www.editada.org

International Journal of Combinatorial Optimization Problems and Informatics, 13(2), May-Aug 2022, 98–113. ISSN: 2007-1558

Fuzzy Filter: a method to solve a dynamic portfolio selection problem with preference incorporation

Daniel A. Martínez-Vega[1], Laura Cruz-Reyes[1], Claudia Gómez-Santillán[1], Héctor Fraire[1], Nelson Rangel-Valdez[2]*

[1]*Tecnológico Nacional de México/I.T. Ciudad Madero, México*

[2]*Cátedras CONACyT-Tecnológico Nacional de México/I.T. Ciudad Madero, México*

*Correspondence: claudia.gomez@itcm.edu.mx

Abstract. Many real-world optimization problems are dynamics. A solution proposal is approached for the dynamic problem of selecting project portfolios with multiple objectives and decision makers' preferences. The objective is to determine the projects that optimize benefits considering the budgetary restrictions that change periodically. It is well-known that the problem's difficulty increases with the preferences of the decision maker and more than one objective to satisfy. This work presents a new formulation of the problem and a fuzzy method to incorporate the preferences, named fuzzy filter (FF). This method uses fuzzy outranking relations to include controlled intensification and diversification to the solution process. It keeps only non-dominated solutions in agreement with a decision maker for intensification. For diversification, it creates a nadir point from the filtered solutions and generates new solutions from this point. Two state-of-the-art algorithms were adapted to incorporate the proposed FF method. To validate this proposal, instances with controlled difficulty were generated from easy to difficult. The experimentation showed that the FF improves the ability of algorithms to approach the region of interest of the decision maker when compared against their dynamic version without preferences.

Keywords: Dynamic multi-objective problem, project portfolio selection, fuzzy preference incorporation, evolutionary algorithms.

Article Info

Received: August 23, 2021
Accepted: January 21, 2022

1 Introduction

A task carried out in many areas, such as government offices, research centers, and private sector companies, is to evaluate a set of projects that compete to be financed when the budget is limited. A subset of these projects, selected for financing, is called the project portfolio [1][2]. Choosing a portfolio that provides the most significant benefit is the Project Portfolio Selection (PPS) problem.

Although the fundamental PPS problem's computational complexity is considered NP-Hard, in the real world, it is often accompanied by other NP-hard problems making it even more complex [3]. In this paper, PPS involves the dynamic allocation of resources (DAR), called together DPPS, which consists of monitoring and periodic adjustment of the actions; these operations improve the quality of a portfolio due to the more significant benefit they produce in the medium and long-term [3]. Although, in practice, many mathematical models and heuristics have limited utility, since they do not consider the intrinsic dynamic nature of

the portfolio processes, a large part of the research focuses on the dynamic optimization of problems with a single criterion or the static optimization of multiple criteria, as in the cases of Mora et al. [4], and Martínez-Vega et al. [5], respectively.

We approach DPPS as a Dynamic Multi-objective Optimization Problems (DMOPs). Evolutionary Algorithms (EAs) have been widely used for solving multi-objective problems, more recently DMOPS, because they can deal with a set of possible solutions simultaneously, obtaining in a single execution an approximation of the optimal Pareto frontier. Although finding the Pareto frontier is not enough; it is part of solving a multi-objective problem (MOP); it can be said that the process is complete until the DM identifies the best compromise. The decision-making process, which includes the DM's preferences, can be integrated with the solver algorithm in three ways: *a priori*, *a posteriori*, or *interactively*.

When the search process is finished, the EAs generate a set of efficient solutions. An a posteriori preference incorporation method is needed to obtain only one solution (the best compromise) that supports the decision-making process. The Decision Maker (DM) is responsible for providing preferences for obtaining the best compromise [6]. The *a priori* preference incorporation can reduce the search space, filtering non-dominated solutions to progressively identify solutions closer to the *Region of Interest* (RoI), the zone of the Pareto frontier that is more aligned with the DM's preferences. This advantage is fundamental compared to the first alternative.

Few studies have been conducted to solve DMOPs and less for solving real-world problems with dynamic multi-objective evolutionary algorithms (DMOEAs) [7][8]. Besides, limited research has been developed to incorporate preferences into dynamic multi-objective algorithms [8][9]. Therefore, more research is required to create an efficient decision-making process for these problems.

To contribute to DMOPs research, this paper proposes the Fuzzy Filter (FF), an a priori method to incorporate preferences into DMOEAs, particularly for the Dynamic Multi-objective Project Portfolio Selection Problem (DMO-PPSP). The FF method uses fuzzy outranking relations to keep only non-dominated solutions that agree with the DM and create a nadir point from the filtered solutions to generate new solutions from this point. These actions of intensification and diversification, respectively, can guide the search toward the RoI.

This paper is structured as follows. The first section presents the introduction to the research problem. The second section contains the background. The third section is about the formulation of the DMO-PPSP with preferences and the generation of instances with difficulty controlled by correlation. The fourth section explains FF; the proposed a priori preference incorporation method; also, the adaptation of two EAs to incorporate FF and dynamism. Section five presents the experiments with a new dynamic optimization benchmark, and the results are discussed. Finally, the conclusions and future work are detailed.

2 Background

2.1 Preference Incorporation Strategies

The DM is the centerpiece for the different methods of incorporating preferences; this can be a person or group of people whose preference system is used to guide the search toward the RoI, where the best compromise is found. There are three ways a DM can incorporate his preferences when solving a MOP: a priori, a posteriori, and interactively [10].

- In an a priori method, the DM's preferences are incorporated before executing the algorithm. Thus, the preference information guides the process optimization. However, the DM may not be satisfied with the solutions obtained. Therefore, it requires a subjective adjustment in the preference parameters, which is difficult since the DM knows the results until the optimization algorithm terminates [11]. This difficulty can be overcome with automatic methods of elicitation of preference parameters [12].
- An a posteriori method incorporates preferences at the end of the optimization process, leaving aside complex implications about DM preferences; the disadvantage emerges when the number of functions objectives increases since it becomes difficult to select the best compromise [13][14].
- Interactive approaches involve more participation of the DM in the solution process since as the process progresses, the preferences must be specified interactively and adjusted to the DM aspiration levels [15].

The interactive approach and the a priori approach provide relevant advantages over the a posteriori one. They increment the selective pressure toward solutions closer to the RoI, helping to find better solutions. Besides, they obtain a short number of

candidate solutions, reducing the DM's cognitive effort to choose the best compromise. The a priori approach is the only one that does not demand preference relations with full comparability and transitivity.

In this work, the incorporation of a priori preferences is applied. Different methods are available, among them: weighted approaches, fuzzy logic, lexicographic method, programming by goals, delimited objective function, physical programming. This research follows the lexicographic approach [16].

2.2 Outranking model

The model of outranking relationships proposed by Fernández et al. [16] is based on the preference relationships raised by Roy [17]. The model determines the quality of the solutions using Pareto dominance and outranking relationships, which provide greater discriminatory capacity. The definition of a possible preference relationship between a pair of solutions depends on some parameters and on the degree of credibility of the statement "x is at least as good as y", denoted by $\sigma(x,y)$. This value can be calculated using some methods reported in the literature, for example, ELECTRE [18][19] and PROMETHEE [20][21][22]. In this research work, the calculation of $\sigma(x,y)$ is based on the ELECTRE III method [18]; this is obtained in the same way throughout the entire document. The parameters used in the model to determine an outranking relation are as follows: an acceptable credibility threshold λ (determines the level of stringency of the outranking relation); asymmetry parameter ε (ensures the indifference relationship) and an asymmetry parameter β (ensures the strict preference relationship or k-preference). The outranking model can establish any of the following preference relationships for each pair of solutions (x,y).

