Rivera, Gilberto; Gómez, Claudia; Cruz, Laura; García, Rogelio; Balderas, Fausto A.; Fernández, Eduardo R.; López, Fernando
Solution to the Social Portfolio Problem by Evolutionary Algorithms
International Journal of Combinatorial Optimization Problems and Informatics
Morelos, México

Available in: http://www.redalyc.org/articulo.oa?id=265224454004
Solution to the Social Portfolio Problem by Evolutionary Algorithms

Gilberto Rivera, Claudia Gómez, Laura Cruz, Rogelio García, Fausto A. Balderas
Instituto Tecnológico de Cd. Madero, M éxico.
riveragil@gmail.com, cggs71@hotmail.com, lcruzreyes@prodigy.net.mx,
rgarciardz@hotmail.com, tono.21@gmail.com

Eduardo R. Fernández
Universidad Autónoma de Sinaloa, M éxico.
eddyf@uas.uasnet.mx

Fernando López
Universidad Autónoma de Nuevo León, M éxico.
flopez65@gmail.com

Abstract. The formation of project portfolio is a multi-objective problem that has a high impact on public and private organizations, and has generally been addressed by evolutionary algorithms. They often seek an approximation of the Pareto front, and then the decision maker must choose an only solution from the set. This is not a difficult task when you have to select a solution from a small set evaluated in two or three objectives. But when the set of solutions grows, or the number of objectives increases, the choice is often a complicated process. It is necessary to present the decision maker only the subset of the Pareto front according to your preferences. This paper describes an optimization algorithm that steers the search process towards such solutions. The performance of the algorithm is evaluated with respect to the most related algorithm found in the state of the art.

Keywords: Project portfolio, evolutionary algorithm, preferences, decision maker, multi-objective optimization.

1 Introduction

Organizations to ensure their growth and permanence invest continuously and simultaneously on projects, however, confront the problem of having more projects than resources to implement them. One of the main tasks of the manager is to select projects that best meet the objectives of the company [1]. Incorrect decisions regarding the selection of projects have two main consequences:

1. Resources, generally limited, are wasted on projects that, although they may be good, not the most appropriate for the company.
2. The organization loses the benefits that could have gotten if it had invested in more suitable projects.

The selection of projects for a portfolio of social projects needs special treatment for the following reasons [2]:

1. The quality of projects is generally described by multiple criteria that are often in conflict.
2. Typically, requirements are not accurately known. Many concepts have no mathematical support for having entirely subjective nature.
3. Heterogeneity, or differences between the objectives of the projects, makes it difficult to compare.

These features of the Social Portfolio Problem represent a challenge for multi-objective optimization algorithms [3]. Moreover, although optimal solutions are found, the problem has not been completely resolved. Even the decision maker has the task of implementing just one of the alternatives presented. The decision maker will evaluate the alternatives according to her/his criteria and preferences.
2 Background

This section presents the theoretical foundations on which this paper is based.

2.1 Portfolio Problem

A project is a temporary, unique and unrepeatable process which pursues a specific set of objectives [4]. In this work, it is not considered that the projects can be broken down into smaller units such as tasks or activities. In other words, a project cannot be divided to run only a part, however, different versions of the same project can be proposed, each version may vary in amount of activity, time required and requested resources.

A portfolio consists of a set of projects that can be performed in the same period of time [4]. For this reason, projects in the same portfolio share available resources in the organization. They can complement each other, which is called synergy. Thus, it is not sufficient to compare the projects individually, but must compare groups of projects to identify what portfolio makes the greatest contribution to the organization objectives.

The proper selection of projects to integrate the portfolio, which will receive the organization's resources, is one of the most important decision problems for both public and private institutions [5, 6]. The main economic and mathematical models to the portfolio problem assume that there is a defined set of \(n\) projects, each project well characterized with costs and revenues, of which the distribution over time is known. The Decision Maker (DM) is responsible for selecting the portfolio that the company will implement [2].

2.2 Social projects

Social projects are characterized by objectives whose fulfillment benefits society. These objectives are generally intangible, such as social and scientific impact, as well as human resource training, among others, without regard to potential economic benefit as the main element of measure. In addition, the amount of desired objectives in these projects can be of several tens, depending on the level of detail and the conditions under which it is restricted.

It is also important to note that such projects are usually assigned to one area and region. The project area is mainly the social sector, e.g. education, public health, safety, scientific development, among others. The region is primarily concerned with the physical area that will benefit, for example by state, county, district, or similar. Thus, to form social portfolios should be considered:

1. No area/region monopolizes most of the budget, leaving remaining areas/regions with poor resources.
2. All areas/regions receive at least a minimal budget, ensuring its permanence and growth.

