Solving a comprehensive model for multiobjective project portfolio selection

A.F. Carazo\textsuperscript{a,\ast}, Trinidad Gómezb, Julián Molinab, Alfredo G. Hernández-Díaza, Flor M. Guerreroa, Rafael Caballero\textsuperscript{b}

\textsuperscript{a}Departamento de Economía, Métodos Cuantitativos e Historia Económica, Universidad Pablo de Olavide, Carretera Utrera Km. 1, 41013 Sevilla, Spain
\textsuperscript{b}Departamento de Economía Aplicada, Matemáticas, Universidad de Málaga, Campus El Ejido s/n, 29071 Málaga, Spain

\textbf{ARTICLE INFO}

\begin{abstract}
Any organization is routinely faced with the need to make decisions regarding the selection and scheduling of project portfolios from a set of candidate projects. We propose a multiobjective binary programming model that facilitates both obtaining efficient portfolios in line with the set of objectives pursued by the organization, as well as their scheduling regarding the optimum time to launch each project within the portfolio without the need for a priori information on the decision-maker’s preferences. Resource constraints, the possibility of transferring resources not consumed in a given period to the following one, and project interdependence have also been taken into account. Given that the complexity of this problem increases as the number of projects and the number of objectives increase, we solve it using a metaheuristic procedure based on Scatter Search that we call SS-PPS (Scatter Search for Project Portfolio Selection). The characteristics and effectiveness of this method are compared with other heuristic approaches (SPEA and a fully random procedure) using computational experiments on randomly generated instances.

\textbf{Statement of scope and purpose} This paper describes a model to aid in the selection and scheduling of project portfolios within an organization. The model was designed assuming strong interdependence between projects, which therefore have to be assessed in groups, while allowing individual projects to start at different times depending on resource availability or any other strategic or political requirements, which involves timing issues. The simultaneous combination of project portfolio selection and scheduling under general conditions involves known drawbacks that we attempt to remedy. Finally, the model takes into account multiple objectives without requiring a priori specifications regarding the decision-maker’s preferences.

The resolution of the problem was approached using a metaheuristic procedure, which showed by computational experiments good performance compared with other heuristics.

© 2009 Elsevier Ltd. All rights reserved.
\end{abstract}

\textbf{1. Introduction}

Any organization needs to continuously invest in both consecutive and simultaneous projects to guarantee healthy and profitable growth. However, organizations are often confronted with having more projects to choose from than the resources to carry them out and thus one of the main management tasks is to select from an array of projects those better adapted to the organization’s objectives [1]. Wrong decisions in project selection have two negative consequences. On the one hand, resources are spent on unsuitable projects and, on the other hand, the organization loses the benefits it may have gained if these resources had been spent on more suitable projects [2].

In this context, a project is a unique and unrepeatable temporal process having a set of specific objectives. Throughout this work we regard a project as a whole, without taking into consideration that it can be broken down into a set of activities or tasks [3]. In addition, the project cannot be divided in order to execute some parts of it, although different versions of a single project can be addressed, as long as each version is treated as an indivisible proposal.

On the other hand, a project portfolio is a set of projects that share resources during a given period, among which there may be complementarity, incompatibility or synergies produced by sharing costs and benefits derived from conducting more than one project at the same time [4]. This means that it is insufficient to simply compare two projects, but rather we need to compare groups of projects [5] in order to identify the one best adapted to the needs of the organization.

A great variety of methods for project selection exist in the literature [6]. The scoring method [7,8], the multiattribute utility theory
and the analytical hierarchy process [10] are among the most widely used. These models aim at ranking the project set, after which resources are distributed following the priorities established in the ranking. However, this approach assumes that candidate projects are independent, which is not always true, and the interrelationship among them [11,12] means that the best individual projects do not necessarily make the best portfolio [5]. Neither are these methods applicable in situations with multiple constraints (e.g. resource, strategic or political constraints) [13]. These limitations have led to increasing interest in mathematical programming models as they can integrate such considerations into the project portfolio selection process. This interest is supported by advances in the technical procedures used to solve the models generated [14].

Many works emphasize the importance of taking into account the interdependence between projects (see [4,2,15,16,17,12,18]), for a suitable selection of project portfolios. A significant advance in this field was made by Stummer and Heidenberger [19] who designed a more flexible formalization of the interdependencies between any given number of projects. To this end, they introduced an additional term into the corresponding assessment function that activated when the portfolio had at least (or at most) a given number of projects (NP) with a positive (or negative) synergy between them. In the present paper, the proposed model follows this approach in a more general way.