- *Strict preference* (xPy). It represents the situation where the DM has clear and well-defined reasons to justify the choice of x over y. For a strict preference relationship to be established, at least one of its conditions must be met.
- *Indifference preference* (xIy). This relationship represents the situation in which the DM has clear and positive reasons that justify an equivalence between x and y. Thus, indifference is satisfied when the two conditions that represent it are fulfilled.
- *Weak preference* (xQy). The situation where the DM wavers between xPy and xIy. This relationship is a consequence of the conjunction of the conditions that represent it.
- Incomparability preference (xRy). It simulates the situation in which the DM perceives a high degree of heterogeneity between x and y. Besides, the DM cannot express a preference in favor of either.
- *k-preference* (xKy). Represents the situation where the DM wavers between xPy and xRy. This relationship is fulfilled if the three conditions that identify it are satisfied.

2.3 Classification of Dynamic Problems

In the scientific literature, it exists different ways to classify dynamic problems; among them are those based on [7][23], see Figure 1.

- *Frequency*. This classification considers the time elapsed between one change and another and the time spent adapting the problem between changes, which is inversely proportional.
- *Severity*. It is based on the strength of the impact of changes in the problem. It is easier to converge to the optimum when the changes are small because the new search space is close to the one already explored.
- *Predictability*. When there is a dependency between the solutions from one period to another, it is said to be a predictable problem; otherwise, it is said that we are talking about random changes and an independent or unforeseen problem.
- *Relationship between the Pareto Optimal Front (FOP) and the Pareto Optimal Solution (SOP)*. Four problems have been identified within this classification.

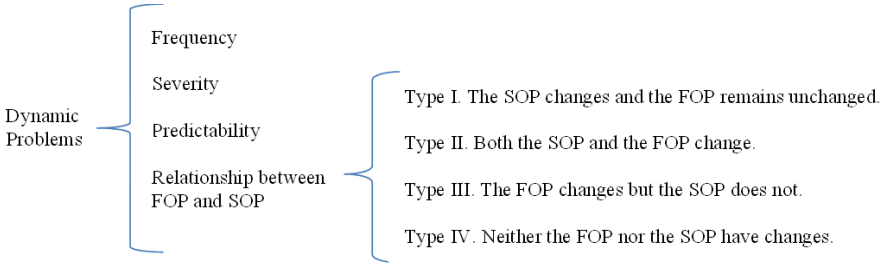

Fig. 1. Classification of dynamic problems

2.4 Dynamic Change Detection Mechanism

According to Baykasoğlu & Ozsoydan [24], the dynamic changes can arise from two sources, as shown in Figure 2, the first arises from changes in the problem data, as it happens in real-world -costs, capacities, budgets, among others. These changes can arise from external agents such as the global market or internal agents such as the maintenance of infrastructures; these are known as non-dimensional changes. The second source arises from changes in the domain of the problem, for example, in the definition of variables, parameters, and constraints. This second source is known as dimensional changes.

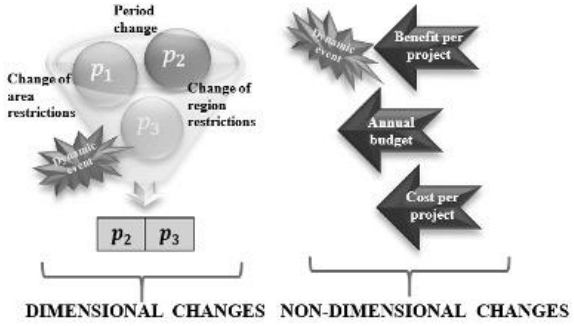

Fig. 2. Classification of the dynamic change detection mechanism

The case study of this work (DMO-PPSP) involves non-dimensional changes because it handles variations in the annual budget, the costs of financing each project, the benefits that each project produces, among others with the time course.

There are several techniques to detect the occurrence of changes, then the change detection technique used in this work is described:

1. We use two parameters to detect the change of period, the *number of periods to be evaluated* (*T*) and the *number of evaluations of the objective function* (max_eval).
2. Initially, the algorithm calculates a *base* variable equal to (*max_eval* / *T*); This variable controls the same number of objective function evaluations for each period.
3. At the end of each iteration of the algorithm, the floor of the current number of evaluations of the objective function is divided between the *base* value calculated in step 2; this determines the period in which this calculation is located.
4. After the calculation of step 3, if the value obtained is different from the defined per period, it is considered a period change.

2.5 Dynamic performance indicator

In the literature, we can find metrics to measure the quality of a specific characteristic of the resulting solutions set, for example, its dispersion or how uniformly distributed are the solutions in the POF, its proximity to the optimal solutions, or the best solutions found. Two well-known metrics are the inverted generational distance and the hypervolume [7].

Since dynamic problems usually change their POF or their POS, or both, over time, a measurement with conventional metrics could generate an error that, according to the type of problem, it could be huge. Due to this, some authors propose using performance indicators based on the conventional metrics [25] in each of the periods (separately) as if they belong to independent static periods; after, the average of the obtained values is calculated [26]. As an example, two performance indicators for dynamic problems are formulated with Equations 2 and 4; both are defined from indicators for static problems in Equations 1 and 3, respectively.

$$IGD(PF^*, PF) = \frac{\sum_{v \in PF^*} d(v, PF)}{|PF^*|} \tag{1}$$

$$MIGD = \frac{1}{T}\sum_{t=1}^{T} IGD\,(PF_t^*, PF_t) \tag{2}$$

$$HV(PF) = volume\left(\bigcup_{i=1}^{|PF|} h_i\right) \tag{3}$$

$$MHV = \frac{1}{T}\sum_{t=1}^{T} HV\,(PF_t) \tag{4}$$

Two static metrics are transformed into dynamic performance indicators in the previous example. Equation 1 is used to calculate the Inverted Generational Distance (IGD); this static IGD becomes a dynamical measure in Equation 2, which in the literature is called Modified Inverted Generational Distance (MIGD). In Equations 3 and 4, the same thing happens; the Hypervolume (HV) metric becomes the Modified Hypervolume (MHV). In the Equations 2 and 4, *T*, *PF**, and *PF* are the number of periods, the POF and the POS, respectively.

Figure 3 shows graphically the calculation of MIGD. Due to each period producing a different PF*, the IGD for each period is calculated independently. In the end, the MIGD is the average of all IGDs.

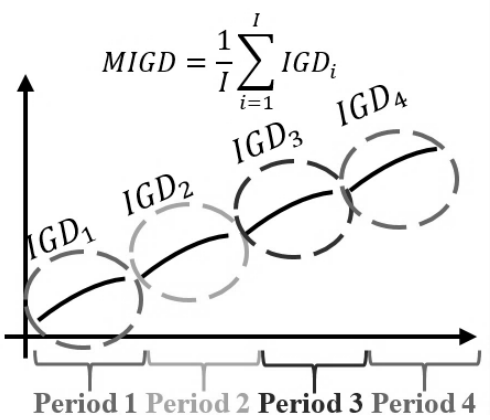

Fig. 3. Example of the calculation of the MIGD indicator for dynamic problems.