2.3 Multi-objective optimization

Real-world optimization problems are extremely complex with many attributes to evaluate and multiple objectives to optimize [7, 8]. The attributes correspond to quantitative values that describe the problem and are expressed in terms of decision variables. The objectives are the directions for improvement of the attributes and can be to maximize or minimize.

In many cases, due to the conflicting nature of attributes is not possible to obtain a single solution and therefore the ideal solution for a multi-objective problem (MOP) cannot be achieved because there is no one solution the problem. Typically, solving a MOP has a set of solutions that reached an aspiration level expected by the DM [7].

Figure 1 graphically illustrates the space of feasible solutions to a maximization problem (Figure 1a) and to a minimization problem (Figure 1b), with two objectives both. Note in Figure 1 that \(p_4\) can improve their performance in both objectives without leaving the feasible solution space, so \(p_4\) is not an optimal solution (it can still be improved). It can also see that \(p_1\), \(p_2\) and \(p_3\) cannot improve one objective without harming the other. Therefore they are solutions that, although different, their performance is mathematically equivalent and cannot be overcome on both objectives simultaneously without leaving the feasible solution space [9]. This set of solutions is called the Pareto front, and find it is one of the main purposes of solving a MOP [10].
But finding the Pareto front does not completely solve a MOP. Now the DM should choose a solution from the front, according to his/her own criteria. This is not a difficult task if you are managing two or three objectives. However, when the number of objectives increases, three major difficulties arise:

1. The capacity of algorithms for finding the Pareto front is rapidly degraded [11].
2. It becomes extremely difficult for the DM, and even impossible, to establish valid criteria for comparing solutions when there are conflicting objectives [9].
3. The size of the Pareto front can grow exponentially with respect to the number of objectives. This complicates the task of the DM to choose a solution [9].

A compromise solution is understood as a Pareto solution in which the objectives achieved acceptable values for the DM, and therefore could be selected. The best compromise is the compromise solution that meets best the preferences of the DM. Thus, the solution to a MOP is not only finding the Pareto front, but also to identify the best compromise.

Identify the Pareto front (or at least an approximation) has been commonly the task of multi-objective algorithms, leaving the identification of the best compromise to the user. However, a typical DM is capable of processing only at most five to ten pieces of information at once [12], thus being unable to identify the best compromise when he/she needs to compare sets of solutions of a MOP over five or nine objectives. To address this problem requires the creation of algorithms for MOP that show a set of solutions as small as possible, but without discarding those that the DM could choose as a final solution.

Since all Pareto solutions are mathematically equivalent, the DM should provide information about his/her preferences to MOP algorithms. Such information can be provided before or after to generate the Pareto solutions, or the process can be interactive, progressively consulting DM preferences [3].

2.4 Formal definition for multi-objective problem

Fernandez [3, 9] proposed a model to describe a multi-objective problem (to identify the best compromise). The model is based on solving Equation 1.

\[
x^* = \min_{x \in O} \{ |S(O, x)| \} \tag{1}
\]

where \(x^*\) is the best compromise, \(O\) is the solution space, \(S(O, x)\) is a function that returns the set of solutions that exceed on preference to \(x\), which can be defined as in Equation 2.

\[
S(O, x) = \{ y \in O \mid yP(\lambda, \beta)x \} \tag{2}
\]
where $y \P(\lambda, \beta)x$ is a relational operator, called the operator of outranking, which indicates that the solution $y$ is more preferable than solution $x$. It reads like "$y$ outranks $x" and indicates that $y$ is at least as good as $x$. An outranking relation $y \P(\lambda, \beta)x$ can be justified if at least one of the following conditions is true:

1. $x$ domines $y$.
2. $\sigma(x, y) \geq \lambda \land \sigma(y, x) \leq 0.5$.
3. $\sigma(x, y) \geq \lambda \land (0.5 \leq \sigma(y, x) \leq \lambda) \land (\sigma(x, y) - \sigma(y, x)) \geq \beta$.

The function $\sigma(x, y)$ returns a value in the range $[0, 1]$, indicating how much the DM is agree with the statement "$x$ is at least as good as $y"$. This value can be calculated using some proven methods such as ELECTRE-III [13, 14] and PROMETHEE [15].

