Received: September 06, 2022; **Accepted:** March 18, 2023.
DOI. 10.30473/coam.2023.65380.1213
Winter-Spring (2024) Vol. 9, No. 1, (131-147)
Research Article

Open Access

Control and Optimization in Applied Mathematics - COAM

An Efficient Variable Neighborhood Search for Solving Multi-Criteria Project Portfolio Selection

Ali Akbar Sohrabi[1], **Reza Ghanbari**[1], **Khatere Ghorbani Moghadam**[2]

[1]Faculty of Mathematical Sciences, Department of Applied Mathematics, Ferdowsi University of Mashhad, Mashhad, Iran.
[2]Mosaheb Institute of Mathematics, Kharazmi University, Tehran, Iran.

Correspondence:
Khatere Ghorbani Moghadam
E-mail:
k.ghorbani@khu.ac.ir

How to Cite
Sohrabi, A.A., Ghanbari, R., Ghorbani Moghadam Kh. (2024). "An efficient variable neighborhood search for solving multi-criteria project portfolio selection", Control and Optimization in Applied Mathematics, 9(1): 131-147.

Abstract. Project portfolio selection is a critical challenge for many organizations as they often face budget constraints that limit their ability to support all available projects. To address this issue, organizations seek to select a feasible subset of projects that maximizes utility. While several models for project portfolio selection based on multiple criteria have been proposed, they are typically NP-hard problems. In this study, we propose an efficient Variable Neighborhood Search (VNS) algorithm to solve these problems. Our algorithm includes a formula for computing the difference value of the objective function, which enhances its accuracy and ensures that selected projects meet desired criteria. We demonstrate the effectiveness of our algorithm through rigorous testing and comparison with a genetic algorithm (GA) and CPLEX. The results of the Wilcoxon non-parametric test confirm that our algorithm outperforms both GA and CPLEX in terms of speed and accuracy. Moreover, the variance of the relative error of our algorithm is less than that of GA.

Keywords. Project portfolio selection, Project interaction, Multi-criteria, Meta-heuristic algorithms.

MSC. 90C34; 90C40.

https://mathco.journals.pnu.ac.ir

1 Introduction

Project portfolio selection problem (PPSP) plays an important role in some fields such as various fields, including project management, risk management, and financial management (see [17, 19]). As a pivotal decision in many organizations and fields, the PPSP is challenging as it involves determining which projects should be funded, and making the wrong decision can have dire consequences [7].

According to [5], the PPSP involves selecting project of a portfolio from accessible project plans and ongoing projects while ensuring that the portfolio meets the firm's resource constraints. In other words, the PPSP can be defined as follow: under the limited resources, the decision-makers (DMs) select the best subset of projects from the pool the candidate projects. In general, the literature on project portfolio selection can be categorized into two major categories [34]. The first category aims to solve the project selection problems using various methods and tools, many of which are based on programming methods, such as linear, nonlinear, integer, dynamic, goal, fuzzy, and stochastic programming. Other mathematical models include the decision-tree approaches, game-theoretical approaches, simulation models, and heuristic methods. The second category focuses on the application of project selection in real-world areas such as research and development, information technology, public administration, and capital budgeting. Selecting the project portfolio and allocating resources are the main problems faced by DMs. Due to the importance of the PPSP, it has received significant attention from researchers (see [2, 6, 18, 20, 21, 24, 28, 32, 33, 36]).

The PPSP is a crucial problem for organizations, and over the subsequent decades, several researchers have contributed to designing and implementing various approaches to solve it. Aker et al. [1] suggested using mixed-integer quadratic programming (MIQP) to select $R\&D$ project portfolio. Golabi et al. [14] proposed a multi-criteria model for the PPSP, which was used by the department of energy to select a portfolio for solar energy projects. Song et al. [30] proposed an approach based on stochastic multi-criteria acceptability analysis (SMAA) to effectively manage multi-criteria project portfolio selection and scheduling problems. This SMAA-based approach has two main procedures: first, a backtracking algorithm is used to find all satisfactory portfolios, and second, the SMAA is conducted to evaluate satisfactory portfolios and help decision-makers select the optimal portfolio. Cooper et al. [9] conducted studies studies on some companies, and they presented reasons to illustrate the importance of the portfolio. Kyparisis et al. [29] proposed a comprehensive nonlinear model for the PPSP, considering the interaction between projects, which was used in large service organizations. Ghasemzadeh et al. [13] presented a MIQP model for the PPSP, taking the schedule into account. Stummer and Heidenbeger [31] compared the income of portfolios by considering the interaction between projects, and he indicated that neglecting mutual relations among projects can lead to wrong decisions. Medaglia et al. [26] presented a multi-objective model under uncertain condition, while Almeida and Duarte [4] presented a nonlinear model for the PPSP, taking into account the synergy between projects. Numerous researchers have considered uncertain criteria values and preference information in project portfolio selection, proposing useful methods. For uncertain criteria values, the most common methods used are fuzzy set and stochastic theories [23].

From a computational complexity point of view, numerous metaheuristics have been proposed for the PPSP. Crama et al. [10] suggested using simulated annealing to solve their model. Dorner et al. [11] formulated a multi-objective model for the PPSP and implemented a partial swarm algorithm to solve it. Carazo et al. [8] used scatter search to solve the multi-objective model of the PPSP. Yu et al. [35] formulated nonlinear programming for the PPSP, and used GA to solve it. They compared their result with CPLEX. Upon reviewing the literature on the PPSP, several applications can be observed. For instance, Ghasemzadeh et al. [13] utilized the PPSP for telecommunication manufacturing firms. In 2008, Medaglia et al. [25] employed portfolio selection in the public enterprise water management.

In this paper, we present a VNS algorithm for solving the PPSP, as the VNS has been successfully applied to a broad range of different NP-hard problems and has a simpler structure compared with other metaheuristics. To the best of our knowledge, no VNS algorithm has been employed to solve the PPSP. To demonstrate the efficiency of the proposed algorithm, we evaluate its performance on a set of randomly generated instances and compare its results with those of the GA presented in [35] and CPLEX as a well-known commercial solver.

The rest of the paper is organized as follows: In Section 2, we introduce the PPSP model suggested by Yu et al. [35]. In Section 3, we present the VNS algorithm is proposed for solving the PPSP. In Section 4, we give the numerical result of the VNS algorithm and compare its performance with the GA and CPLEX. Finally, we conclude the paper in Section 5.