Many other models consider the selection process for a specific period only [15,20,18] or, if they include a planning horizon, it is assumed that all the projects selected start at period one [19]. This can lead to some projects not being implemented because of a lack of resources in a given period, whereas it would be possible to implement them if the model allowed flexibility concerning the moment of starting the projects. The latter option involves simultaneous scheduling and selection, but the greater complexity involved may account for the fact that this approach has been little studied in the literature. The works of Sun and Ma [21], Ghemawat et al. [1] and Medaglia et al. [22] are among the few that study the simultaneous selection and scheduling of project portfolios. Nevertheless, Sun and Ma [21] only deal with one objective, whereas the other two works do not take into account possible synergies between projects, and so the value of the portfolio is obtained by simply summing the value of each individual project.

Within the wide field of multiobjective programming applied to project portfolio selection, some works use goal programming ([20,17,23] among others), where it is assumed that decision-makers are able to set up target values for their objectives, and where information is available regarding their preferences. Other authors, such as Ghemawat et al. [1] and Medaglia et al. [22], integrate the different objectives into a single function by assigning different weighting scores to each objective according to their importance to the decision-maker. Klapp and Piños [18] minimize the distance to the ideal point. Finally, some works do not use a priori information about the decision-maker’s preferences to obtain the set of efficient solutions, but include such preferences at a later stage using interactive techniques [24,19]. Although there are many techniques available to carry out this interactive process, identifying the set of efficient solutions remains a challenge [25].

The way these models have evolved ultimately reflects the attempt to deal with the different aspects involved in project portfolio selection in such a way that the decision-making process gains in rigor and transparency. Such models also need to be flexible enough to be accepted by managers [26]. Our work proposes a nonlinear combinatorial multiobjective model which simultaneously combines the selection and scheduling of project portfolios under general conditions making it applicable to public and private settings. To this end, we have taken into account different types of interactions between candidate projects and the possibility of transferring nonconsumed money resources from a given period to the following one, as well as the availability of resources or other strategic or political requirements in different periods. Furthermore, we introduce multiple objectives without the need for a priori information on the decision-maker’s preferences.

From the mathematical standpoint, the proposed model is an NP-hard problem [27]. Solving such a problem with an exact algorithm is computationally very expensive or even impossible in some cases, which has led to increasing interest in the use of heuristic procedures [28,18,29,25,30]. These procedures offer a good compromise between the quality of the solution obtained and run-time. There are several multiobjective versions of Genetic (GA), Simulated Annealing (SA) and Tabu Search (TS) algorithms [31]. In this work, we develop a metaheuristic algorithm that we call SS-PPS (Scatter Search for Project Portfolio Selection), which is an adaptation of the SSPMO evolutionary method (Scatter Search Procedure for Multiobjective Optimization, Molina et al. [32]) used to determine the set of efficient portfolios for the selection and scheduling of project portfolios.

This paper is structured as follows: we formalize the model proposed in Section 2; in Section 3 we analyze the metaheuristic procedure used to solve it; Section 4 presents the instances and discusses the results; the final section offers the conclusions.

2. Selection and scheduling of project portfolios model

Let us assume an organization with I project proposals from which we have to select the best portfolios according to a set of objectives and some constraints. We also want to specify when each project will start within a given planning horizon (PH) divided into T periods.

Thus, the decision-making variable is denoted by \( x_{it} \) (\( i = 1, \ldots, I; \ t = 1, \ldots, T \)) and is defined by

\[
x_{it} = \begin{cases} 
1 & \text{if project } i \text{ starts at } t \\
0 & \text{otherwise}
\end{cases}
\]

and thus \( x = (x_{11}, \ldots, x_{IT}, x_{21}, \ldots, x_{2T}, \ldots, x_{I1}, \ldots, x_{IT}) \) is a vector with \( T \cdot I \) binary variables which represent one portfolio.

On the other hand, we have to differentiate between the specific period we are in \( (k) \), and the time the selected project \( i \) is within the planning horizon. If project \( i \) starts in \( t \) and lasts \( d_i \) periods, then the execution time of project \( i \) in period \( k \) is \( k+1-t \). If \( k+1-t \leq 0 \), the project has not started yet, and if \( k+1-t > d_i \), the project has already been completed. Thus, project \( i \) will be active in \( k \) only if

\[
\sum_{t=k-d_i+1}^{k} x_{it} = 1
\]

We now describe the objectives used by the organization to select the efficient portfolios\(^1\) followed by the set of constraints that conform the feasible set of portfolios.