3 The dynamic multi-objective portfolio selection problem

3.1 Elements of the project portfolio selection

The main elements of the Project Portfolio Selection problem enriched with dynamic resource allocation are as follows:

- *A set of N projects requiring financing*. The problem lies in that a limited budget is available; therefore, these projects compete for obtaining the necessary resource to be carried out.
- *The number of objectives O that the DM seeks to cover best*. They let measure the quality of portfolios and projects.
- *A set of areas and regions*. Each project x_j is assigned to a specific area a_j and region g_i; each has minimum and maximum budget limits, so it must comply with all of them for a portfolio to be feasible.
- *The number of periods T that be analyzed* (e.g., days, months, or years). In a dynamic context, many elements can change between periods; among them are the budget for the period and the cost per project.
- *The amount of the budget assigned for each period*. These amounts are represented by the vector $B = [B_1, B_2, B_3, \ldots, B_T]$.

Project representation

Each project x_j has different characteristics as time passes; as it happens in real life, the costs of things are not usually the same with time, either the benefit obtained from them. This characteristic corresponds to the problem of dynamic resource allocation.

Therefore, if a project *j* is financed at time *t*, the benefit of this project *j* contributes to each objective *o* only for the period specified by *t*. This benefit is represented as follows: $b_{j,t} = (b_{j,t}^1, b_{j,t}^2, b_{j,t}^3, \ldots, b_{j,t}^o)$, where, $b_{j,t}^o$ is the benefit contributed by project *j*, to objective *o* at time *t*.

Regarding financing costs, something very similar happens to what happens with benefits; it should be noted that this vision of the problem does not work with financing partial; this way of approaching it only considers within the portfolio those projects whose financing is total. The costs associated with each of the projects at a time t are represented as follows: $c_t = (c_{1,t}, c_{2,t}, c_{3,t},\ldots, c_{j,t})$, where $c_{j,t}$ is the cost of project j in period t.

Portfolio Representation

A portfolio X is made up of all the vectors that contain the projects to be financed at each moment t, $X = (x_{1,t}, x_{2,t}, x_{3,t}, \ldots, x_{N,t})$, these are integrated with a matrix two-dimensional of size $N \times T$. The cost associated with each portfolio in a period t can be calculated as

$$C_{X,t} = \sum_{j=1}^{N} c_{j,t} \cdot x_{j,t} \quad \forall t, \tag{5}$$

where, $C_{X,t}$ represents the cost of financing portfolio X at time t, which is calculated with the sum of the products of the costs $c_{j,t}$ associated with each project j according to time t, $\forall j \in \{1,2,\ldots, N\}$ and the binary vector X, which indicates which projects are considered within of the portfolio as indicated by Equation 6.

$$x_{j,t} = \begin{cases} 1 & \text{if the project is in the portfolio at time } t, \text{ and} \\ 0 & \text{in another case.} \end{cases} \tag{1}$$

The principal feasibility condition is budgetary; therefore, for a portfolio X to be considered feasible, its cost $C_{X,t}$ should not exceed the budget B_t for the period t it belongs to. This constraint is expressed as

$$C_{X,t} \leq B_t \tag{7}$$

3.2 Formulation of DMO-PPSP

The problem of project portfolio selection is formulated as a multi-objective optimization problem. This model is based on the well-known binary formulation that has been studied in the literature, in works such as of the Stummer and Heidemberger [27]; Kremmel et al. [28]; Amiri [29]; and Fernandez et al. [30]. Next, a brief description of the basic mathematical model is made.

A portfolio is a set of maximum N projects $X = \{x_1, x_2, \ldots, x_N\}$ that needs financing some required cost $C = \{c_1, c_2, \ldots, c_N\}$ from a budget that is usually limited. Each project provides a set of profits $P = \{p_1(X), p_2(X), \ldots, p_O(X)\}$, which are intended to be improved per each objective o; however, they are often in conflict with each other, making this task complicated. At the same time, each project has associated a region $G = \{g_1, g_2, \ldots, g_{ng}\}$, which can be a geographical region, and an area $A = \{a_1, a_2, \ldots, a_{na}\}$, which represents a specific work area. This problem now becomes one in which the portfolio is a set of projects, but in time $t \in T$: $X = \{x_{1,t}, x_{2,t}, \ldots, x_{N,t}\}$.

To define the DMO-PPSP, we consider the following additional elements.

Decision variables:
$x_{j,t}$ – A binary matrix that represents if the project j is funded (1) in the period t or not (0),
$B_{a,t}$ – Budget required for the area a in the period t, and
$B_{g,t}$ – Budget required for the region g in the period t.

Constants:
T – number of periods,
na – Number of areas,
ng – Number of regions,
$a_{j,l}$ – A binary matrix that indicates if the project j belongs to the area l,

$g_{j,g}$ – A binary matrix that indicates if the project j belongs to the region g,
B_t – The annual budget for the year t,
$B_{l_{min},t}$ – Minimum budget for the area l in the period t,
$B_{l_{max},t}$ – Maximum budget for the area l in the period t,
$B_{g_{min},t}$ – Minimum budget for the region g in the period t,
$B_{g_{max},t}$– Maximum budget for the region g in the period t,
$b_{j,t}^{o}$ – The benefit of the project j to the objective o in the period t, and
$c_{j,t}$ – A matrix that stores the cost of each project j in the period t.

The quality of portfolio *x* is determined by the union of the benefits of each of the projects that compose it at time *t* and is expressed as

$$P(x) = \{\langle p_{1,t}(x), p_{2,t}(x), \dots, p_{O,t}(x) \rangle\} \tag{8}$$

where $p_{j,t}(x)$ in its simplest form it is defined as

$$p_{j,t}(x) = \sum_{j=1}^{N} b_{j,t}^{o} x_{j,t}\,. \tag{9}$$

RF_t is denoted as the region of feasible portfolios at time *t*, and then the dynamic project portfolio problem is to identify one or more portfolios that solve

$$max_{x_t \in RF_t}\{P(x)\}, \tag{10}$$

subject to a set of restrictions that define RF_t by the formulas:

$$\left(\sum_{i=1}^{n} x_{j,t}\, c_{j,t}\right) \leq B_t \qquad \forall_t, \tag{11}$$

$$B_{l_{min},t} \leq B_{l,t} \leq B_{l_{max},t} \qquad \forall_{l,t}, \tag{12}$$

$$B_{g_{min},t} \leq B_{g,t} \leq B_{gmax,t} \qquad \forall_{g,t}, \tag{13}$$

$$B_{l,t} = \sum_{i=1}^{n} x_{j,t}\, c_{j,t}\, a_{j,l} \qquad \forall_{l,t}, and \tag{14}$$

$$B_{g,t} = \sum_{i=1}^{n} x_{j,t}\, c_{j,t} g_{j,g} \qquad \forall_{g,t}. \tag{15}$$

where $t \in \{1,2,\dots,T\}, i \in \{1,2,\dots,N\}$, $l \in \{1,2,\dots,na\}$, $r \in \{1,2,\dots,ng\}$, and $o \in \{1,2,\dots,O\}$.

Equation 11 deals with the budget limit for the period that should not be exceeded. Equations 12 and 13 indicate that the budgets by area and region did not exceed the limits assigned to each period. Finally, Equations 14 and 15 verify that the budgets of each area and region for each period have been calculated correctly.

3.3 Generation of instances with preferences and difficulty level

We incorporate Fernández´s outranking model [31] to guide the search on the solution space *RF* toward the RoI of a DM specified by his(her) preferences. The parameters of the outranking model reflect the preferences. The parameter values (weight and thresholds) are part of a problem instance, assuming they can be inferred from solution examples. The preference incorporation is

described in section 4.1, and the instances generation with preferences and controlled difficulty is described below. This set and its generator are part of the proposed benchmark[1] in this article.