2 Multi-Criteria Formulation of the PPSP

While there have been many papers that investigate project portfolio selection in detail, some of them consider the interaction between projects. In the real world, the interaction among projects can be observed from various perspectives, such as cost and resources, which greatly affect project selection in the portfolio. The Interaction among projects creates a the difference between selecting independent projects and selecting portfolio projects. In this section, we describe the optimization model proposed by Yu et al. [35], which our algorithm is based on.

Suppose there are I projects that are evaluated and selected, and J is the number of criteria DMs. Let x_i denote the be a decision variable corresponding to i-th project $(i = 1, 2, \ldots, I)$, where $x_i = 1$ if the i-th project is selected in the portfolio, and $x_i = 0$ otherwise. Therefore, a project portfolio can be represented by a vector $X = (x_1, x_2, \ldots, x_I)$. Each criterion has a preference weight for DMs, denoted by w_j $(j = 1, 2, \ldots, J)$. Suppose c_{ij} is the value of project i $(i = 1, 2, \ldots, I)$ based on the j-th criterion $(j = 1, 2, \ldots, J)$, and also $d_j(s_k)$ is the value of interaction effect in a combination of K projects based on j-th criterion. The following model is built to select M preferred from I candidate projects while consideration interaction among projects [35].

$$\begin{cases} \max & V = \sum_{i=1}^{I}(\sum_{j=1}^{J} w_j c_{ij})x_i + \sum_{j=1}^{J}\sum_{k=1}^{K}(w_j(d_j(S_k))(\sum_{i=1}^{L} c_{ij}))\prod_{i=1}^{L} x_i, \\ \text{s.t.} & \\ & \sum_{i=1}^{I} x_i = M, \quad x_i = \{0,1\}, \end{cases} \tag{1}$$

where, L represents the number of interactive effects projects, while V denotes the total utility value obtained from selecting projects for the portfolio.

Example of model

In this study, we provide an example for the model (1) presented by Yu et al. [35]. We consider a scenario where decision-makers are tasked with selecting a portfolio of projects, choosing from a set of five candidates (a_1, a_2, a_3, a_4 and a_5). The decision-makers must evaluate each project based on three criteria: risk tolerance, capital return, and project feasibility. Their aim is to select two projects from the set of five that will maximize the value of the portfolio. Table 1 presents the input data for each of the five projects.

Table 1: Basic data for five different projects.

j	w_j	a_1	a_2	a_3	a_4	a_5
Reversibility	3	0.33	0.27	0.56	0.44	0.78
Risk	1	0.65	0.48	0.89	0.51	0.42
Feasibility	4	0.5	0.75	0.48	0.68	0.70

where the first column lists the evaluation criteria, while the second column shows the weight assigned to each criterion. The third, fourth, fifth, sixth, and seventh columns display the evaluations for each project. For instance, consider project a1, which has been evaluated based on three criteria: reversibility, risk ability, and project feasibility. Its respective scores for these criteria are 0.42, 0.78, and 0.70, respectively.

Usually, the weight of criteria is normalized in the range $[0,1]$. Here, the normalization is done through the following formula:

$$w'_j = \frac{w_j}{\sum_{j=1}^{J} w_j}.$$

After normalizing the weights, Table 1 is converted into Table 2.

Table 2: Data normalization for five different projects.

j	w_j	a_1	a_2	a_3	a_4	a_5
Reversibility	0.375	0.33	0.27	0.56	0.44	0.78
Risk	0.125	0.65	0.48	0.89	0.51	0.42
Feasibility	0.500	0.5	0.75	0.48	0.68	0.70

In the previous discussions, the mutual effects that different projects may have on each other were taken into account. However, in many practical problems related to choosing the optimal portfolio, project interactions play a significant role. For instance, as shown in Table 3, if projects a_1 and a_5 are selected together, the possibility of capital reversibility increases while the risk decreases. To better understand the mutual effects of the projects, refer to Table 3. Consider the third criterion, feasibility, and the pairs of projects $S_1 = a_1, a_4$, $S_2 = a_2, a_4$, $S_3 = a_4, a_5$. These pairs exhibit interactions with values $d_3(S_1) = 0.35$, $d_3(S_2) = -0.42$ and $d_3(S_3) = -0.8$, respectively.

Table 3: Interaction between projects.

j	a_{12}	a_{13}	a_{14}	a_{15}	a_{23}	a_{24}	a_{25}	a_{34}	a_{35}	a_{45}
Reversibility	0.2	0.45	0.4	0.65	0.45	0.35	0.55	0.4	0.6	0.55
Risk	-0.15	-0.25	-0.14	-0.15	-0.20	-0.05	-0.10	-0.15	-0.25	-0.15
Feasibility	0	0	0.35	0	0	-0.42	0	0	0	-0.8

Therefore, we incorporate this factor into the model and utilize the data presented in Tables 2 and 3, The results are shown as:

$$\begin{cases} \max \quad V = 0.455x_1 + 0.53625x_2 + 0.56125x_3 + 0.56875x_4 + 0.695x_5 + 0.0238x_6 \\ \qquad +0.1021x_7 + 0.3035x_8 + 0.2505x_9 + 0.1058x_{10} - 0.5136x_{11} + 0.2053x_{12} \\ \qquad +0.1238x_{13} + 0.2606x_{14} - 0.2972x_{15}, \\ \text{s.t.} \\ \qquad \sum_{i=1}^{5} x_i = 2, \quad x_i = \{0,1\}. \end{cases} \tag{2}$$

Notably, the model becomes linear when project interactions are considered. . After solving the problem with the CPLEX optimizer the optimal solution is By solving this problem using the CPLEX optimizer, we obtain an optimal solution of $x = (0,0,1,0,1)$ and the objective function value of $V = 1.51685$.

The model (1) is a nonlinear integer programming (NLIP) problem which is known to be NP-hard [35]. To address this issue, Yu et al. proposed a GA for solving (1). In this paper, we introduce the VNS algorithm, as discussed in Section 3, as an alternative approach to solve the model.