2.1. Objective functions

Let us assume that the organization wants to evaluate the portfolios according to a set of attributes (cash-flow, sales, risk, etc.), which, at every period \( k \) of the planning horizon, depend on the specific period each selected project is in [1]. The organization has also specified different subsets of projects \( A_j \) such that if in period \( k \) the portfolio contains a number of projects that is at least equal to \( m_j \) and at most equal to \( M_j \), there is an increase (or decrease) in value \( q_{avg}(\text{synergy } j, q = 1, \ldots, \hat{Q}) \). Therefore, the functions that formalize the attributes under consideration have

\(^1\) A project portfolio is efficient if it is feasible and there is no other feasible portfolio able to improve some of the objectives without making others worse.
the structure shown in expression (3):

$$C_{q,k}(x) = \sum_{i=1}^{T} \sum_{t=1}^{k} c_{i,k+1-t} x_{i} + \sum_{j=1}^{s} g_{jk}(x) a_{jk} \quad q = 1, \ldots, \hat{Q}, \quad k = 1, \ldots, T$$

(3)

where $c_{i,k+1-t}$ represents the individual contribution of project $i$—if it was selected and started at time $t$—to function $q$, in period $k$. Such a project would be at execution time $k+1-t$ in period $k$. In addition, $g_{jk}(x)$ is a function that takes value 1 when synergy $j$ occurs, and 0 otherwise. Thus, the second part of expression (3) represents the effect of positive (or negative) synergies between projects, which is similar to Stummer and Heidenberger’s proposal [19].

On the other hand, the organization may be interested in optimizing the weighted aggregated value of some attributes ($q = \hat{Q}+1, \ldots, Q$) at different periods. In such a case, the functions would be

$$C_{q}(x) = \sum_{k=1}^{T} w_{q,k} C_{q,k}(x) \quad q = \hat{Q} + 1, \ldots, Q$$

(4)

where $w_{q,k}$ is the weight assigned to the $q$th function in period $k$. Furthermore, if some attribute $q^*$ has an economic value $2$ and we want it to be sensitive to the interest rate to reflect different monetary values in each period, then $w_{q,k} = (1 + \text{Rate}_{q}(k))^{(k-1)}$ where $\text{Rate}_{q}(k)$ is the interest rate to be applied to the attribute in period $k$.

Therefore, we have to optimize the following set of functions:

$$C(x) = (C_{q,k}(x), q = 1, \ldots, \hat{Q}, k = 1, 2, \ldots, T)$$

$$C_{q}(x), \quad q = \hat{Q} + 1, \ldots, Q$$

(5)

2.2. Feasible set

The set of constraints can be divided into two large blocks: temporal constraints (affected by time) and, global constraints (independent of time).

2.2.1. Temporal constraints

2.2.1.1. Constraints on available resources such as work force, machine hours, funds and material availability. We assume these resources to be renewable, that is, the availability of each resource is renewed at each period of the planning horizon [33]. Let $r_{i,k+1-t}$ be the amount of type-$u$ resources consumed by project $i$ starting at time $t$. Note that during period $k$ project $i$ is at execution time $k+1-t$. On the other hand, let $a_{j,u,k}$ be the value for the decrease (or increase) of resources due to synergy $j$ between projects within the set $A_j$ (as in the objective functions, but now $j = s+1, \ldots, S$), and let $R_{u,k}$ be the total amount of resources in category $u$ ($u = 1, \ldots, U$) available in the organization for period $k$. Thus, the constraints are

$$\sum_{i=1}^{S} r_{i,k+1-t} x_{i} + \sum_{j=1}^{S} g_{jk}(x) a_{jk} \leq R_{u,k} \quad u = 1, \ldots, U, \quad k = 1, \ldots, T$$

(6)

There are also some resources (e.g. funds) ($u = \hat{U}+1, \ldots, \hat{U}$) which, if not completely consumed in a period, can be transferred to the next period, with the corresponding interest rate ($\text{Rate}_{u}(k)$), and therefore, for $u = \hat{U}+1, \ldots, \hat{U}$; $k = 1, \ldots, T$:

$$\sum_{i=1}^{S} r_{i,k+1-t} x_{i} + \sum_{j=1}^{S} g_{jk}(x) a_{jk} \leq (R_{u,k-1} + \text{Rate}_{u}(k) R_{u,k-1})$$

(7)

In expressions (6) and (7), $g_{jk}(j = s+1, \ldots, S)$ is a binary variable that plays the same role as in expression (3), but that is now associated with synergy $j = s+1, \ldots, S$. If funds are simply passed to next period (without capitalizing), we can also use this formulation just fixing $\text{Rate}_{u}(k) = 0$.