The dynamic section of the instances is repeated in structure, but not in the data, the number of times indicated by the number of periods. The dynamic (changing) part of these instances is budget, cost of projects, area, and region to which each project belongs, objective limits, area limits, region limits, and benefits. The difficulty level is based on the proposal in [32] and [33], considering that a closer correlation between the benefits and the costs of the projects for this kind of problem produces a higher level of difficulty in finding their solution. For easy instances, the uncorrelation was obtained, generating these random variables independently. The costs $c_{j,t}$ are chosen randomly in medium and hard instances, but the benefits $b_{i_t}^o$ depend on the costs and a random variation, whose reduction increases the correlation and, therefore, the difficulty. Equation 16 calculates the benefits in instances of medium difficulty, and Equation 17 is for hard difficulty. All random numbers are uniformly distributed.

$$b_{j,t}^o = c_{j,t} + \alpha_{i,j} \qquad \forall i,j \tag{16}$$

$$b_{j,t}^o = c_{j,t} + \beta_{i,j} \qquad \forall i,j \tag{17}$$

where:

$j =$ number of project, $o =$ number of objectives, $t =$ period,

$\alpha_{i,j} =$ a random number in $\left[\frac{upper\_cost\_limit}{2}, upper\_cost\_limit\right]$, and

$\beta_{i,j} =$ a random number in $\left[\frac{upper\_cost\_limit}{10}, upper\_cost\_limit\right]$.

For the case study the common configuration used by the instance generator were the following: periods $T = 3$, objectives $o = 2$, number of areas $a = 3$, number of regions $g = 2$, areas budget $B = \{200000, 300000, 280000\}$, weights of the objects $w = \{60, 40\}$, veto thresholds $vt = \{51000, 15000\}$ and indifference thresholds $it = \{7500, 1500\}$, limits of cost $c = \{2500, 5000\}$ and limits of benefits $b = \{\{1000, 10000\}, \{500, 8000\}\}$ for easy instances. With this configuration, 15 instances of different difficulty levels were generated; the general descriptions of each of these instances are shown in Table 1.

Table 1. DMO-PPSP instances with difficulty based on the correlation of cost and benefit

Difficulty level (correlation)	Name	Number of objectives	Number of projects	Periods
Easy (uncorrelated)	Easy_1, Easy_2, …, Easy_5	2	100	3
Medium (regular)	Medium_1, …, Medium_5			
Hard (high)	Hard_1, Hard_2, …, Hard_5			

4 Preference incorporation in metaheuristic algorithms

A new method for incorporating DM´s preferences in an algorithm is proposed and detailed in this section. Two MOEAs from the literature were adapted for evaluating the proposal: DNSGA-II (dynamic non-dominated sorting algorithm II) and ABySS (Archive-Based hYbrid Scatter Search).

4.1 Proposed method: Fuzzy Filter

The proposed Fuzzy Filter (FF) uses fuzzy outranking relations that are the base of the outranking model of Fernández et al. [31]. The model guides the search toward the RoI to find the best solution, called the *best compromise*. The model evaluates the fuzzy statement '*x is as least as good as y*', denoted '*x outranks y*' or *xSy*, where *x* and *y* are alternative solutions. In Algorithms 1 and 2, the $outranking\_method$ evaluates *xSy* for a given set of solutions following the below three steps and returns the solutions non outranked by any other.

[1] The proposed benchmark can be found in "repository on dynamic multi-objective optimization" at http://www.cruz-reyes.com/repositories.html.

Outranking method. The first step is to infer the model's parameters using a preference disaggregation method from preferred solution examples given by the DM [34]; this step is repeated only if the preferences change over time. The second one is to determine the degree of credibility of *xSy* denoted by σ(*x*,*y*). According to the well-known Electre-III, the value of σ is calculated by multiplying the concordance index of *xSy* by the discordance index of *xSy* as defined in [31]; these measures characterize each problem objective's degree of concordance and discordance between the solution alternatives being evaluated, respectively. The third step is determining and counting preference relations between all pairs of solutions based on the inferred parameters and σ(*x*,*y*). The strict preference *P* is defined when the DM has clear and well-defined reasons to prefer *x* over *y*. The indifference *I* occurs when the DM notes a high degree of equivalence between *x* and *y*. The weak preference *W* occurs when the DM vacillates between *xPy* or *xIy* [31]. Finally, the best compromise solution is a non-dominated solution, which maximizes three objectives related to counting not strictly and weakly outranked [31]. The term non-outranked front is used for these three fuzzy objectives, while the non-dominated front is for the original objectives of the problem. FF is implemented at two points of the algorithms, at the time of generating the initial population (Algorithm 1) and at the end of *n* full iterations (Algorithm 2).

***Incorporation of FF to the initial set of solutions* (*Algorithm 1*).** This algorithm constructs a set of random, feasible, and non-repeated solutions; its size is five times the size of the initial population (Step 1). Generating such a large initial set of solutions provides greater diversity when filtering these solutions using the above-described outranking method. The solutions, obtained by filtration of the zero-non-dominated front, become part of the initial population (Step 2). The nadir point (NP, the worst solutions by objective) is obtained from the filtered solutions (Step 3). Finally, the population is completed with randomly generated solutions that exceed at least one objective to the NP (Steps 4-7). Thus, the *FF_initial* procedure generates an initial population of good (using filtering) and diverse (using NP) individuals, attempting to influence the search to intensify and diversify simultaneously.

Algorithm 1 *FF_initial*

Input: population size (N)
Output: initial population (P_0)

1. $P_x \leftarrow generate\_random\_solutions()$
2. $P_0 \leftarrow outranking\_method(P_x)$
3. $nadir\_point \leftarrow get\_nadir\_point(P_0)$
4. $\boldsymbol{while}(|P_0| < N)$
5. $\quad new\_solution \leftarrow generate\_new\_solution()$
6. $\quad \boldsymbol{if}(isBetter(new\_solution, nadir\_point))$
7. $\quad\quad P_0 \leftarrow P_0 \cup new\_solution$
8. ***return*** P_0

***FF at the end of each m iterations* (*Algorithm 2*).** Repeatedly, at the end of *m* iterations, the current population P_i is submitted to the *FF_iterative* method to generate a filtered population in P_{i+1}. In the beginning, the zero-non-outranked front of P_i is extracted (Step3) and included in P_{i+1} (Step 6); this is repeated with what remains in P_i (Step 7) until P_{i+1} reaches a size of 30% of the population size (Steps 2-7). If the addition of the extracted front exceeds the said percentage (Step 4), the extracted set is reduced, refilling with solutions taken randomly from it (Step 5). The NP is obtained from the solutions in (P_{i+1}) (Step 8). Finally, the population is completed with feasible and non-repeated randomly generated solutions that exceed that point in at least one objective (Steps 10-15). Due to the greed of the last step, it is given a maximum of *m* opportunities to find a random solution that exceeds the NP (Step 12); otherwise, a simple random solution that is only feasible and not repeated be included. As can be seen, it is a technique focused on intensification; however, it allows diversifying when the intensification reaches a very high point (when it is difficult for the algorithm to find better solutions than the NP).

4.2 Dynamic solvers with the proposed Fuzzy Filter

In this section, a short description is made of the DMOEAs proposed to solve dynamic problems with preferences. The dynamic condition incorporates the change detection method and the change adaptation operator, called Hypermutation [7]. The proposed FF method, introduced in Section 3.1, integrates DM´s preferences without modifying the basic structure of a DMOEA. The description of DNSGA-II is found in [35], the paper that gives rise to Algorithm 3; the difference is the change in Step 2 and the additional Steps 18, 20 and 21, to incorporate the FF.