3 Variable Neighborhood Search Algorithm

The VNS is a metaheuristic algorithm proposed by Hansen and Mladenovic [16, 27]. It is one of the most popular algorithms used by researchers to solve optimization problems. As described in [16] the algorithm systematically changes neighborhoods to find local optima and escape valleys. Compared to other metaheuristic algorithms, VNS has the advantage of requiring fewer parameters and providing rational solutions.Consequently, the VNS algorithm is widely used in various fields including location theory, cluster analysis, vehicle routing, network design, and biology [12]. To introduce the VNS algorithm, we should first define some of the following concepts:

3.1 Feasible space

In the PPSP, each solution is represented as $X = (x_1, x_2, \ldots, x_I)$. The feasible solution set can be defined as follow:

$$\Gamma = \{(x_1, x_2, \ldots, x_I)|x_i \in \{0,1\} \ i = 1,2,\ldots I, \sum_{i=1}^{I} x_i = M\}. \tag{3}$$

3.2 Initial solution

In the PPSP, projects may have interactions with each can be positive or negative. Let $d_k(s_{ij})$ represent the effect of the project i on project j based on k-th criterion. Furthermore, suppose Z is a matrix that shows the interactions among projects, defined as follows:

$$Z = \begin{bmatrix} z_{11} & z_{12}\cdots & z_{1I} \\ z_{21} & z_{22}\cdots & z_{2I} \\ \vdots & \ddots & \vdots \\ z_{I1} & z_{I2}\cdots & z_{II} \end{bmatrix}, \tag{4}$$

In this matrix, z_{ij} $(i,j = 1,2,\ldots,I)$ represents the effect of project i project on project j based on all of the criteria calculated as follows:

$$z_{ij} = \sum_{k=1}^{J} d_k(s_{ij}) w_k(c_{ik} + c_{jk}) \quad i,j = 1,2,\ldots,I \ \ (i \neq j). \tag{5}$$

Note that the elements on the diagonal of Z are zero, as projects do not affect themselves (i.e., $z_{ii} = 0, \ (i = 1,2,\ldots,I)$).

Definition 1. During the process of selecting projects for the portfolio, may some projects are selected and also some projects may be chosen while others are not. Let $X = (x_1, x_2, \ldots, x_I)$ be a feasible solution for the PPSP. We define two sets M_X and $(I-M)_X$ as follows.

$$M_X = \{x_i \in X | x_i = 1\},$$
$$(I-M)_X = \{x_i \in X | x_i = 0\}.$$

To obtain the initial solution, we use a greedy method which creates the solution as follows.
Let $\beta = (\beta_1, \beta_2, \ldots, \beta_I)$ be a vector, where

$$\beta_i = \sum_{j=1}^{J} c_{ij} w_j + \sum_{j=1}^{I} z_{ij}, \quad i = 1, 2, \ldots, I, \tag{6}$$

and β_i is the value of i-th project.

The β_i is defined and used to create an initial solution using the greedy algorithm, which is described in detail below.

To find a feasible greedy solution, we first calculate the β_i value for all projects $(i = 1, 2, \ldots, I)$ and sort them in descending order from highest to smallest, as the problem is a maximization problem. Initially, we assume that none of the projects are selected, i.e., $\mid (I - M)_X \mid = I$. We start by selecting the project with the maximum value of β_i, and update the sets of $M_X \leftarrow M_X \cup \{x_i\}$ and $(I - M)_X \leftarrow (I - M)_X - \{x_i\}$. If decision-makers plan to select more projects, we choose the second project with the highest value of β_i and update the sets of M_X and $(I - M)_X$. This iterative procedure continues until the constraint is not violated. Since we order the values of the β_i, the computational complexity of finding an initial solution is $O(ILogI)$.

To illustrate this point more clearly, let us consider a scenario where we have five projects and aim to select two projects out of the five. Let the values β_i be 0.2, 0.5, 0.1, 0.75 and 0.35 respectively. To obtain the initial solution, the β_i values are sorted in descending order. Accordingly, we first select the 4-th project, which has the highest β_i value, and add it to the project portfolio. As the limit of the number of projects has not been violated, we select the second project that with the highest β_i value (which is the second project in this case) and adds it to the project portfolio. Since the number of selected projects must be 2, if we choose another project, the limit will be violated, and we cannot add another project to the project portfolio. Therefore, we include projects 2 and 4 in the project portfolio.

3.3 Neighborhood

Suppose $X = (x_1, x_2, \ldots, x_I)$ is a feasible solution for the PPSP. The k-th neighborhood of X is denoted by $N_k(X)$, and it defined as follow:

$$N_k(X) = \{Y \in \Gamma | \ \ |Y - X| = k\}, \tag{7}$$

or, equivalently,

$$N_k(X) = \{Y \in \Gamma \mid \text{The solutions } X \text{ and } Y \text{ differ in } k \text{ projects}\}. \tag{8}$$

For example, if $X = (1, 0, 0, 1, 0)$, $Y = (0, 1, 0, 1, 0)$ and $Z = (0, 1, 1, 0, 0)$ are three feasible solutions, then $Y \in N_1(X)$ and $Z \in N_2(X)$.

3.4 Local search

Local Search (LS) is a general method for solving hard optimization problems. By making minor adjustments, the LS algorithm aims to enhance existing solutions. It navigates through the search space to move from one solution to a superior one. To discover a new solution, LS utilizes two general algorithms known as the first and best improvement algorithms [12]. In Algorithm 5, we utilize the first improvement method to obtain the new solution.

Definition 2. Let X and X' be two feasible solutions for the PPSP. The function $\varphi(X, X')$ is defined as follows:

$$\varphi(X, X') = \{k | x_k \neq x'_k, x_k \in M_X, x'_k \in M_{X'}\}.$$

Therefore, the distance between two solutions X and X' is denoted by $d(X, X')$ and calculated as $d(X, X') = |\varphi(X, X')|$. For example, if $X_1 = (1, 0, 1, 0)$ and $X_2 = (0, 1, 1, 0)$ then $d(X_1, X_2) = 1$.

One of the main features of VNS is the definition of neighborhoods. Solutions, that are located in each other's neighborhoods, have certain similarities. This efficient feature accelerates the process of comparing the objective functions values [3]. To utilize this feature for the PPSP, we express the following theorem.