2.2.1.2. Constraints reflecting synergies between projects. Each synergy $j$ ($j = 1, \ldots, S$) has an associated set of projects $A_j$, a lower ($m_j$) and upper ($M_j$) bound, and a corresponding modification in attribute $a_{j,u,k}$ and/or resource $a_{j,u,k}$. Similarly, the function $g_{jk}(x)$ has to detect if there is a synergy $j$ between some given projects or not and can be expressed as the product of two functions $g_{jk}(x)$ and $g_{jk}(x)$.

The first function indicates whether the lower bound is effective or not, and the second indicates the effectiveness of the upper bound. Thus, if the number of active projects of set $A_j$ in period $k$ is at least $m_j$, then it should be the case that $g_{jk}^{m}(x) = 1$ ($= 0$, otherwise).

Taking into account (2), we impose the following constraints:

$$\sum_{i=1}^{S} \sum_{j=1}^{S} x_{i} - m_{j} + 1 \leq g_{jk}^{m}(x) \leq \sum_{i=1}^{S} \sum_{j=1}^{S} x_{i} - m_{j} + 1 \quad (j = 1, 2, \ldots, S)$$

(8)

On the other hand, if the number of projects does not exceed the upper bound $M_j$, then $g_{jk}^{M}(x) = 1$ ($= 0$, otherwise).

Thus, we impose the following constraints:

$$M_j - \sum_{i=1}^{S} \sum_{j=1}^{S} x_{i} + 1 \leq g_{jk}^{M}(x) \leq M_j - \sum_{i=1}^{S} \sum_{j=1}^{S} x_{i} + 1 \quad (j = 1, 2, \ldots, S)$$

(9)

If one of the two bounds is not verified, then $g_{jk}^{m}(x) \cdot g_{jk}^{M}(x) = 0 = g_{jk}(x)$, which means that no synergy $j$ exists, and therefore, the corresponding objective function and/or the resource constraint are not affected by it. In other case, the interaction is produced and it adds to the expressions (3), (6) or (7).

2.2.1.3. Linear constraints. This set includes the limitations the organization wants to impose on the active projects in each period $k$, but which do not depend on their execution time (timing), but rather on being active or inactive in $k$. Let us assume that there are $\eta$ constraints of this kind:

$$b^{\text{inv}}(x) \leq B(k) \leq b^{\text{upp}}(x) \quad k = 1, \ldots, T$$

(10)

where $B(k)$ is a coefficient matrix (order $\eta \times I$), and $b^{\text{inv}}(x)$ and $b^{\text{upp}}(x)$ are the respective upper and lower bound vectors for period $k$.

2.2.2. Global constraints on the projects

(a) Each selected project can only start once within the planning horizon. Furthermore, some projects have to necessarily be part of any feasible portfolio. Therefore,

$$CL \leq \sum_{t=1}^{T} x_{i} \leq 1 \quad \forall i \in I$$

(11)

where $CL$ is a lower bound equal to one or zero depending on whether the $i$th project is mandatory or not.

(b) Bounds are set for the starting time of given projects belonging to the $E$ subset of the total project set:

$$\sum_{t=1}^{T} x_{i} \leq \sum_{t=1}^{T} \delta x_{i} \leq \beta \quad \forall i \in E$$

(12)
Given these constraints, we can formalize the conditions stating that some projects have to be completed within the planning horizon, or that given projects, if selected, should start within a set time-frame:

(c) Global linear constraints.

These are analogous to expression (10), but in this case the conditions imposed on the portfolio projects do not depend on period k. For example, in this group we can specify that different versions of the same project cannot be part of the same portfolio. Therefore, if we assume that there are µ constraints of this kind, then

\[ b_{\text{lower}} \leq B \cdot \begin{pmatrix} \sum_{t=1}^{T} x_{lt} \\ \cdots \\ \sum_{t=1}^{T} x_{lt} \end{pmatrix} \leq b_{\text{upper}} \]  

(13)

where \( B \) is a matrix of order \( \mu \times l \) and \( b_{\text{lower}} \) and \( b_{\text{upper}} \) are vectors for the upper and lower bounds. Within this type of constraints we can represent also non-renewable resources consumption, this is, situations where availability of a resource does not depend on the active projects on a given time period, but only on the projects finally selected for the whole planning horizon.

(d) Precedence constraints between projects

A project cannot be selected if its precursor has not been selected:

\[ \sum_{t=1}^{T} x_{lt} \geq \sum_{t=1}^{T} x_{lt} , i \in P_{l} \]  

(14)

where \( P_{l} \) is the set of precursor projects for a given project \( l \), \( l = 1, \ldots, L \).