The parameter $0 \leq \lambda \leq 1$ is a threshold that indicates what the minimum magnitude of $\sigma(x, y)$ to consider credible the statement "$x$ is at least as good as $y"$. In general, it is considered that $\lambda \geq 0.5$. The parameter $\beta$ is a threshold that indicates what should be the minimum difference between $\sigma(x, y)$ and $\sigma(y, x)$ for believing that one of them is significantly higher than the other.

The operator of outranking $x \P(\lambda, \beta)y$ serves to calculate, using $S(O, x)$, the number of preferred solutions with respect to a given solution. When $S(O, x) = \emptyset$, means that no exist, or not found (in the case of approximate algorithms), no other solution that is preferred over $x$.

The set of solutions did not outranked in $O$ is a subset of the Pareto front. For a solution $x \in O$, the outranking set is defined as $S(O, x) = \{y \in O, \ y \P(\lambda, \beta)x\}$. The cardinal of this set, $|S(O, x)|$, is an entire function which depends on $x$, and if it is equal to zero, $x$ should be presented as a solution that is probably the best compromise.

Figure 2 illustrates how the search process is addressed when incorporating information about the preferences of decision maker. As shown in Figure 2b, the search is directed to privileged regions of the Pareto front, those where $|S(O, x)| = 0$. These regions contain the solutions that were not outranked by any other solution in $O$. The set of all solutions contained in these regions is called non-outranked front. Figure 2a illustrates how a multi-objective algorithm without a preferential model is not directed toward the non-outranked front, but towards the entire Pareto front.

2.5 Multi-Objective Evolutionary Algorithms

Multi-Objective Evolutionary Algorithms (MOEA) have become a popular technique for solving multi-objective problems [7]. The MOEA are very attractive for solving multi-objective problems because they treat simultaneously a set of possible solutions, which allows obtaining an approximation of the Pareto front in a single execution. Thus, using MOEA, the DM does not need to do a series of optimizations for each objective, as is usually done in the methods of operations research [16, 17].
However, a limitation on the MOEA is the fact that only involves the process of finding a solution set without considering the most important aspect, the decision process. Most current approaches to MOEA are focused on finding an approximation to the optimal set of Pareto front, however, identify the best compromise has usually been omitted.

2.6 Hyper-heuristics

The term hyper-heuristic was first used in 1997 by Jorg Denzinger, Matthias Fuchs and Marc Fuchs [18]. They used the term to describe a protocol that selects and combines various methods of artificial intelligence. Later, in 2000, Cowling and Soubeiga used the term hyper-heuristic to describe the idea of a "heuristic that selects heuristics" in the context of combinatorial optimization [19].

The hyper-heuristics are simple heuristics or high level, that given a particular problem and a number of Low Level Heuristics (LLH) that act on the problem, select and apply the most appropriate LLH in each decision point. The way why the solution space is managed by Hyper-Heuristics is shown in Figure 3, where we can see that a hyper-heuristic runs a series of LLH, which modify the decision variables of an optimization problem, causing a change in the function objective. In the example in Figure 3, the problem has two decision variables ($x_1$ and $x_2$) and three objective functions ($f_1$, $f_2$ and $f_3$).

Fig. 3. Space where hyper-heuristics operate

Usually, meta-heuristics work directly with the solution of the problem by modifying the solution directly, while hyper-heuristics modify the solution indirectly using the available LLH. The hyper-heuristics have a level of abstraction and generality higher than most current meta-heuristics. It's called High Level Heuristics (HLH) to heuristics that controls and directs the HBLLH, and can range from simple search heuristics to more complex meta-heuristics, such as a genetic algorithm [20].

2.7 Hyper-heuristics taxonomy

Burke [21] proposes two criteria to classify hyper-heuristics: Depending on the feedback and depending to the nature of the search space. These taxonomies are shown in Figure 4. According to feedback, the hyper-heuristics may be:

1. Without learning: do not use of feedback during the search process.
2. With off-line learning: learn, from a training instance set, a method that could be generalized to unseen instances.
3. With online learning: learn while solving an instance of the problem.