Theorem 1. Suppose X and Y are two feasible solutions for the PPSP, where $Y \in N_K(X)$. Furthermore, if $x_{\alpha_1}, x_{\alpha_2}, \ldots, x_{\alpha_k} \in M_X - M_Y$ and $x_{\alpha'_1}, x_{\alpha'_2}, \ldots, x_{\alpha'_k} \in M_Y - M_X$, then, the change in the objective function value is calculated as follows:

$$\begin{aligned} \triangle F(X,Y) &= F(X) - F(Y) \\ &= \sum_{k=1}^{K}(\sum_{j=1}^{J} c_{kj}w_j)x_{\alpha_k} + \sum_{i=1}^{K}\sum_{j=1}^{I} z_{ij}x_{\alpha_i}x_j \\ &\quad - \sum_{k=1}^{K}(\sum_{j=1}^{J} c_{kj}w_j)x_{\alpha'_k} - \sum_{i=1}^{K}\sum_{j=1}^{I} z_{ij}x_{\alpha'_i}x_j. \end{aligned} \tag{9}$$

Proof. Let K=1. Suppose that the portfolios X and Y differ in two projects f and g (i.e., $x_f \in M_X - M_Y$ and $x_g \in M_Y - M_X$), then,

$$\begin{aligned} \triangle F(X,Y) = &\sum_{\substack{i=1 \\ i\neq f}}^{I-1}(\sum_{j=1}^{J} c_{ij}w_j)x_i + \sum_{j=1}^{J} c_{fj}w_jx_f + \sum_{\substack{i=1 \\ i\neq f}}^{I-1}\sum_{j=1}^{I} z_{ij}x_ix_j + \sum_{j=1}^{I} z_{fj}x_fx_j \\ &- \sum_{\substack{i=1 \\ i\neq g}}^{I-1}(\sum_{j=1}^{J} c_{ij}w_j)x_i - \sum_{j=1}^{J} c_{gj}w_jx_g - \sum_{\substack{i=1 \\ i\neq g}}^{I-1}\sum_{j=1}^{I} z_{ij}x_ix_j - \sum_{j=1}^{I} z_{gj}x_gx_j, \end{aligned}$$

then,

$$\triangle F(X,Y) = \sum_{j=1}^{J} c_{fj}w_j + \sum_{j=1}^{I} z_{fj}x_fx_j - \sum_{j=1}^{J} c_{gj}w_jx_g - \sum_{j=1}^{I} z_{gj}x_gx_j. \tag{10}$$

If we extend this idea to $K > 1$, we can easily obtain formulation (9). □

The first improvement algorithm can be used for local searches. When we need to search the entire neighborhood $N(X)$, we use the first improvement algorithm. This algorithm returns the best value of the objective function (minimum or maximum) after searching the entire neighborhood. When a decreasing direction (or increasing direction for maximization problems) is found, the move is made.

The advantage of Theorem 1 is that it eliminates the need to compute objective functions from scratch when comparing two solutions that differ in only a few components, especially when the size of projects in the portfolio is large enough. In such cases, it is only necessary to calculate the objective values for the components that differ between the two solutions. Therefore, we incorporate Theorem 1 into Algorithm 5 to compare two points.

3.5 Stopping Criterion

The lack of an effective stopping criterion is one of the main disadvantages many of metaheuristics. However, in many algorithm implementations, the stopping criteria usually include a maximum number of iterations or a maximum number of consecutive iterations without any improvement in the best

Algorithm 5 First Improvement(X)

$X' \leftarrow X i \leftarrow 1$ $j \leftarrow 1$ **Do** $x_i \in M_{X'}$ **Do** $x_j \in (I-M)_{X'}$ $M_{X'} \leftarrow M_{X'} \cup \{x_j\} - \{x_i\}$ $(I-M)_{X'} \leftarrow (I-M)_{X'} \cup \{x_i\} - \{x_j\}$ **If** $(\triangle F(X', X) > 0)$ (See (9)) **Return** X' break **else** $X' \leftarrow X$ $j \leftarrow j+1$ **Until** $(j = |(I-M)_{X'}|)$ $i \leftarrow i+1$ **Until** $(\triangle F(X', X) > 0 \ or \ i = |M_{X'}|)$ **If** $\triangle F(X', X) > 0$ **Return** X' **else** **Return** X

solution value and so on. In this paper, the stopping criterion is determined as the maximum number of iterations without any improvements of objective functions taken over two consecutive improvements. This means that the algorithm is stopped if the objective function does not improved after certain number iterations.

We now summarize our proposed algorithm in Algorithm 2.

Algorithm 6 Algorithm VNS For the PPSP(X, K_{max})

Let $MaxImprovement$ is the stopping condition (See **??**) Let $L \leftarrow 1$ $X \leftarrow$ Initial Solution(X) (See 3.2) **while** $L \leq MaxImprovemnt$ **do** $k \leftarrow 1$ **while** ($k \leq K_{max}$) **do** Select $X' \in N_k(X)$ randomly $X'' \leftarrow$ First Improvement(X') (See 3.4) **If** $(\triangle F(X'', X) > 0)$ $X \leftarrow X''$ $k \leftarrow 1$ $L \leftarrow 1$ **else** $k \leftarrow k+1$ **End while** $L \leftarrow L+1$ **End While** return X

3.6 Genetic algorithm

The genetic algorithm is an evolutionary algorithm that operates based on the principles of nature. In nature, individuals that can adapt to changing environment survive, while those who cannot adapt perish. The characteristics of individuals are encoded in genes that are stored in chromosomes. Individuals with better adaptability to the environment have a higher chance of survival than those with less adaptability. In GA, new individuals are produced through crossover and genetic mutation. Each solution in a population is assigned a fitness value, which is used to calculate the quality of the solution. The convergence time and direction of evolution greatly depend on selection strategies employed. The genetic algorithm follows a series of selection, crossover, and mutation operations. Initially, individuals with the best fitness value are selected. During the crossover, genes are recombined to create a new generation, with the aim of inheriting the desirable features of the parents. Finally, mutation creates a new individual by altering a gene.

When using GA to solve various problems, each chromosome represents a complete solution. The fitness value is used to evaluate a solution, with a lower fitness value indicating a better solution. Genetic algorithms employ crossover, mutation, and selection operations to find an optimized solution. The selection operation chooses the best chromosomes and discards others chromosomes based on their fitness value. After a few generations, the population consists of individuals with the best fitness values, which represents an optimal solution.