A project cannot begin until at least \( h_{l} \) periods and at most \( H_{l} \) periods have passed since the start of its precursors:

\[ \sum_{t=1}^{T} x_{lt} \left( \sum_{t=1}^{T} d_{lt} + h_{l} \right) \leq \sum_{t=1}^{T} d_{lt} \leq \sum_{t=1}^{T} d_{lt} + H_{l} , \forall l \in P_{l} \]  

(15)

Note that if projects \( i (i \in P_{l}) \) and \( l \) are included in the portfolio, then (15) implies the classical formulation for precedence relations with time windows (see [33]):

\[ \left( \sum_{t=1}^{T} d_{lt} + h_{l} \right) \leq \sum_{t=1}^{T} d_{lt} \leq \sum_{t=1}^{T} d_{lt} + H_{l} \]

But if project \( i \) is included in the portfolio but project \( l \) is not \( (\sum_{t=1}^{T} x_{lt} = 0 \text{ and } \sum_{t=1}^{T} d_{lt} = 0) \), then our formulation does not imply infeasibility as we will have \( 0 \leq 0 \leq \sum_{t=1}^{T} d_{lt} + H_{l} , i \in P_{l} \), which is always true. On the other hand, classical formulation, as included in Demelemeester and Herroelen [33], will imply infeasibility for points that are not really infeasible in our case, as it does not take into account if a project is selected or not. This is, if a project is not selected then we must not worry about its precedence relations.

Therefore, the model we are dealing with is a multiobjective problem with binary variables and a nonlinear structure. Using an exact method for its resolution becomes harder as the number of projects and the scheduling horizons increase. Thus, we have approached it using a metaheuristic procedure that we describe and empirically validate in the following section.

3. Scatter Search for Project Portfolio Selection (SS-PPS)

Multiobjective optimization is one of the research areas in which the use of metaheuristic algorithms is becoming more popular. The majority of exact techniques are designed for continuous and discrete linear problems, but when the problem involves a considerable degree of difficulty (such as nonlinear functions or constraints) the resolution can be very expensive or even unfeasible. However, these types of difficulties are very common when faced with the real-life problems of Multiobjective Programming (MP). In recent years, this has led to great progress in the application of metaheuristic techniques to such problems, as reported by Jones et al. [34], Ehrgott and Gandibleux [31], and Coello and Lamont [35], among others.

Evolutionary methods are the most widely applied techniques and predominate in research into metaheuristics for MP. A full review of these methods can be found in Coello et al. [36]. One of the most recognized methods is SPEA2 [37] which is the reference method used in this work.

As mentioned in the introduction, we propose an adaptation of the evolutionary method—Scatter Search Procedure for Multiobjective Optimization [32] to solve selection and scheduling problems in project portfolios. Our method, the Scatter Search for Project Portfolio Selection has two stages:

Stage I: Generating an initial set of efficient points using Tabu Search.

Stage II: Improving this initial set using Scatter Search (SS).

In the SSPMO approach, a core element is Tabu Search understood as a local search able to manage memory in order to make the search more efficient [38].

Thus, in SS-PPS (Stage I), a series of Tabu Search iterations are performed where the initial point in each search is the last point visited by the previous one. In each iteration, we verify whether the current point is dominated (according to Pareto order) by some elements in its neighbourhood. Otherwise, the point is taken as a possibly efficient and it is sent to the set of efficient points to verify whether it is truly efficient or not. If it is efficient, we update the list of efficient points.

By continuously updating possible efficient points, our aim is to obtain a list of every possible efficient point visited by each Tabu Search. Different areas of the decision space can be accessed via Tabu Search, thus enabling a far more effective exploration of the efficient set, as each point visited can become part of the final approximation obtained.

For each search, SS-PPS uses a neighbourhood system at two levels, one related to the inclusion/noninclusion of a given project in the portfolio, and another related to the time of starting the project. That is, for each solution, the neighbourhood includes solutions using the same project portfolio but with different starting times, as well as solutions using different project portfolios.

The second stage is based on the use of a Reference Set (RS), which is at the core of any Scatter Search method. Bearing in mind that we are dealing with a multiobjective problem, the quality of a given solution is measured against \( Q \) objectives and diversity is measured in the objective space. The candidate points to be included in the RS are all the efficient points obtained up to that moment and we denote this efficient set by \( \hat{E} \). Thus, the RS will be made up of \( b \) points \( (b > Q) \) chosen as follows:

1. Construct a restricted candidate list (RCL) from the set \( \hat{E} \).
2. Select the best points for each objective \( Q \) from this list, and include them in RS. Add these points to TabuRS (this list includes the points that cannot be included in the RS set again).
3. Sequentially select from RCL those \( b-Q \) solutions that maximize the distance to TabuRS. Add these points to TabuRS.