Algorithm 2 *FF_iterative*

Input: current population (P_i), population size (N), interval size for *FF_iterative* (m)
Output: new population **(** P_{i+1})

1. $P_{i+1} \leftarrow \emptyset$
2. $\boldsymbol{while}(|P_{i+1}| < N * 0.3)$
3. $P_{aux} \leftarrow outranking\_method(P_i)$ //solutions non-outranked by any other
4. $\boldsymbol{if}(|P_{i+1}| + |P_{aux}| > N * 0.3$
5. $P_{aux} \leftarrow reduce(P_{aux})$
6. $P_{i+1} \leftarrow P_{i+1} \cup P_{aux}$
7. $P_i \leftarrow P_i - P_{aux}$
8. $nadir\_point \leftarrow get\_nadir\_point(P_{i+1})$
9. $counter \leftarrow 0$
10. $\boldsymbol{while}(|P_{i+1}| < N)$
11. $new\_solution \leftarrow generate\_new\_solution()$
12. $\boldsymbol{if}(isBetter(new\_solution, nadir\_point)\ or\ counter > m\ )$
13. $P_{i+1} \leftarrow add(new\_solution)$
14. $counter \leftarrow -1$
15. $counter \leftarrow counter + 1$
16. $\boldsymbol{return}(P_{i+1})$

Algorithm 3 *DNSGA_II_FF*

Input: population size (N), interval size to apply *FF_iterative* (m)
Output: zero non-outranked front from the final population of each time change (S_t)

1. $i \leftarrow 0; t \leftarrow 0$
2. $P_i \leftarrow FF\_generate\_initial\_population()$
3. $P_i \leftarrow non\_fast\_dominated\_sorting(P_i)$
4. $Q_i \leftarrow generate\_new\_offspring\_populatio(P_i)$
5. $\boldsymbol{while}(\neg\ termination\_condition)$ //evaluations of the objective function
6. $P_i \leftarrow P_i \cup Q_i$
7. $F \leftarrow non\_fast\_dominated\_sorting(P_i)$
8. $P_{i+1} \leftarrow \emptyset; j \leftarrow 0$
9. $\boldsymbol{do}$
10. $P_{i+1} \leftarrow P_{i+1} \cup F_j$
11. $j \leftarrow j + 1$
12. $\boldsymbol{while}(|P_{i+1}| + |F_j| \leq N)$
13. $\boldsymbol{if}(|P_{i+1}| \neq N)$
14. $F_j \leftarrow crowding\_distance(F_j)$
15. $P_{i+1} \leftarrow P_{i+1} \cup F_j\ [1:(N - |P_{i+1}|)]$
16. $\boldsymbol{if}(time\_change\_detected)$
17. $S_t \leftarrow outranking\_method(P_{i+1})$ //solutions non-outranked by any other
18. $t \leftarrow t + 1$
19. $hypermutation(P_{i+1})$
20. $\boldsymbol{if}(i\ mod\ m = 0)$
21. $P_{i+1} \leftarrow FF\_iterative(P_{i+1})$
22. $Q_{i+1} \leftarrow generate\_new\_offspring\_population(P_{i+1})$
23. $i \leftarrow i + 1$
24. **return** S_t

Algorithm 4 is based on the original ABySS algorithm [36], with preference management and working on dynamic problems. The original algorithm operates on a set of solutions called the reference set and a population *P*. The solutions in the reference set are grouped systematically into two subsets. The reference set is a collection that contains both high-quality solutions and diverse solutions that are used to generate new ones. The first one contains the best quality solutions in *P*, while the second one includes those solutions that incorporate diversity to the set. The external file keeps a historical record of the non-dominated solutions found during the search, trying, at the same time, to keep those that produce a better distribution on the Pareto front. The proposed algorithm generates a population with the *FF initial* method (Step 2). Subsequently, in Step 3, an improvement is made in the initial solutions (P_i). In Step 4, the main loop begins, with the number of objective function evaluations as stopping criterion. Reference sets (ref_set_1 and ref_set_2) are generated and evaluated (Steps 5-6). In Step 8, the condition of the internal loop is verified, which marks that the loop is repeated while new solutions are found during the previous iteration. This inner loop is first responsible for defining all possible combinations of parents from ref_set_1 and ref_set_2 (Step 10), to later combine and improve these combinations generating the $offsprings$ set (Steps 11-12); with this, the reference sets are updated (Step 13). In Step 4, the *external file* is updated. The restart method generates P_{i+1} taking all solutions in *ref_sets* and *external_file* and completing with P_i. After, the algorithm checks for dynamic changes; if any, zero non-outranking front (S_t) is obtained through the outranking method, and the hypermutation is applied for starting a new period (Steps 16-19). If the algorithm is in the *m-th* iteration, the *FF_iterative* method filters and fills P_{i+1} (Steps 20-21). Finally, P_{i+1} is improved (Step 22), and the counter used to activate *FF_iterative* is increased (Step 23).

Algorithm 4 *DABySS-FF*

Input: the size of all sets, interval size to apply FF_iterative (m)

Output: zero non-outranked front from the final population of each time change (S_t)

1. $i \leftarrow 0;\ \ t \leftarrow 0;\ external\_file \leftarrow \emptyset;$
2. $P_i \leftarrow FF\_generate\_initial\_population()$
3. $P_i \leftarrow improve(P_i)$
4. $\boldsymbol{while}(\neg\ termination\_condition)$
5. $\quad ref\_sets \leftarrow generate\_ref\_sets(P_i)$ //the object *ref_sets* contains ref_set_1 and ref_set_2
6. $\quad evaluate\_ref\_sets(ref\_sets)$
7. $\quad new\_solution \leftarrow true$
8. $\quad \boldsymbol{while}(new\_solution)$
9. $\quad\quad new\_solution \leftarrow false$
10. $\quad\quad subsets \leftarrow generate\_subsets(|ref\_set_1|, |ref\_set_2|)$
11. $\quad\quad offsprings \leftarrow combination(ref\_sets, subsets)$
12. $\quad\quad offsprings \leftarrow improve(offsprings)$
13. $\quad\quad [ref\_sets, new\_solution] \leftarrow update\_ref\_sets(ref\_sets, offsprings)$
14. $\quad\quad external\_file \leftarrow update(external\_file, ref\_sets)$
15. $\quad\quad P_{i+1} \leftarrow restart(P_i, ref\_set_1, external\_file\ )$
16. $\quad\quad \boldsymbol{if}(change\_detected)$
17. $\quad\quad\quad S_t \leftarrow outranking\_method(P_{i+1})$
18. $\quad\quad\quad t \leftarrow t + 1$
19. $\quad\quad\quad hypermutation()$
20. $\quad\quad \boldsymbol{if}(i\ mod\ m = 0)$
21. $\quad\quad\quad P_{i+1} \leftarrow FF\_iterative(P_{i+1})$
22. $\quad\quad P_{i+1} \leftarrow improve\ (P_{i+1})$
23. $\quad\quad i \leftarrow i + 1$
24. **return** S_t

5 Computational experiment

The experimentation was carried out by solving the instances of DMO-PPSP (Table 1) through two dynamic algorithms with preferences, DNSGA-II-FF and DABySS-FF.

The hardware and software of the computer used for the experimentation are described in Table 2.