Due to the NP-hardness of the problem (1), no polynomial time algorithm has been found to solve it. The flowchart for GA is depicted in Figure 1.

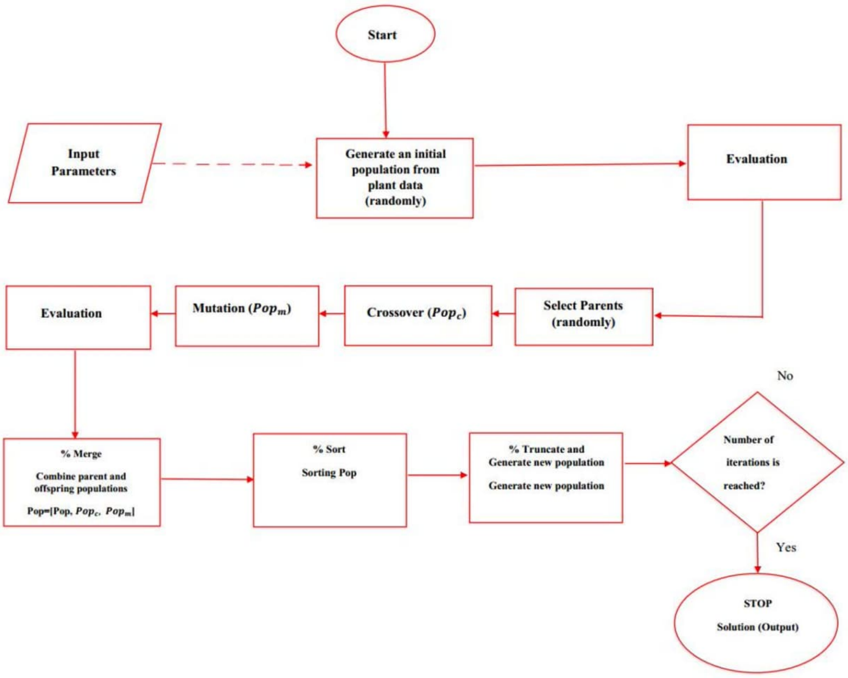

Figure 1: Flowchart of GA.

3.6.1 Encoding

Various encoding strategies are used for different problems, including symbol encoding, binary number encoding, and real number encoding. In this study, we employ binary encoding, where the solution space is mapped to a set of strings consisting of zeros and ones. For our proposed problem, the value of 1 indicates that the i-th vertex is selected for graph partitioning, while 0 indicates that it is not. The length of the chromosome is equal to the number of vertices in the graph. For example, if there are 7 vertices and 3 of them are selected for partitioning then Figure 2 represents a simple chromosome. Each column in the chromosome represents a gene, and the set of chromosomes is called a population.

0	1	1	0	0	1	0

Figure 2: A chromosome.

3.6.2 Fitness

In a genetic algorithm, crossover and mutation operations are performed in each iteration, generating new solutions. It is necessary to ensure that the solutions are feasible. The fitness value is used to evaluate the quality and feasibility of the solution. In our proposed algorithm, the fitness value is evaluated by the objective function of problem (1). The lower the fitness value indicates a better solution.

3.6.3 Selection

During selection, solutions with the best fitness are chosen, while others are discarded. In our proposed algorithm, we utilize a roulette wheel selection scheme [15].

3.6.4 Crossover

Crossover is employed to generate new offspring from individuals selected during the selection phase. Various crossover techniques have been proposed in the literature, but we use a two-point crossover. In this technique, the chromosomes are first divided into two sections using two randomly selected cut points. The positions of the cut points can be randomly selected as:

$$[C_1, C_2] = \text{rand}(2) \times (TK - 1) + 1,$$

where, C_1 and C_2 are two cut points and $\text{rand}(2) \times (TK - 1) + 1$ generate a random number between 1 and to the total length of the chromosome.

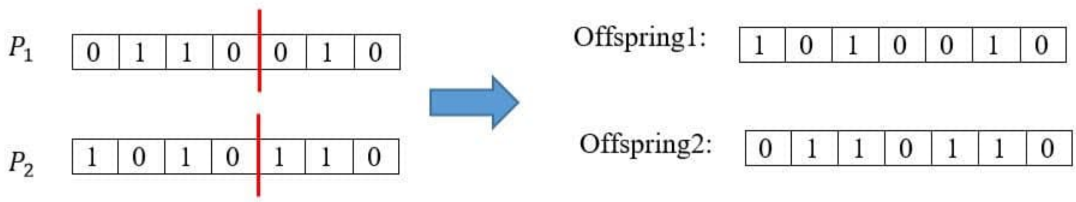

Figure 3: Crossover.

3.6.5 Mutation

The mutation operation introduces new variations in chromosomes. Various mutation techniques have been proposed, including swap mutation, uniform mutation, and interchange mutation. In these techniques, a position in the solution is randomly selected, and its gene value is replaced with the gene value of another randomly selected position. The main steps of our proposed algorithm can be summarized using the following pseudocode, which represents Algorithm 7:

4 Numerical Results

In this section, we evaluate the numerical results of our proposed algorithm. Firstly, we compare the result of our algorithm with the GA using an example presented in [35]. Secondly, we generate some random samples and compare the numerical result of our proposed algorithm with GA and CPLEX. All computational experiments were conducted conducted using Visual Studio 2010 and the programs were executed on a notebook with a core i5 CPU running at 2.50 GHz and with 6.00 GB of RAM.

Yu et al. [35] considered an example with 5 projects, where interaction among projects were assumed. The numerical model used in their study is as follows:

Algorithm 7 Genetic algorithm

Input: $maxiter$, Initial population, $maxiter$.

1: $t \leftarrow 0$.

2: While $t <= maxiter$ do:

2-1 : Evaluate the objective function for the initial population.

2-2 : Select parents randomly ($P'(t)$).

2-3 : $Pop_c \leftarrow Crossover(P'(t))$.

2-4 : $Pop_m \leftarrow Mutation(Pop_c)$.

2-5 $t \leftarrow t + 1$.

