significantly. Although, these findings has gone some ways towards solving the given specific problem in large-scale, more improvements could be possible to reduce the computational time while increasing the convergence speed. Moreover, some improvements in population generator operators could help in this regard to diminish the need of repairing operators. A further study could assess and develop more MOEAs to compare with the current proposed ones. Finally, we will investigate the spatial zoning optimization problem applicability in the real world and test the robustness of the proposed

MA on more complex spatial data with multi-agents in the future.

7. Conflict Of Interest

All authors declare that they have no conflicts of interest.

8. Data Availability Statement

The anonymous link where anyone with the link can view all coded algorithms along with the datasets generated during and/or analyzed during the current study is provided in the *MH-Single-MSP* repository [3].

References

[1] J. C. Aerts, M. v. Herwijnen, and T. J. Stewart. Using simulated annealing and spatial goal programming for solving a multi site land use allocation problem. In *International conference on evolutionary multi-criterion optimization*, pages 448–463. Springer, 2003.

[2] T. S. Agardy. Marine protected areas and ocean planning. In *Routledge handbook of ocean resources and management*, pages 476–492. Routledge, 2015.

[3] M. Basirati. Mh-single-msp, Jun 2022. URL `https://osf.io/dx7z8/?view_only=f27d1a89a5ae49439f2dd57687735721`.

[4] M. Basirati, M. R. Akbari Jokar, and E. Hassannayebi. Bi-objective optimization approaches to many-to-many hub location routing with distance balancing and hard time window. *Neural Computing and Applications*, 32(17):13267–13288, 2020.

[5] M. Basirati, R. Billot, P. Meyer, and E. Bocher. Exact zoning optimization model for marine spatial planning (msp). *Frontiers in Marine Science*, 8, 2021.

[6] J. Blank and K. Deb. Pymoo: Multi-objective optimization in python. *IEEE Access*, 8:89497–89509, 2020. doi: 10.1109/ACCESS.2020.2990567.

[7] Y. Cao, B. J. Smucker, and T. J. Robinson. On using the hypervolume indicator to compare pareto fronts: Applications to multi-criteria optimal experimental design. *Journal of Statistical Planning and Inference*, 160:60–74, 2015.

[8] R. Dahl, I. O. Commission, et al. Marine spatial planning: a step-by-step approach toward ecosystem-based management. *Paris (France) UNESCO/IOC*, 2009.

[9] K. Deb. Multi-objective optimisation using evolutionary algorithms: an introduction. In *Multi-objective evolutionary optimisation for product design and manufacturing*, pages 3–34. Springer, 2011.

[10] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In *International conference on parallel problem solving from nature*, pages 849–858. Springer, 2000.

[11] J. Decerle, O. Grunder, A. H. El Hassani, and O. Barakat. A memetic algorithm for a home health care routing and scheduling problem. *Operations research for health care*, 16:59–71, 2018.

[12] J. Deng and L. Wang. A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem. *Swarm and evolutionary computation*, 32:121–131, 2017.

[13] I. S. Doolun, S. Ponnambalam, N. Subramanian, and K. G. Data driven hybrid evolutionary analytical approach for multi objective location allocation decisions: Automotive green supply chain empirical evidence. *Computers & Operations Research*, 98:265–283, 2018. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2018.01.008. URL `https://www.sciencedirect.com/science/article/pii/S030505481830008X`.

[14] C. M. Fonseca, L. Paquete, and M. López-Ibánez. An improved dimension-sweep algorithm for the hypervolume indicator. In *2006 IEEE international conference on evolutionary computation*, pages 1157–1163. IEEE, 2006.

[15] K. Gokbayrak and A. S. Kocaman. A distance-limited continuous location-allocation problem for spatial planning of decentralized systems. *Computers & Operations Research*, 88:15–29, 2017. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2017.06.013. URL `https://www.sciencedirect.com/science/article/pii/S030505481730151X`.

[16] G. Gong, Q. Deng, R. Chiong, X. Gong, and H. Huang. An effective memetic algorithm for multi-objective job-shop scheduling. *Knowledge-Based Systems*, 182:104840, 2019.

[17] A. P. Guerreiro, V. Manquinho, and J. R. Figueira. Exact hypervolume subset selection through incremental computations. *Computers & Operations Research*, 136:105471, 2021. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2021.105471. URL `https://www.sciencedirect.com/science/article/pii/S0305054821002215`.

[18] M. J. Gwaleba and U. E. Chigbu. Participation in property formation: Insights from land-use planning in an informal urban settlement in tanzania. *Land Use Policy*, 92:104482, 2020.

[19] N. A. Heckert, J. J. Filliben, C. M. Croarkin, B. Hembree, W. F. Guthrie, P. Tobias, J. Prinz, et al. Handbook 151: Nist/sematech e-handbook of statistical methods. In *e-Handbook of Statistical Methods*, pages 2–p, 2002.

[20] T. H. Hejazi, M. Bashiri, J. A. Dı, K. Noghondarian, et al. Optimization of probabilistic multiple response surfaces. *Applied Mathematical Modelling*, 36(3):1275–1285, 2012.

[21] D. N. Jayakumar and P. Venkatesh. Glowworm swarm optimization algorithm with topsis for solving multiple objective environmental economic dispatch problem. *Applied Soft Computing*, 23:375–386, 2014.

[22] J. R. Karmoker, I. Hasan, N. Ahmed, M. Saifuddin, and M. S. Reza. Development and optimization of acyclovir loaded mucoadhesive microspheres by box–behnken design. *Dhaka University Journal of Pharmaceutical Sciences*, 18(1):1–12, 2019.