Table 2. Hardware and software

Hardware	Software
Intel Core i3-6100U processor at 2.3 GHz, 4 Gb of DDR4 memory 2133 MHz	O.S. Windows 10 x64 Java language and JDK1.8

The quality indicators to evaluate the solutions obtained by the algorithms in the two experiments were defined as explained in Section 2.5. The four dynamic indicators considered are based on static indicators taken from [25]: 1) MIGD, based on the IGD metric, measures the average of the distances between each solution of the approximated RoI (that is called only RoI for ease from now on) to the closest solution in the output set for each metaheuristic; 2) MHV, based on HV metric, measures the search space that covered the set of obtained solutions; 3) MGD, based on the GD metric, calculate the average of the distances between each solution of the output set to the closest solution in the RoI for each metaheuristic; and the 4) Mspread, based on the spread metric, measures the uniformity of the dispersion of obtained solutions. The RoI in this experiment was obtained by joining all the final outputs of each algorithm (belonging to the same period) and using the outranking method to get the zero non-outranked solutions of this entire set. All the calculations used standardized data.

The algorithms were configured using the proposal of Ghahremani & Naderi [37]. Also, the model parameters were configured with a preference disaggregation analysis (PDA), which uses examples of preferred solutions provided by the DM, as proposed by Covantes et al. [34]; This approach allows overcoming one of the limitations of the outranking methods pointed out by Coello [38].

Results: Performance evaluation of dynamic multi-objective algorithms

The objective of the first experiment is to identify with which method of incorporation of preferences the algorithm manages to maximize performance. The algorithms used were DNSGA-II-FF and DABySS-FF; at the end of each period, if a change is detected in the objective function, restrictions, or search space of the problem, a hypermutation method is implemented to 40% of the population. The stop criterion used in this experiment is 25,000 evaluations of the objective function. For this experiment, 30 runs were generated on each metaheuristic. For the experimentation, multi-objective instances created with the artificial instance generator were used; the data is shown in Table 3.

Table 3. Instance data

Name Instance	Objective Number	Projects number	Year number
DPPSSP_1	3	100	1
DPPSSP_2	3	100	3
DPPSSP_3	9	100	1
DPPSSP_4	9	100	3

As can be seen in Table 4 and 5, where the four indicators that evaluate the quality of the solutions obtained with three and nine objectives are shown for the two algorithms analyzed, it is observed that there is no variant of incorporation of preferences that predominates in all The experiments, that is, the results of the indicators show that by varying the number of objectives or the type of algorithm there is a change in the response of the algorithm's performance.

When working with three objectives for the two algorithms, it is observed that the variant of incorporation of preferences *a priori* is the one that achieves the best results in five of the eight indicators. For nine objectives for the two algorithms, the variants of incorporation Preferences *a priori* and Objective function preferences present the best values in the indicators; each one is the best in three of eight indicators. Finally, it could be concluded that the variant of incorporation of preferences *a priori* has the highest frequency of appearance in the experiments.

After selecting the strategy of incorporation of preferences *a priori*, a second experiment is proposed that has the objective of the experiment was to analyze the behavior of two proposed dynamic metaheuristic methods and the effect that FF produces on them in a bi-objective problem through instances with easy, medium, and hard difficulty. Table 1 shows the instances used to make 30 runs of the algorithms for each.

Table 4. Results of quality indicators on the solutions obtained with 3 objectives

Methodology	MGD	MIGD	Mspread	MHV
DNSGAII-FF				
No preferences	0.01385	0.01158	0.95953	9348.49
A priori preferences	0.01021	0.00992	0.97682	9388.16
A posteriori preferences	0.01894	0.01588	0.90394	9274.52
Objective function preferences	0.01289	0.01109	0.89277	9108.31
A priori + A posteriori + Objective function preferences	0.01787	0.01492	0.91658	9755.24
DABySS-FF				
No preferences	0.01628	0.01580	078092	5947.54
A priori preferences	0.01598	0.01499	0.88232	6093.48
A posteriori preferences	0.04929	0.04028	0.89029	5504.71
Objective function preferences	0.02884	0.02703	0.77920	5201.58
A priori + A posteriori + Objective function preferences	0.04186	0.03819	0.87905	5078.76

Table 5. Results of quality indicators on the solutions obtained with 9 objectives

Methodology	MGD	MIGD	Mspread	MHV
DNSGAII-FF				
No preferences	0.01347	0.00482	0.46653	9.01129E11
A priori preferences	0.01361	0.00455	0.44520	8.95738E11
A posteriori preferences	0.01429	0.00867	0.49442	9.00573E11
Objective function preferences	0.01429	0.00517	0.39985	8.53839E11
A priori + A posteriori + Objective function preferences	0.01906	0.01098	0.51208	8.91632E11
DABySS-FF				
No preferences	0.04678	0.04046	0.75839	7.82883E10
A priori preferences	0.04685	0.03595	0.77374	8.03848E10
A posteriori preferences	0.09478	0.09141	0.88394	5.24727E10
Objective function preferences	0.03285	0.03783	0.74495	7.14722E10
A priori + A posteriori + Objective function preferences	0.10578	0.09836	0.83874	6.13034E10

The main parameters used in the settings of both metaheuristics are: 25,000 evaluations of the objective function were used as a stopping criterion, a One-point cross was used with a 90% probability of use, a Binary mutation with a 2% probability of mutating, and a 70% probability of Hypermutation for each solution when a time change was detected. Besides, the population size for DNSGA-II-FF was 105 solutions, and for DABySS-FF, 40 solutions (20 in each reference set).

The closeness to the RoI is a relevant quality indicator for algorithms with preferences incorporation. Table 3 contains, for each instance and each algorithm, the medians of the MIGD of the 30 runs. A preliminary analysis of this table reveals some aspects of the impact of FF over the behavior of the algorithms; to facilitate this analysis, instances are organized by difficulty. For each instance, the numbers in parenthesis indicate the position of the best (1) to the worst (4) in performance measured by MIGD. For each test instance, there is a superiority marked by DABySS-FF concerning the rest of the metaheuristics; its dynamic base version DABySS presents better results than DNSGA-II and DNSGA-II-FF. Besides, FF improves the results when implemented in both algorithms, as shown in Figure 4. Another important fact is that when the difficulty of the instances increases, the performance gap between the base algorithms and their versions with FF becomes narrower. It is due to the greediness of the technique, being more complicated to obtain different random solutions, and with good fitness on instances with a lower variability of its correlated data, as explained in Section 2.2.

As shown in Table 6, similar calculations were obtained for all dynamic quality indicators to corroborate the previous performance observations statistically. For each instance and algorithm, the median of the values obtained from each dynamic quality indicator of the 30 runs was calculated. Non-parametric Friedman ranking tests were performed with the null hypothesis that the means of the results of two or more algorithms are the same. According to the Friedman test, a p-value of 0.005 demonstrates significant differences between the performance of the evaluated algorithms. Table 4 shows the Friedman ranking.