Output: return X.

$$\begin{cases} \max \quad V = 0.455x_1 + 0.53625x_2 + 0.56125x_3 + 0.56875x_4 + 0.695x_5 \\ \qquad +0.0238x_1x_2 + 0.1021x_1x_3 + 0.3035x_1x_4 + 0.2505x_1x_5 \\ \qquad +0.1058x_2x_3 - 0.5136x_2x_4 + 0.2053x_2x_5 \\ \qquad +0.1238x_3x_4 + 0.2606x_3x_4 - 0.2972x_4x_5 \\ \text{s.t.} \\ \qquad \sum_{i=1}^{5} x_i = 2, \quad x_i = \{0, 1\}. \end{cases} \tag{11}$$

After solving this problem using GA and VNS, the objective function value was 1.51685 in zero seconds. CPLEX can obtain a similar objective function value in 0.22 seconds.

4.1 Generating random samples

To evaluate the efficiency of our proposed algorithm compare to GA and CPLEX, we generate random test problems of varying sizes, ranging from small to large. In these instances, the coefficients c_{ij} are generated using random numbers with a uniform distribution between 0 and 1. Each criterion has a preference weight, w_j, which is randomly generated by first selecting values from the interval $[0, 5]$, and then normalizing these coefficients. Finally, the interaction among projects, $d_j(s_k)$ is built by generating random numbers in the interval $[-1, 1]$, as interactions may be positive or negative. We consider different criteria and select disparate projects for each set of projects in this study.

The notations used in the tables are defined as follows:

REAvg represents the average relative error, which is calculated as follows:

$$\textbf{REAvg} = \frac{\text{exact solution} - \text{approximate solution}}{\text{exact solution}}.$$

Time shows the required execution time for solving the test problems. To solve the PPSP using the VNS algorithm, we need to determine some parameters, such as the number of neighborhoods and also stopping condition.

To estimate the best value of the algorithm's parameters, we run the VNS algorithm with different parameter values and select the best one. The test problems are generated in three different sets: small, medium, and large. The numerical results are presented in three separate tables (Tables 5, 6 and 7). The columns labeled I, J, and M represent the number of candidate projects, criteria, and selected projects, respectively.

4.2 Setting parameters

We used the Irace package [22] to set the parameters of the GA to ensure a fair comparison. Based on one randomly chosen instance from each subset, Table 4 shows the parameters of the GA, their corresponding types and intervals, as well as the best values returned by Irace for all instances.

Table 4: Parameters of the GA.

Parameter	Description	Type	Interval	IRACE
MaxIt	Maximum number of iteration	integer	$(50, 100, 200, 300)$	300
nPop	Number of population	integer	$(20, 100, 150, 200)$	150
pc	Crossover rate	continues	$(0.1, 0.8)$	0.76
pm	Mutation rate	continues	$(0.05, 0.5)$	0.3

4.3 Small dataset

Firstly, we examine the performance of the algorithms on small sets of projects. In this category, we create instances with 5 to 50 projects using random numbers (See Subsection 3.1). For example, triplex $(40, 6, 6)$ represents an instance with 40 candidate projects and 6 criteria, where the goal is to select 6 projects from the candidate pool, resulting in a portfolio with maximum utility. The optimal solution, obtained using CPLEX, is 8.7798 in 0.67 seconds. The VNS and the GA produce similar results at zero seconds.

From the information presented in Table 5, it is apparent that the VNS and the GA almost obtain similar results to CPLEX. However, for larger test problems, these algorithms may approach the exact solutions but with significantly reduced computation times.

Table 5: Computational results for small set.

I	J	M	Optimal			Time(s)			REAvg(%)	
			CPLEX	GA	VNS	CPLEX	GA	VNS	GA	VSN
5	2	1	0.8194	0.8194	0.8194	0.58	10^{-5}	10^{-8}	0.0000	0.0000
	3	2	1.72503	1.72503	1.72503	0.33	2×10^{-5}	3×10^{-5}	0.0000	0.0000
	4	3	2.2193	2.2193	2.2193	0.58	5×10^{-5}	6×10^{-5}	0.0000	0.0000
10	3	2	1.3903	1.3903	1.3903	0.44	5×10^{-5}	8×10^{-6}	0.0000	0.0000
	4	3	2.7915	2.7915	2.7915	0.5	4×10^{-6}	2×10^{-8}	0.0000	0.0000
	5	4	5.4043	5.4043	5.4043	0.25	3×10^{-8}	6×10^{-7}	0.0000	0.0000
20	4	2	2.2564	2.2564	2.2564	0.42	7×10^{-4}	8×10^{-8}	0.0000	0.0000
	6	6	4.4792	4.4792	4.4792	0.22	3×10^{-6}	3×10^{-7}	0.0000	0.0000
	8	6	6.3401	6.3401	6.3401	0.2	3×10^{-10}	6×10^{-7}	0.0000	0.0000
30	4	3	3.6297	3.6297	3.4297	0.45	1	5×10^{-6}	0.0000	0.0000
	6	4	4.7559	4.7559	4.7559	0.5	1	4×10^{-6}	0.0000	0.0000
	8	5	5.9165	5.9165	5.9165	**0.45**	1	8×10^{-10}	0.0000	0.0000
40	4	5	9.4578	9.4578	9.4578	0.27	2	3×10^{-7}	0.0000	0.0000
	6	6	8.7798	8.7798	8.7798	0.67	2	2×10^{-9}	0.0000	0.0000
	8	7	9.3986	9.3986	9.3986	1.22	6×10^{-11}	6×10^{-8}	0.0000	0.0000
50	4	6	11.2575	11.2575	11.2575	0.75	1	8×10^{-8}	0.0000	0.0000
	6	7	11.5023	11.5023	11.5023	0.86	1	8×10^{-7}	0.0000	0.0000
	8	9	12.1334	12.0868	12.1334	10.14	**2**	1	0.3823	0.0000

4.4 Medium dataset

In this category, we generate some test problems with 60, 70, 80, and 90 candidate projects as shown in Table 6.

Regarding the results in Table 6, the required time for CPLEX to solve the test problems is significantly longer than for both the GA and VNS algorithms. Additionally, the VNS algorithm solves these problems in less time than the GA algorithm.

Table 6: Computational results for medium-sized.