Depending on the nature of the search space, the hyper-heuristics include: selection heuristic and generation heuristic. The first type has a set of heuristics that solve fully or partially the central problem. The aim is to discover the sequence of how to apply these heuristics to solve the problem efficiently. In the second category are the heuristics that generate heuristics. This kind of hyper-heuristics has a set of constructive heuristics that solve the central problem. First starts with an empty solution and selects intelligently the constructive heuristics for improving gradually the solution. This process continues until to have a complete solution to the central problem.
3 State of the art

SS-PPS [4], Scatter Search for Project Portfolio Selection, is a scatter search algorithm that addresses the portfolio problem. In [4] not only is proposed the algorithm, but also a mathematical model for portfolio selection problem. SS-PPS has basically two stages, the first one generates a set of initial points efficiently using tabu search, and the second phase improves the initial set by scatter search. The algorithm is tested on 76 instances randomly generated, showing a better performance than SPEA2 [21], a multi-objective highlighted algorithm in the literature in MOP. Each instance contains between 10 and 60 candidate projects evaluated in two, four and six objectives, depending on the instance. SS-PPS searches a solution set as scattered as possible in the Pareto front, leaving the final selection of the portfolio to DM.

Doerner et al [22] proposed P-ACO (Pareto Ant Colony Optimization), an algorithm based on the known ant colony meta-heuristic for generating the Pareto front for the most efficient portfolios. Each ant in the colony generates a candidate portfolio, and the amount of pheromone deposited by the ant is inversely proportional to the number of solutions that dominate it. The algorithm stores the solutions that have never been dominated, which form an approximation of the Pareto front. This algorithm is tested on 30 test instances evaluated with nine restrictions. Each instance has between 20 and 30 projects evaluated in 4, 6, 8 or 10 objectives (depending on the instance type). P-ACO during the search does not incorporate the preferences of the DM, so that the final selection of the portfolio depends on the effort that the DM spent for finding a solution according to his preferences in the approximation of Pareto.

Reiter [23] proposes APS (Adaptive Pareto Sampling), an algorithm that uses a sampling approach for estimation of the Pareto front, based on Monte Carlo simulation method. APS is quick to reduce the computational effort without losing accuracy. It does this through a metric that guides the simulation process called Importance Sampling. Reiter compares APS with P-ACO over one hundred test instances, exceeding the performance and runtime. Each project is evaluated on two objectives only. Although APS does not search the non-outranking front, includes techniques to facilitate the dispersion of the solutions in the Pareto front.

NOSGA [9], unlike the works mentioned above, is designed for solving the SPP model formulated in Section 2.4 by Fernandez [3.9]. NOSGA (Non-Outranked-Sorting Genetic Algorithm) is based on the well-known algorithm NSGAII [24], the main change being the addition of the outranking operator in order to obtain the best solutions according to the preferences of DM, and not only the Pareto front as NSGA-II does it.

Unlike NSGA-II, NOSGA is not interested in finding a uniform Pareto front, but looks the non-outranking front. It is noteworthy that NOSGA largely outperforms NSGA-II, according to experimental evidence, for the SPP on a set of instances of three and nine objectives. The main differences between NOSGA and NSGA-II are:

1. The use of $\sigma$ in NOSGA as fitness of individuals, instead of the values in the objective functions.
2. The order based on dominance of the Pareto front is replaced by the strict preference provided by the outranking operator. Thus, zero-order individuals (that have not been outranked by any other solution found) have priority in the crossover and survival to the next generation.

Table 1 presents a comparison among the works presented in this section, where it is seen that NOSGA is the research work with greater affinity to proposal of this article. This is seen in the dimensionality (number of objectives) and that NOSGA takes into account the preferences of the DM during the search for the best portfolio. For this reason, NOSGA is considered as the algorithm which will contrast the performance of the technique presented in this document.
Table 1. Related works to multi-objective portfolio problem

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Basic technique</th>
<th>Objectives</th>
<th>Consider the DM’s preferences</th>
</tr>
</thead>
<tbody>
<tr>
<td>SS-PPS</td>
<td>Scatter search and tabu search</td>
<td>2, 4 and 6</td>
<td></td>
</tr>
<tr>
<td>P-ACO</td>
<td>Ant colony optimization</td>
<td>4, 6, 8 y 10</td>
<td></td>
</tr>
<tr>
<td>APS</td>
<td>Monte Carlo simulation</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>NOSGA</td>
<td>Genetic algorithm</td>
<td>6 and 9</td>
<td></td>
</tr>
</tbody>
</table>

4 Proposed algorithm

This section describes in detail the hyper-heuristic algorithm used for SPP developed by Garcia [6], called Hyper-Heuristic Genetic Algorithm for Social Portfolio Problem (HHGÃ-SPP).

In HHGÃ-SPP, the chromosome of individuals represents the sequence of how LLH are executed. The LLH will interact with the solution of the problem by adding, removing or exchanging projects. None of these low-level heuristics can solve the SPP itself. The genetic algorithm never directly manipulates the solutions even if individuals have in their associated solutions to the problem. The LLH are responsible for modifying the SPP solutions to generate new solutions.