[23] J. P. Kleijnen. Response surface methodology for constrained simulation optimization: An overview. *Simulation Modelling Practice and Theory*, 16(1):50–64, 2008.

[24] O. Köksoy. A nonlinear programming solution to robust multi-response quality problem. *Applied mathematics and computation*, 196 (2):603–612, 2008.

[25] Y. Levi, S. Bekhor, and Y. Rosenfeld. A multi-objective optimization model for urban planning: The case of a very large floating structure. *Transportation Research Part C: Emerging Technologies*, 98:85–100, 2019. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2018.11.013. URL https://www.sciencedirect.com/science/article/pii/S0968090X18305278.

[26] D.-F. Li and S.-P. Wan. Fuzzy linear programming approach to multiattribute decision making with multiple types of attribute values and incomplete weight information. *Applied Soft Computing*, 13(11):4333–4348, 2013.

[27] S.-H. Liao, B.-L. Sun, and R.-Y. Wang. A knowledge-based architecture for planning military intelligence, surveillance, and reconnaissance. *Space Policy*, 19(3):191–202, 2003.

[28] B. Lokman, M. Köksalan, P. J. Korhonen, and J. Wallenius. An interactive approximation algorithm for multi-objective integer programs. *Computers & Operations Research*, 96:80–90, 2018. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2018.04.005. URL https://www.sciencedirect.com/science/article/pii/S0305054818300881.

[29] Y. Z. Mehrjerdi. Strategic system selection with linguistic preferences and grey information using mcdm. *Applied Soft Computing*, 18:323–337, 2014.

[30] R. H. Myers, D. C. Montgomery, G. G. Vining, C. M. Borror, and S. M. Kowalski. Response surface methodology: a retrospective and literature survey. *Journal of quality technology*, 36(1):53–77, 2004.

[31] L. Paquete, B. Schulze, M. Stiglmayr, and A. C. Lourenço. Computing representations using hypervolume scalarizations. *Computers & Operations Research*, 137:105349, 2022. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2021.105349. URL https://www.sciencedirect.com/science/article/pii/S0305054821001283.

[32] J. Pereira, M. Ritt, and Ó. C. Vásquez. A memetic algorithm for the cost-oriented robotic assembly line balancing problem. *Computers & Operations Research*, 99:249–261, 2018.

[33] M. Shaito and R. Elmasri. Map visualization using spatial and spatio-temporal data: Application to covid-19 data. In *The 14th PErvasive Technologies Related to Assistive Environments Conference*, pages 284–291, 2021.

[34] K. S. Shehadeh and R. Padman. Stochastic optimization approaches for elective surgery scheduling with downstream capacity constraints: Models, challenges, and opportunities. *Computers & Operations Research*, 137:105523, 2022. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2021.105523. URL https://www.sciencedirect.com/science/article/pii/S0305054821002628.

[35] M. O. Sidi, A. Kadrani, B. Quilot-Turion, F. Lescourret, and M. Génard. Compromising nsga-ii performances and stopping criteria: case of virtual peach design. In *International Conference on Metamaterials, Photonic Crystals and Plasmonics*, pages 2–p, 2012.

[36] T. J. Stewart and R. Janssen. A multiobjective gis-based land use planning algorithm. *Computers, environment and urban systems*, 46: 25–34, 2014.

[37] T. J. Stewart, R. Janssen, and M. van Herwijnen. A genetic algorithm approach to multiobjective land use planning. *Computers & Operations Research*, 31(14):2293–2313, 2004. ISSN 0305-0548. doi: https://doi.org/10.1016/S0305-0548(03)00188-6. URL https://www.sciencedirect.com/science/article/pii/S0305054803001886.

[38] E.-G. Talbi. *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons, 2009.

[39] C.-W. Tsai, L.-I. Tong, C.-H. Wang, et al. Optimization of multiple responses using data envelopment analysis and response surface methodology. *Journal of Applied Science and Engineering*, 13(2):197–203, 2010.

[40] O. Veblen. The heine-borel theorem. *Bulletin of the American Mathematical Society*, 10(9):436–439, 1904.

[41] J.-W. Wang, C.-H. Cheng, and K.-C. Huang. Fuzzy hierarchical topsis for supplier selection. *Applied Soft Computing*, 9(1):377–386, 2009.

[42] L. Wenwen, F. Goodchild, and R. Church. An efficient measure of compactness for 2d shapes and its application in regionalization problems. *International Journal of Geographical Info Science*, pages 1–24, 2013.

[43] F. Yang, R. Wu, T. Jin, Y. Long, P. Zhao, Q. Yu, L. Wang, J. Wang, H. Zhao, and Y. Guo. Efficiency of unlocking or locking existing protected areas for identifying complementary areas for biodiversity conservation. *Science of The Total Environment*, 694:133771, 2019.

[44] J. Yao, A. T. Murray, J. Wang, and X. Zhang. Evaluation and development of sustainable urban land use plans through spatial optimization. *Transactions in GIS*, 23(4):705–725, 2019.

[45] S. H. Zanakis, A. Solomon, N. Wishart, and S. Dublish. Multi-attribute decision making: A simulation comparison of select methods. *European journal of operational research*, 107(3):507–529, 1998.

[46] Q. Zhou, U. Benlic, and Q. Wu. An opposition-based memetic algorithm for the maximum quasi-clique problem. *European Journal of Operational Research*, 286(1):63–83, 2020.