I	J	M	Optimal			Time(s)			REAvg(%)	
			CPLEX	GA	VNS	CPLEX	GA	VNS	GA	VSN
60	6	4	5.624	5.642	5.624	5.624	1	8×10^{-7}	0.0000	0.0000
	8	5	7.4782	7.4782	7.4782	1.04	2	5×10^{-7}	0.0000	0.0000
	10	6	7.8959	7.8959	7.8959	5.38	8	4	0.0000	0.0000
70	10	6	8.1488	8.1488	8.1488	16.89	6	1	0.0000	0.0000
	12	7	9.3521	9.3521	9.3521	32.98	5	2	0.0000	0.0000
	14	8	9.9499	9.9499	9.8991	71.26	11	1	0.0000	0.5105
80	10	5	6.1151	6.1151	6.0357	118.96	11	9	0.0000	1.2984
	12	7	9.1124	8.88	9.1124	202.27	17	2	2.5504	0.0000
	14	9	12.431	12.431	12.431	02.27	17	8	0.0000	0.0000
90	11	6	8.2836	8.2836	8.2836	176.3	21	3	0.0000	0.0000
	13	8	10.7976	10.7976	10.7976	2500	66	24	0.0000	0.0000
	15	10	13.7647	13.7647	13.7647	3200	60	14	0.0000	0.0000

4.5 Large dataset

Finally, we generate some large test problems with 100, 125, and 150 candidate projects. Since CPLEX requires significant computational time, we limit the time to 5000 seconds to solve the test problems. For example, in the triplex (150, 14, 12), CPLEX obtains the best feasible solution of 18.3532 in 5000 seconds, while the GA and VNS algorithms achieve 18.5289 in 49 and 34 seconds respectively. Concerning Table 7, the REAvg value for two proposed algorithms is -0.9573. This suggests that these algorithms can obtain better solutions compared to CPLEX, but VNS algorithm achieves it in less time than GA.

Table 7: Computational results for large set.

I	J	M	approximate solution			Time(s)			REAvg(%)	
			CPLEX	GA	VNS	CPLEX	GA	VNS	GA	VSN
100	10	8	12.3898	12.3898	12.3898	5000	26	3	0.0000	0.0000
	12	10	14.75701	14.462	14.6022	5000	63	5	2.1020	1.001
	14	12	14.0343	14.0343	14.0343	5000	17	3	0.0000	0.0000
125	16	20	32.0054	31.064	32.0054	5000	168	28	2.9414	0.0000
	18	22	32.9444	32.4116	32.8773	5000	6	20	1.6321	0.2036
	20	24	36.8191	36.8197	36.8191	5000	427	19	−0.00162	0.0000
150	14	12	18.3532	18.5286	18.5286	5000	49	30	−0.9573	−0.9573
	16	14	22.1738	20.51	22.1738	5000	29	36	7.5034	0.0000
	18	16	24.2137	24.3089	24.2137	5000	275	9	0.3932	0.0000

Hypothesis test

To demonstrate the efficiency of the proposed algorithm compared to GA, we use Wilcoxon non-parametric test. All statistical computational are implemented in R 3.4.3. Initially, we consider the following hypothesis:

$$\begin{cases} H_0 : \mu_{\text{VNS}} = \mu_{\text{GA}}, \\ H_1 : \mu_{\text{VNS}} \neq \mu_{\text{GA}}, \end{cases}$$

where μ_{VNS} and μ_{GA} represent the average relative errors of VNS and GA respectively. The results of the Wilcoxon test on mean relative errors for a significance 0.05, as shown in Table 8, indicate that VNS and GA are relatively the same regarding the relative error. Therefore, we consider the following hypothesis:

$$\begin{cases} H_0 : \sigma^2_{\text{VNS}} = \sigma^2_{\text{GA}}, \\ H_1 : \sigma^2_{\text{VNS}} < \sigma^2_{\text{GA}}, \end{cases}$$

where σ^2_{VNS} and σ^2_{GA} represent the variance of relative error of VNS and GAs , respectively. From the evidence presented in Table 9, we can conclude that the null hypothesis H_0 is rejected. This means that the variance relative error of the proposed algorithm is less than the GA.

Table 8: Statistical comparison of average relative error between VNS and GA using the Wilcoxon test.

Algorithms	GA vs VNS
P-value	0.239

Table 9: Statistical comparison of variance of relative error between VNS and GA using the Wilcoxon test.

Algorithms	GA vs VNS
P-value	1.225×10^{-15}

Additionally, we consider the following hypothesis regarding the mean execution time:

$$\begin{cases} H_0 : \mu_{\text{VNS}_t} = \mu_{\text{GA}_t} \\ H_1 : \mu_{\text{VNS}_t} \neq \mu_{\text{GA}_t} \end{cases}$$

Here, μ_{VNS_t} and μ_{GA_t} represent the average execution time of VNS and GA respectively. As shown in to Table 10, the alternative hypothesis is accepted. This indicates that, regarding the execution time, the proposed algorithm is significantly better than GA.

Table 10: Statistical comparison of average execution time between VNS and GA using the Wilcoxon test.

Algorithms	GA vs VNS
P-value	2.2×10^{-16}

5 Conclusion

In this paper, we have proposed an efficient VNS algorithm for solving the PPSP. To implement the VNS algorithm effectively, we expressed a theorem that modifies the objective function. Additionally, to demonstrate the productivity of the proposed algorithm, we randomly generated different datasets and compared our numerical results with those obtained using the GA and CPLEX. Finally, we utilized Wilcoxon non-parametric test to compare the performance of VNS and GA. The test results illustrated that the proposed algorithm is significantly more effective than GA in solving the PPSP.

Declarations

Availability of supporting data

All data generated or analyzed during this study are included in this published paper.

Competing interests

The authors declare no competing interests are relevant to the content of this paper.

Authors' contributions

The main manuscript text is collectively written by all authors.

Acknowledgements

The first and second authors would like to express their gratitude to the Research Council of Ferdowsi University of Mashhad and the Optimization Laboratory of Ferdowsi University of Mashhad. The third author would like to thank the Mosaheb Institute of Mathematics at Kharazmi University for supporting this work.

References

[1] Aaker, D., Tyebjee, T. (1978). "A model for the selection of interdependent R & D projects", IEEE Transactions on Engineering Management, 25, 30-36.

[2] Abbasi, D., Ashrafi, M., Ghodsypour, S.H. (2020). "A multi objective-BSC model for new product development project portfolio selection", Expert Systems with Applications, 162.