Table 6. MIGD medians comparison of DABySS, DNSGA-II, DABySS-FF, and DNSGA-II-FF

Algorithms	**Easy_1**	**Easy_2**	**Easy_3**	**Easy_4**	**Easy_5**
DABySS	0.096 (2)	0.039 (4)	0.033 (2)	0.039 (2)	0.035 (2)
DNSGA-II	0.113 (3)	0.014 (1)	0.162 (4)	0.202 (4)	0.097 (4)
DABySS-FF	0.025 (1)	0.026 (3)	0.024 (1)	0.024 (1)	0.026 (1)
DNSGA-II-FF	0.164 (4)	0.016 (2)	0.151 (3)	0.199 (3)	0.091 (3)
	Medium_1	**Medium_2**	**Medium_3**	**Medium_4**	**Medium_5**
DABySS	0.013 (2)	0.013 (2)	0.012 (2)	0.014 (2)	0.013 (2)
DNSGA-II	0.070 (4)	0.083 (3)	0.128 (4)	0.105 (3)	0.133 (3)
DABySS-FF	0.009 (1)	0.009 (1)	0.009 (1)	0.010 (1)	0.010 (1)
DNSGA-II-FF	0.068 (3)	0.099 (4)	0.113 (3)	0.127 (4)	0.148 (4)
	Hard_1	**Hard_2**	**Hard_3**	**Hard_4**	**Hard_5**
DABySS	0.010 (2)	0.011 (1)	0.008 (2)	0.011 (1)	0.009 (1)
DNSGAII	0.027 (3)	0.070 (3)	0.083 (4)	0.072 (3)	0.056 (2)
DABySS-FF	0.008 (1)	0.011 (1)	0.007 (1)	0.011 (1)	0.009 (1)
DNSGA-II-FF	0.042 (4)	0.068 (4)	0.075 (3)	0.068 (2)	0.056 (2)

Table 7. Friedman's Ranking comparison for three quality indicators

MGD		**MIGD**		**MHV**	
DABySS-FF	1.200	DABySS-FF	1.200	DNSGA-II-FF	1.600
DABySS	2.133	DABySS	2.067	DNSGA-II	1.667
DNSGA-II-FF	3.067	DNSGA-II-FF	3.333	DABySS	3.067
DNSGA-II	3.600	DNSGA-II	3.400	DABySS-FF	3.667

The results of MGD and MIGD, in Table 7, show how the proximity of the compromise solutions obtained by DABySS and DABySS-FF to the RoI is more significant. However, according to MHV, the DNSGA-II and DNSGA-II-FF metaheuristics show greater exploration or coverage in the search space. Although more experimentation is required and, above all, in-depth graphic analysis, this may be due to the average size of the set of compromise solutions obtained through DNSGA-II and DNSGA-II-FF, which is larger than the size obtained by the other pair of algorithms.

From the results of Table 6 and observed in Figure 4, it is inferred that the FF technique provides a more considerable improvement in the DABySS algorithm than in DNSGA-II. That is, at least for the set of instances used in this experiment, FF adapts better to the characteristics of DABySS than to those of DNSGA-II.

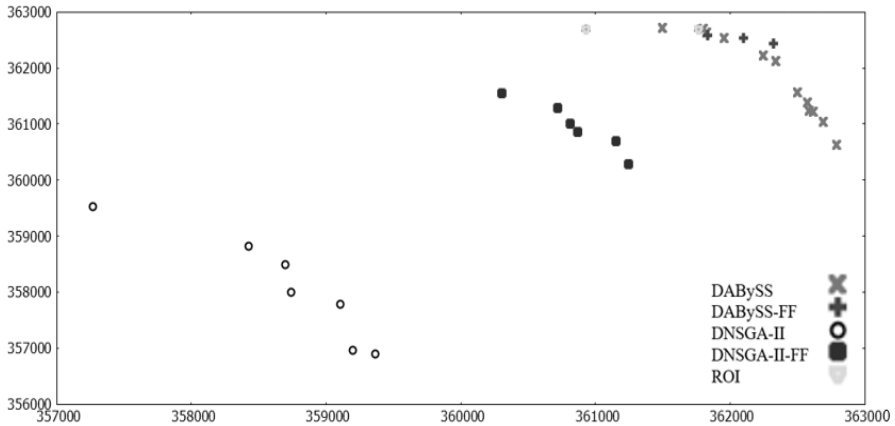

Fig. 4. Final solutions obtained by each algorithm for the Medium_1 instance

6 Conclusions

This paper presents a new formulation of the portfolio selection problem with dynamic and preferential conditions, called DMO-PPSP. To solve this problem, a novel Fuzzy Filter (FF) method is proposed to incorporate DM´s preferences into evolutionary algorithms with an a priori approach. Two algorithms were used in this work, DNSGA-II y DABySS. The first one was taken from state of the art, and the second one is a proposal, based on the first, for the management of dynamic problems; both were adapted incorporating FF. We called this pair of algorithmic proposals DNSGA-II-FF and DABySS-FF, respectively.

The proposed solution was validated with two experiments; in the first one, the objective is to identify with which method of incorporation of preferences the algorithm manages to maximize performance, and in the second experiment, the objective is to analyze the behavior of two proposed dynamic metaheuristic methods and the effect that FF produces.

As a result of the first experiment, the *a priori* preference articulation strategy is selected, which is applied in the second experiment in which we work with a total of 15 instances of DMO-PPSP were generated, divided into three groups of five instances each, with different levels of difficulty: easy, medium, and hard. This optimization benchmark is available for the community. The experimental results showed that the FF technique improves the ability of algorithms to approach the RoI when compared against their dynamic basic version; Although the versions with FF retained advantages at all instances' difficulty levels, the higher the difficulty level, the improvement was more minor; this is the subject of future work to improve the technique. Also, as future work, more tests and adjustments will be made to the proposed technique, focusing on multi-objective and many-objective problems to analyze their behavior.

Acknowledgments

Authors thanks to CONACYT for supporting the projects A1-S-11012 and 3058. Also, we thank TECNM for support of project 5797.19P.

References

1. Bastiani, S. S., Cruz-Reyes, L., Fernandez, E., Gómez, C., & Rivera, G. (2015). "An ant colony algorithm for solving the selection portfolio problem, using a quality-assessment model for portfolios of projects expressed by a priority ranking". In Design of Intelligent Systems Based on Fuzzy Logic, Neural Networks and Nature-Inspired Optimization (pp. 357–373). Springer, Cham. https://doi.org/10.1007/978-3-319-17747-2_28
2. Rivera, G., Gómez, C., Cruz, L., García, R., Balderas, F. A., Fernández, E. R., & López, F. (2012). Solution to the social portfolio problem by evolutionary algorithms. *International Journal of Combinatorial Optimization Problems and Informatics*, 3(2), 21–30.
3. Pajares, J., López, A., Araúzo, A., & Hernández, C. (2009, April). "Project Portfolio Management, selection and scheduling. Bridging the gap between strategy and operations". In XIII Congreso de Ingeniería de Organización (pp. 1421–1429).
4. Mora, T., Deny, S., & Marre, O. (2015). Dynamical criticality in the collective activity of a population of retinal neurons. *Physical review letters*, 114(7), 078105. https://doi.org/10.1103/PhysRevLett.114.078105
5. Martínez-Vega, D. A., Cruz-Reyes, L., Gomez-Santillan, C., Rangel-Valdez, N., Rivera, G., & Santiago, A. (2018). "Modeling and project portfolio selection problem enriched with dynamic allocation of resources". In Fuzzy logic augmentation of neural and optimization algorithms: theoretical aspects and real applications (pp. 365–378). Springer, Cham. https://doi.org/10.1007/978-3-319-71008-2_26
6. Cruz-Reyes, L., Medina-Trejo, C., Lopez-Irarragorri, F., Rivera, G., & Pérez-Villafuerte, M. (2015). Reduction of decision rules for project explanation on public project portfolio. *International Journal of Combinatorial Optimization Problems and Informatics*, 6(3), 5–21.
7. Azzouz, R., Bechikh, S., & Said, L. B. (2017). "Dynamic multi-objective optimization using evolutionary algorithms: A survey". In Recent advances in evolutionary multi-objective optimization (pp. 31–70). Springer, Cham. https://doi.org/10.1007/978-3-319-42978-6_2
8. Helbig, M., Deb, K., & Engelbrecht, A. (2016, July). "Key challenges and future directions of dynamic multi-objective optimization". In 2016 IEEE Congress on Evolutionary Computation (CEC) (pp. 1256–1261). IEEE. https://doi.org/10.1109/CEC.2016.7743931
9. Andersson, J. (2000). "A survey of multiobjective optimization in engineering design". Technical report, Linköping University.
10. Branke, J., & Deb, K. (2005). "Integrating user preferences into evolutionary multi-objective optimization". In Knowledge incorporation in evolutionary computation (pp. 461–477). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-44511-1_21
11. Rivera, G., Florencia, R., Guerrero, M., Porras, R., & Sánchez-Solís, J. P. (2021). Online multi-criteria portfolio analysis through compromise programming models built on the underlying principles of fuzzy outranking. *Information Sciences*, 580, 734–755. https://doi.org/10.1016/j.ins.2021.08.087
12. Rangel-Valdez, N., Fernandez, E., Cruz-Reyes, L., Gomez-Santillan, C., Rivera, G., & Florencia, R. (2018). Robustness analysis of an outranking model parameters' elicitation method in the presence of noisy examples. *Mathematical Problems in Engineering*, 2018. https://doi.org/10.1155/2018/2157937
13. Deb, K., & Kumar, A. (2007, July). "Interactive evolutionary multi-objective optimization and decision-making using reference direction method". In Proceedings of the 9th annual conference on Genetic and evolutionary computation (pp. 781–788). ACM. https://doi.org/10.1145/1276958.1277116
14. Pedro, L. R., & Takahashi, R. H. (2014). INSPM: An interactive evolutionary multi-objective algorithm with preference model. *Information Sciences*, 268, 202-219. https://doi.org/10.1016/j.ins.2013.12.045
15. Vergidis, K. and Tiwari, A. (2008). "The evaluation line: A posteriori preference articulation approach", 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), 2694–2700, https://doi.org/10.1109/CEC.2008.4631160
16. Fernandez, E., Lopez, E., Lopez, F., & Coello, C. A. C. (2011). Increasing selective pressure towards the best compromise in evolutionary multiobjective optimization: The extended NOSGA method. *Information Sciences*, 181(1), 44–56. https://doi.org/10.1016/j.ins.2010.09.007