The RCL is built in the following way: for each point \( x \) in \( \hat{E} / \text{TabuRS} \), we calculate its maximum distance (normalized in \( (0,1) \), NMD_\( x \)) to the TabuRS list. This value is used as the probability of point \( x \) being
included in RCL. Then a random number is generated, and if this is lower than NMD, point \( x \) is included in RCL. In other words, the greater the distance of a point to TabuRS, the greater is its probability of being included in RCL. In this way, the TabuRS list is used to avoid selecting in the future solutions that are close to previously selected solutions, thus preserving diversity in RS, and promoting diversity in the approximation of set \( E \). In addition, given that solutions at point 3 are sequentially selected, the distances are updated after each selection.

The combination of solutions in RS is carried out in the following way. Each possible pair of solutions in the RS is used to produce 4 offsprings, each of which randomly obtains from their parents full information regarding each single project (i.e. whether it is in the portfolio or not, and its starting time). Thus, once a pair of RS solutions (from the parents) is selected, a random number is generated for each offspring and project \( i \). If the number is smaller than or equal to 0.5, the offspring receives project \( i \) (complete) from one parent, and if otherwise, receives it from the other parent. If the solution is unfeasible, it is randomly rebuilt.

Once new solutions are generated, they undergo an improvement method. To this end, we again resort to Tabu Search, starting from each \( x^k \) point generated. The objective function that guides each of these searches corresponds to a minimizing distance \( L_{\infty} \) (with unitary weights) to the ideal point \((x^*, x^*)\). Each component in this point contains the best values (in the space of objectives) reached by its parents, \( x^* \) and \( x^* \). If we call the \( k \)th component of this ideal point \( \text{Ideal}^{(k)} \), then (assuming maximization for all objectives)

\[
\text{Ideal}^{(k)} = \max(f_k(x^*), f_k(x^*))
\]

This type of search attempts to fill the “gap” existing on the front between points \( x^* \) and \( x^* \).

Once all the solution pairs are combined and all the new solutions improved (and, therefore, set \( E \) possibly includes new elements), a new RS is created for the next iteration. This process is repeated until the average value of NMDs probabilities used to calculate the RCL falls below a pre-set level called DistMed. The following summarizes, in pseudo-code, the steps involved in Stage II of the algorithm:

Parameters (DistMed, b):

1. Build the RCL list. Calculate the average NMD values. If this average is less than DistMed, end. Otherwise, go to step 2.
2. Select the RCL best solutions for each \( Q \) objective and add them to RS. Also add these points to TabuRS.
3. Sequentially select from RCL the \( b-Q \) solutions that maximize the distance to TabuRS and add them to RS. Also add these points to TabuRS.
4. Combine each possible pair of solutions in RS, such that four new solutions from each combination are obtained.
5. Improve each solution using Tabu Search guided by a compromise function.

4. Computational experiments

This section presents a set of computational experiments designed to test the performance of SS-PPS.

To this end, a large battery of 52,272 instances was generated, of which 760 were solved. These attempted to address all the possible scenarios that arise in real life when selecting and scheduling project portfolios. Of these, 523 included maximizing objective functions only, and the remaining 237 included both maximization and minimization objective functions. To make the resolution times comparable, all instances were solved using the same type of personal computer (Pentium 4, 2.99 GHz), which itself is an indicator of its ease of use.

A set of parameters or characteristics were taken into account when generating the instances:

- **Number of projects (NP):** For instances which involved maximizing all their objectives, instances with 10, 15, 20, 25, 30, 40, 50, 60, 70, 80 and 90 candidate projects were solved. When both maximizing and minimizing functions were involved, instances with 10, 15, 20, 25 and 30 projects were solved.
- **Number of objectives (NO):** We worked with 2, 4 and 6 objectives. For instances with maximizing and minimizing functions, all criteria but one were set to be maximized.
- **Scheduling planning horizon (PH):** This parameter was set to take as value 4, 6 or 8 time-periods.
- **Average duration of the candidate projects (AD):** This was calculated in relation to the PH of the instance. In general, this parameter can take the following values: 0.25, 0.5 or 0.75. Thus, if \( PH = 8 \) and \( AD = 0.25 \), then projects would have a duration of around two periods (0.25 x 8 = 2).
- **Resources (R):** This parameter represents the percentage of projects that on average should be active in every period. The level or amount of resources available in every period is estimated based on this parameter which can take values 0.25, 0.5 or 0.75.
- **At Least (AL):** This parameter measures the difficulty of satisfying the lower bounds of constraints (10) and (13) in other words, it measures the minimum percentage of candidate projects that should be active in order to fulfil the linear temporal and linear global constraints. In the generated instances, this parameter can take values 0.1, 0.2 and 0.3. Higher values were not considered because very few of the instances generated with values higher than 0.3 were feasible.
- **As Much (AM):** This measures the difficulty of satisfying the upper bounds of the linear temporal and linear global constraints (10) and (13), that is, the maximum average percentage of active projects for the portfolio to be feasible. This parameter can take values 0.25, 0.5 or 0.75.
- **Correlation coefficient between objective functions (CC):** This parameter measures the level of conflict between different criteria. It can take values 0.2, 0.4, 0.6 or 0.8.
- **Scheduling planning horizon (PH):** This parameter was set to take as value 4, 6 or 8 time-periods.