[3] Abyazi-Sani, R., Ghanbari, R. (2016). "An efficient Tabu search for solving the uncapacitated single allocation hub location problem", Computers and Industrial Engineering, 93, 99-109.

[4] Almeida, A.T.D., Duarte, M.D.O. (2011). "A multi-criteria decision model for selecting project portfolio with consideration being given to a new concept for synergies", Pesquisa Operacional, 31, 301-318.

[5] Archer, N.P., Ghasemzadeh F. (1999). "An integrated framework for project portfolio selection", International Journal of Project Management, 17, 207-216.

[6] Caglar, M., Gurel, S. (2007). "Impact assessment based sectoral balancing in public R&D project portfolio selection", Socio-Economic Planning Sciences, 66, 68-81.

[7] Carazo, A.F. (2015). "Multi-criteria project portfolio selection", In Handbook on Project Management and Scheduling, Springer, 709-728.

[8] Carazo, A.F., Gómez T., Molina, J., Hernández-Díaz, A.G., Guerrero, F.M., Caballero, R. (2010). "Solving a comprehensive model for multiobjective project portfolio selection", Computers and Operations Research, 37, 630-639.

[9] Cooper, R.G., Edgett, S.J., Kleinschmidt, E.J. (1999). “New product portfolio management: practices and performance”, Journal of Product Innovation Management, 16, 333-351.

[10] Crama, Y., Schyns, M. (2003). “Simulated annealing for complex portfolio selection problems”, European Journal of Operational Research, 150, 546-571.

[11] Doerner, K., Gutjahr, W.J., Hartl, R.F., Strauss C., Stummer, C. (2004). “Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection”, Annals of Operations Research, 131, 79-99.

[12] Gendreau, M., Potvin, J.Y. (2010). “Handbook of metaheuristics”, Springer, New York.

[13] Ghasemzadeh, F., Archer, N., Iyogun, P. (1999). “A zero-one model for project portfolio selection and scheduling”, Journal of the Operational Research Society, 50, 745-755.

[14] Golabi, K., Kirkwood, C.W., Sicherman, A. (1981). “Selecting a portfolio of solar energy projects using multiattribute preference theory”, Management Science, 27, 174-189.

[15] Goldberg, D.E. “Genetic algorithms in search”, Optimization, and Machine Learning, Addison-Wesley, Reading, Massachusetts.

[16] Hansen, P., Mladenović, N. (2006). “First vs. best improvement: An empirical study”, Discrete Applied Mathematics, 154, 802-817.

[17] Hill, R.A. (2010). “Portfolio theory and financial analyses”, Economics.

[18] Hoseini, A.R., Ghannadpoura, S.F., Hemmati, M. (2020). “A comprehensive mathematical model for resource-constrained multi-objective project portfolio selection and scheduling considering sustainability and projects splitting”, Journal of Cleaner Production, 269.

[19] Hult, H., Lindskog, F., Hammarlid, O., Rehn, C.J. (2012). “Risk and portfolio analysis: Principles and methods”, Springer Science and Business Media.

[20] Korotkova, V., Wu, D. (2020). “Evaluating the quality of solutions in project portfolio selection”, Omega, 91.

[21] Li, X., Huang, Y.H., Fang, S.C., Zhang, Y. (2020). “An alternative efficient representation for the project portfolio selection problem”, European Journal of Operational Research, 281, 100-113.

[22] López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., Stützle, T. (2016). “The Irace package: Iterated racing for automatic algorithm configuration”, Operations Research Perspectives, 3, 43-58.

[23] Marcondes, G.A.B., Leme, R.C., Leme, M.S., da Silva, C.E.S. (2017). “Using mean-Gini and stochastic dominance to choose project portfolios with parameter uncertainty”, The Engineering Economist, 62, 33-53.

[24] Mavrotas, G., Makryvelios, E. (2021). “Combining multiple criteria analysis, mathematical programming and Monte Carlo simulation to tackle uncertainty in research and development project portfolio selection: A case study from Greece”, European Journal of Operational Research, 291(2), 794-806.

[25] Medaglia, A.L., Graves, S.B., Ringuest, J.L. (2007). “A multiobjective evolutionary approach for linearly constrained project selection under uncertainty”, European Journal of Operational Research, 179, 869-894.

[26] Medaglia, A.L., Hueth, D., Mendieta, J.C., Sefair, J.A. (2008). “A multiobjective model for the selection and timing of public enterprise projects”, Socio-Economic Planning Sciences, 42, 31-45.

[27] Mladenović, N., Hansen, P. (1997). “Variable neighborhood search”, Computers and Operations Research, 24, 1097-1100.

[28] Perez, F., Gomez, T., Caballero, R., Liern V. (2018). “Project portfolio selection and planning with fuzzy constraints”, Computers and Operations Research, 131, 117-129.

[29] Santhanam, R., Kyparisis, J. (1995). “A multiple criteria decision model for information system project selection”, Computers and Operations Research, 22, 807-818.

[30] Song, S., Yang, F., Xia, O. (2019). “Multi-criteria project portfolio selection and scheduling problem based on acceptability analysis”, Computers and Industrial Engineering, 135, 793-799.

[31] Stummer, C., Heidenberger, K. (2003). “Interactive R& D portfolio analysis with project interdependencies and time profiles of multiple objectives”, IEEE Transactions on Engineering Management, 50, 175-183.

[32] Wei, H., Niu, C., Xia, B., Dou, Y., Hu, X. (2020). “A refined selection method for project portfolio optimization considering project interactions”, Expert Systems with Applications, 142.

[33] Song, S., Yang, F., Xia, Q. “Portfolio selection of distributed energy generation projects considering uncertainty and project interaction under different enterprise strategic scenarios”, Applied Energy, 236, 444-464.

[34] Yang, F., Song, S., Huang, W., Xia, Q. (2015). “SMAA-PO: Project portfolio optimization problems based on stochastic multi-criteria acceptability analysis”, Annals of Operations Research, 233, 535-547.

[35] Yu, L., Wang, S., Wen, F., Lai, K.K. (2012). “Genetic algorithm-based multi-criteria project portfolio selection”, Annals of Operations Research, 197, 71-86.

[36] Zhang, X., Fangd, L., Hipel, K. W., Ding, S., Tan, Y. (2020). “A hybrid project portfolio selection procedure with historical performance consideration”, Expert Systems with Applications, 142.