17. Roy, B. (1996). Multicriteria methodology for decision aiding, volume 12 of *Nonconvex Optimization and Its Applications*, Springer. https://doi.org/10.1007/978-1-4757-2500-1
18. Figueira, J., Mosseau, V., & Roy, B. (2005). Multiple Criteria Decision Analysis: State of the Art Surveys, volume 78 of *International Series on Operations Research & Management Science*, chapter "ELECTRE methods", (pp. 133–153). Springer-Verlag: Berlin. https://doi.org/10.1007/b100605
19. Roy, B. (1990). Readings in Multiple Criteria Decision Aid, chapter "The Outranking Approach and the Foundations of Electre Methods", (pp. 155–183). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-75935-2_8
20. Brans, J. & Mareschal, B. (1990). Readings in Multiple Criteria Decision Aid, chapter "The Promethee Methods for MCDM; The Promcalc, Gaia And Bankadviser Software", (pp. 216–252). Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/978-3-642-75935-2_10
21. Brans JP., Mareschal B. (2005). "Promethee Methods". In: Multiple Criteria Decision Analysis: State of the Art Surveys. International Series in Operations Research & Management Science, vol 78. Springer, New York, NY. https://doi.org/10.1007/0-387-23081-5_5
22. Brans, J. P., Vincke, P., & Mareschal, B. (1986). How to select and how to rank projects: The PROMETHEE method. *European journal of operational research*, 24(2), 228–238. https://doi.org/10.1016/0377-2217(86)90044-5
23. Farina, M., Deb, K., & Amato, P. (2004). Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Transactions on Evolutionary Computation*, 8(5), 425–442. https://doi.org/10.1109/TEVC.2004.831456
24. Baykasoğlu, A., & Ozsoydan, F. B. (2017). Evolutionary and population-based methods versus constructive search strategies in dynamic combinatorial optimization. *Information Sciences*, 420, 159–183. https://doi.org/10.1016/j.ins.2017.08.058
25. Audet, C., Bigeon, J., Cartier, D., Le Digabel, S., & Salomon, L. (2020). Performance indicators in multiobjective optimization. *European Journal of Operational Research* 292(2), 397–422. https://doi.org/10.1016/j.ejor.2020.11.016
26. Goh, C. K., & Tan, K. C. (2009). A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 13(1), 103-127. https://doi.org/10.1109/TEVC.2008.920671
27. Stummer, C., Heidenberger, K. (2003). Interactive R&D portfolio analysis with project interdependencies and time profiles of multiple objectives, *IEEE Trans. Eng. Manage*. 50 (2) 175–183. https://doi.org/10.1109/TEM.2003.810819
28. Kremmel, T., Kubalik, J., Biffl, S. (2011). Software project portfolio optimization with advanced multi-objective evolutionary algorithm, *Appl. Soft Comput*. 11 (1) 1416–1426. https://doi.org/10.1109/TEM.2003.810819
29. Amiri, B. (2012). A multi-objective hybrid optimization algorithm for project selection problem, *J. Basic Appl. Sci. Res*. 2 (7) 6995–7002.
30. Fernandez, E., Gomez, C., Rivera, G., & Cruz-Reyes, L. (2015). Hybrid metaheuristic approach for handling many objectives and decisions on partial support in project portfolio optimisation. *Information Sciences*, 315, 102–122. https://doi.org/10.1016/j.ins.2015.03.064
31. E. Fernandez, E. Lopez, G. Mazcorro, R. Olmedo, C.A. Coello Coello, Application of the non-outranked sorting genetic algorithm to public project portfolio selection, *Inform. Sci*. 228 (2013) 131–149. https://doi.org/10.1016/j.ins.2012.11.018
32. Pisinger, D. (2005). Where are the hard knapsack problems?. *Computers & Operations Research*, 32(9), 2271–2284. https://doi.org/10.1016/j.cor.2004.03.002
33. Moritz, R. L., Reich, E., Bernt, M., & Middendorf, M. (2016, March). "A property preserving method for extending a single-objective problem instance to multiple objectives with specific correlations". In Evolutionary Computation in Combinatorial Optimization (pp. 18–33). Springer, Cham. https://doi.org/10.1007/978-3-319-30698-8_2
34. E. Covantes, E. Fernández, J. Navarro (2016). Handling the Multiplicity of Solutions in a Moea Based PDA-THESEUS Framework for Multi-Criteria Sorting, *Found. Comput. Decis. Sci.* 41: 213–235. https://doi.org/10.1515/fcds-2016-0013
35. Deb, K., Rao N, U., & Karthik, S. (2007). "Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydrothermal power scheduling". In Evolutionary Multi-Criterion Optimization (pp. 803–817). Springer Berlin/Heidelberg. https://doi.org/10.1007/978-3-540-70928-2_60.
36. Nebro, A. J., Luna, F., Alba, E., Dorronsoro, B., Durillo, J. J., & Beham, A. (2008). AbYSS: Adapting scatter search to multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 12(4), 439–457. https://doi.org/10.1109/TEVC.2007.91310
37. Ghahremani, P., & Naderi, B. (2015). Solution algorithms for the project selection and scheduling problem with resource constraints and time dependent returns. *International Journal of Industrial and Systems Engineering*, 19(3), 348–363.
38. Coello, C. C. (2000, July). "Handling preferences in evolutionary multiobjective optimization: A survey". In Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512) (Vol. 1, pp. 30–37). IEEE. https://doi.org/10.1109/CEC.2000.870272