The remaining characteristics of the problem, such as precedence relationships, number of synergies between projects, etc., were established according to the size of the generated instances (number of projects, number of objectives and scheduling horizon).

Several analyses were performed with the generated instances. First, 760 instances were solved with SS-PPS, and their characteristics, results and run-time analyzed. The results obtained were then compared with those obtained using a reference multiobjective metaheuristic, that is, SPEA2, and a fully random procedure (called Naive) to identify the advantages and disadvantages of our procedure compared with the other two methods.

4.1. Instance resolution results

Once the 760 instances were solved using the SS-PPS metaheuristic, we analyzed its performance on the eight parameters previously mentioned. We used two indicators to evaluate how well this kind of algorithm works: run-time and the number of efficient solutions found for each instance. We calculated possible linear correlations between each parameter and the indicators and found that, in general, the value of these correlations is very small. This means that the metaheuristic is stable in relation to changes in the characteristics of the problem, and is capable of dealing with very different types of problems. At a 1% significance level, and thus reliable, the highest correlations were found between the NP parameter and run-time (0.372), NP and the number of efficient points (0.421), the NO parameter and run-time (0.139), and NO and the number of efficient points (0.242). Figs. 1 and 2 show these relationships in detail.
Fig. 1 was calculated on the basis of the results obtained for 523 instances with only maximizing criteria. Note that each point in the graph is measured by three values: run-time, number of candidate projects and number of objectives for each instance.

The $x$-axis represents 523 instances (problems solved). The order is as follows: first, instances with 10 candidate projects and 2 objective functions are presented, then 10 projects and 4 objective functions, then 10 projects and 6 objective functions, then 20 projects and 2 objectives functions, and so on, until concluding with instances with 90 projects and 6 objectives.

The $y$-axis represents a scale from 0 to 1 and measures the normalized value of each variable under analysis, that is, the number of projects, the number of objectives and run-time. The red line in Fig. 1 measures NO. If the line is at 0 we are dealing with a 2—objective instance, at 0.5 the instance has 4 objectives, and at 1, 6 objectives. The orange line measures NP, such that the smallest number of instances are found at 0 (10 candidate projects), the average number of instances at 0.5 (50 candidate projects), and the greatest number at 1 (90 candidate projects). In terms of the run-time variable, the shorter the time needed to solve the problem, the closer the normalized value is to 0, and the greater the time, the closer the normalized value is to one (shown on the blue line). Finally, the broken vertical grey line represents the boundaries to separate different numbers of objectives within each group of problems with the same number of projects.
It is worth noting that the number of problems solved is not the same for each number of projects and group of objectives. On the one hand, we have solved more problems with 10 candidate projects because they closely resemble real-life problems. On the other hand, execution time was restricted to 3 days, aborting any problem not solved by then and not taking into account the data obtained from such problems. This meant that the number of large instances (70–90 projects with 6 objectives) fully solved was smaller than the number of any other size instance (10–60 candidate projects).

Fig. 2 is similar to Fig. 1, but the number of efficient solutions found is compared with NO and NP. Fig. 2 shows that as the number of projects and objectives increases, so does the number of efficient points found, and this is the main cause of the increase in run-time.

For instances which did not aim at maximizing all the objectives, the results are similar to those previously mentioned.

4.2. Comparison with other algorithms

The SS-PPS metaheuristic was then compared with another metaheuristic widely validated in the literature, SPEA2 [37], and with a fully random algorithm that we call Naive. Two different aspects were compared: the number of solutions found and the run-time of the generated instances.

To carry out these comparisons, 76 random instances of different sizes were solved. They included between 10 and 60 candidate projects with maximizing objective functions, and different values in the other parameters (number of objectives, planning horizon, etc.).