4.1 Solution representation

For the SPP, the representation of the solution is a binary array of size $n$ where each cell represents a project, assigning 0 if the project does not receive support and 1 if the project receives support. Figure 5 shows an example of the representation of the solution for an instance of SPP with 8 projects. In the example in Figure 5, projects 1, 3, 4, 5 and 8 will be supported.

![Fig. 5. Representation of a SPP solution](image)

4.2 Low Level Heuristics

The LLH for SPP were a total of seven:
1. Random change: Change a project for another at random.
2. Generate random solution: Generate a new random solution.
3. Left change: Change a randomly-selected project by the first project that makes the solution feasible, the projects are selected from left to right.
4. Right change: Change a randomly-selected project by the first project that makes the solution feasible, the projects are selected from right to left.
5. Change $x$ region: Take at random one project from each region and replaces it with another project in the same region.
6. Change $x$ area: Take at random one project from each area and replaces it with another project in the same area.
7. Opposite change: Select at random four projects, if a project is in the portfolio is deleted, otherwise it will be chosen.

4.3 High Level Heuristics

The genetic algorithm used as basis for creating the Hyper-Heuristic is the most basic of genetic algorithms. The number of generations is a static parameter, individuals always cross at a cut point and the selection of individuals for the crossover is at random. The mutation rate of HHGÃ-SPP is 25% and the mutation is to exchange one LLH for another randomly selected. In addition, an elitism of 10% of the population is incorporated.
Each individual in the genetic algorithm represents a sequence of LLH. Figure 6 illustrates a chromosome with eight genes of an individual in HHGA-SPP. In each gene is stored the number corresponding to a LLH (see low-level heuristics in Section 4.3) and each individual is associated with the resulting SPP solution by applying the corresponding sequence of HBN.

![Representation of a chromosome in HHGA-SPP](image)

5 Experimental results

This section describes the experiments carried out to evaluate the performance of the proposed algorithm.

5.1 Experimental environment

The following configuration corresponds to the experimental conditions that are common to the tests described in this chapter:
1. Software. Operating system, Linux Ubuntu; programming language, C; compiler, CGG 4.4.4.
2. Hardware. Computer equipment dual-processor Xeon (TM) CPU 3.06 GHz in parallel and 4 GB RAM.
3. Instances. The 30 instances used for this study are randomly generated, consisting of 20 projects evaluated in four objectives.
4. Performance measurement. Performance is measured according to the number of Non-Outranked Solutions (NOS) found by the algorithm.
5. Parameter setting. The parameter setting used to evaluate the performance of the algorithm is that proposed by García [6].

5.2. Algorithm performance

The purpose of this section is to verify the quality of the solutions obtained by HHGA-SPP. A comprehensive search algorithm was run to find optimal solutions for each instance. The results of HHGA-SPP are plotted in Figure 7, where averages of 30 runs of algorithm are showed. In this experiment, a standard deviation of 0.77 among the solutions was observed.

Figure 7 shows that the larger is the optimal set of NOS in the instance, the harder is to find it by HHGA-SPP. The Hyper-heuristic found the optimal for 13 of 30 instances, showing an average error of 26% compared to optimal solutions.

5.3. Performance comparison with related work

The purpose of this section is to verify that HHGA-SPP has a competitive performance with respect to the algorithm NOSGA. The results are plotted in Figure 8, in which the order of appearance of instances is increasing with respect to the optimal number of NOS.

As shown in Figure 8, both algorithms achieve the same performance in 15 test instances, in 4 instances HHGA outperforms NOSGA a 20% on average performance, and in 11 instances NOSGA outperforms HHSAGA-SPP a 17%.
6 Conclusions and future work

This article provides a solution to the Social Portfolio Problem by Evolutionary Algorithms, creating an hyper-heuristic algorithm called HHGA-SPP, that includes the model of Fernandez et al [3, 9], through which DM’s preferences are modeled during the search process.

The results presented show that HHGA-SPP has a competitive performance. In the first experiment HHGA-SPP was compared with an exhaustive search, achieving an average error of 26% compared to optimal solutions. Next was compared with NOSGA, achieving an average error of 3%.

The following future works could be addressed in further study of the topic:
1. Include new LLH that enhance the intensification and diversification capabilities.
2. Analyze the algorithm performance on real instances.
3. Include SPP variants that did not considered in this work.
References