As we faced problems that included different numbers of objectives (2, 4 and 6), it seemed appropriate to use an indicator to evaluate all the instances in the same way. We chose the first objective function optimal value obtained for each procedure. The results obtained are shown in Fig. 3, where the red line represents the value of the first objective function \( f_1 \) obtained by SPEA2, the orange line represents the value obtained from Naive, and the black line the value obtained by SS-PPS. Finally, the vertical broken line delimits the six groups (10–15–20–25–30–60 candidate projects) the instances have been divided into.

The graph shows that for very small problems (10 candidate projects) the values obtained for \( f_1 \) are almost identical when using SPEA2 and SS-PPS and slightly worse when using Naive. This is due to the fact that the size of the feasible set is reduced in small problems and so exploring it is not such a complex task even when using a random method. As the size increases (especially for problems with 25–60 projects) the results of SS-PPS are better than those obtained by SPEA2. The results from Naive indicate fairly stable behaviour throughout the graph, regardless of the size of the problems. Naive yields acceptable results for small problems, but for larger problems the results are very different to those obtained by SPEA2 and SS-PPS. This means as the complexity of the problem increases, the more efficient the two metaheuristics become compared with a basic random search.

Fig. 3 shows that SS-PPS has higher performance than SPEA2, due to the fact our algorithm is better designed to make intelligent use of the information provided by the problem.

To explore this issue further, we solved these 523 instances with the Naive algorithm and compared the results to those obtained by SS-PPS. To clarify the results, we only reported those problems where the results of Naive were better than those from SS-PPS for a given objective. In these cases, we calculated the normalized aggregate deviation of the best values obtained with Naive for each objective and the best values obtained by SS-PPS for each objective. The results are shown in Fig. 4, where 0 was assigned to those instances where SS-PPS provides better results than those using Naive.

In this graph, the blue line shows that the closer the Naive value is to 1, the better the result of this heuristic compared with SS-PPS, and the closer the value is to 0, the better the result using SS-PPS. This graph suggests that Naive provides reasonable results for small problems but, as the size increases, it is never better than SS-PPS.
On the other hand, we compared the results of SS-PPS and SPEA2 taking into account not only the optimal value of the first objective function but also the number of efficient points found for the 76 instances solved. These results are shown in Fig. 5. The y-axis shows the number of efficient solutions found for each problem.

Fig. 5 shows that for small-sized problems, the number of solutions found by SPEA2 and SS-PPS is very similar, but as the size of the problem increases, the differences become increasingly accentuated. Taken together, Figs. 3 and 5 corroborate that SS-PPS performs better in relation to the number and quality of solutions found. Such improvements in quality are obtained at the expense of greater run-time when solving larger problems, as inferred from Fig. 6.

This graph has a very similar structure to the previous ones. The x-axis represents the number of projects and the vertical broken lines separate groups of projects according to size. The y-axis indicates the average time the algorithm takes to find a solution (measured in seconds). The red and black lines represent the same parameter for SPEA2 and SS-PPS, respectively. As the size of the problem increases, the run-time becomes greater for SS-PPS. This is mainly due to the fact that the stopping condition for SS-PPS is adjusted to the...
quality of the approximation obtained, and thus, as the difficulty of the problem increases, the metaheuristic intensifies the search. On the other hand, adjustment is linear using SPEA2 and so run-time increases linearly as the difficulty of the problem increases, thus leading to rapid deterioration in quality as the size and the difficulty of the problem increase.

5. Conclusions

This work proposes a model for the project portfolio selection problem where we simultaneously tackle two different problems: how to select and schedule (choosing the starting point in time) efficient project portfolios. These two processes have to be considered interdependent if we want to distribute the resources better over the planning horizon. Our model includes some key elements required to make a suitable selection: multiple objectives which are often in conflict with each other; uneven availability and consumption of resources; the possibility of transferring un-consumed resources from one period to the following; and complementarity, incompatibility, synergy and precedence relationships between projects.

From a mathematical standpoint, the proposed model is a combinatorial nonlinear multiobjective problem, with increasing complexity as the number of candidate projects and objectives becomes larger. We designed a metaheuristic procedure rather than an exact technique for its resolution. We adapted the SSPMO evolutionary method, based on scattered search, and called it SS-PPS. According to the computational experiments performed, the proposed metaheuristic is stable given changes in the characteristics of the problem, and the results obtained are better than those using SPEA2. Furthermore, it performs better than a basic random search due to making the best use of the information relating to the problem.

Acknowledgments

The authors wish to express their gratitude to the referees for their valuable and helpful comments, which have contributed to improve the quality of the paper. This research has been partially founded by the research projects of Andalusian Regional Government and Spanish Ministry of Education y Ciencia.

References


