Scaling up a Project Portfolio Selection Technique by using Multiobjective Genetic Optimization

Márcio de O. Barros[1], Hélio R. Costa[2], Fábio V. Figueiredo[1], Ana Regina C. da Rocha[2]

[1]Post-graduate Information Systems Program – PPGI/UNIRIO
Av. Pasteur 458, Urca – Rio de Janeiro, RJ – Brazil

[2]COPPE / UFRJ – System Engineering and Computer Science Department
Caixa Postal: 68511 – CEP: 21945-970 – Rio de Janeiro – Brazil.

{marcio.barros,fabio.vitorino}@uniriotec.br, heliorcosta@gmail.com,
darocha@centroin.com.br

ABSTRACT. This paper proposes a multiobjective heuristic search approach to support a project portfolio selection technique on scenarios with a large number of candidate projects. The original formulation for the technique requires analyzing all combinations of the candidate projects, which turns to be unfeasible when more than a few alternatives are available. We have used a multiobjective genetic algorithm to partially explore the search space of project combinations and select the most effective ones. We present an experimental study based on four real-world project selection problems that compares the results found by the genetic algorithm to those yielded by a non-systematic search procedure (random search). A second experimental study evaluates the best parameter settings to perform the heuristic search. Experimental results show evidence that the project selection technique can be used in large-scale scenarios and that the genetic algorithm presents better results than simpler search strategies.

KEYWORDS: Multiobjective optimization; portfolio selection; risk; dependency.

1. INTRODUCTION

Project Portfolio Management has gained attention in recent years, as organizations became increasingly project-, program-, and portfolio-oriented [3]. The limited resources available in organizations do not allow executing every project that may be presented for its executives. Thus, it is necessary to establish a procedure to select a subset of those candidate projects that can be executed within the available resources while maximizing profits and minimizing portfolio risk. Levine [4] defines project portfolio management as the administration of a company's portfolio, aiming to maximize the contribution of the projects under execution to the overall welfare and success of the company. Cooper et al. [5] outline the major goals for portfolio management as maximizing portfolio value, selecting the right projects to comprise the portfolio, and linking the portfolio to the organization's business strategy.

Selecting the projects which will be executed by the company is a major component of portfolio management processes. Project selection aims to define an optimal (or close to optimal) subset of projects to comprise the company's portfolio, taking into account their characteristics and relationships among them [8]. Suboptimal portfolios may include projects that do not contribute to the company's strategic or tactic goals, consuming effort, money, and time that could be dedicated for more productive actions. Many project selection techniques can be found in the literature [3] [5] [7], but few directly address an aspect that becomes important if these projects are to be executed together, instead of as separately managed efforts: the dependencies among candidate projects.

Recently, Costa et al. [2] presented a project selection technique based on Modern Portfolio Theory [6]. The technique evaluates all portfolios which can be formed by combining a set of candidate projects, introduces a systematic procedure to calculate the dependencies among them, estimates the risks of all portfolios prone to be selected, and generates a return x risk indicator for each portfolio. It was evaluated through a set of experimental studies involving decision-makers from the industry and results show indications that taking project dependencies into account tends to support better decisions while selecting project portfolios.

On the other hand, the computational cost of executing the technique is a power function of both the number of candidate projects and the number of independent risks that may affect these projects. The high cost is due to analyzing all combinations of the candidate projects and prevents using the technique in large-scale scenarios, with more than a few candidate projects. For instance, the technique evaluates 32 portfolios in a scenario with five candidate projects, but if there are 40 available projects the number of combinations surpasses a trillion possibilities. In such a scenario, which is common for large companies, the technique cannot be executed in a feasible timeframe.

In this paper, we present a multiobjective heuristic optimization approach to support the application of the technique proposed by Costa et al. [2] in large-scale scenarios on regard of the number of candidate projects available to comprise the portfolio. We present a formal representation for the project selection problem and use a bi-objective genetic algorithm to find effective portfolios in terms of their risk x return profiles without examining all possible combinations of the available projects. The optimization approach was evaluated using four project selection problems made available by a large Brazilian company. Experimental results show that the multiobjective heuristic search can find good portfolios in feasible time and finds better results than simpler search procedures, such as Random Search.

Our primary contributions are as follows: (i) a multiobjective heuristic optimization approach to support the application of the project portfolio selection technique in scenarios with a large number of candidate projects; and (ii) experimental studies to determine the most appropriate parameter settings for the proposed multiobjective heuristic search and to compare it with a simpler, non-systematic search procedure.

Besides this introduction, this paper is organized in six sections. Next, we present the Modern Portfolio Theory, which provides the theoretical basis for the project selection technique supported by the search approach proposed in this paper (multiobjective genetic algorithms). The technique itself is presented in Section 3. In Section 4 we describe the multiobjective heuristic search approach that was used to support the portfolio selection technique. Experimental studies that were designed and executed to evaluate the heuristic search approach are presented in Section 5. Section 6 presents related work, while future works and conclusions are drawn in Section 7.

2. MODERN PORTFOLIO THEORY

Modern Portfolio Theory (MPT) is a disciplined procedure to support the allocation of capital in investment portfolios comprised of financial assets [6]. Under this theory, a portfolio is a weighted combination of assets, the weight of each asset being proportional to the amount of capital invested in it. MPT suggests how much of the available capital an investor should allocate to each asset to maximize the expected return and minimize the risk incurred by the portfolio. It requires calculating the return and risk of each possible portfolio which can be built from the available assets. Next, the portfolios are depicted in a scatter plot chart presenting portfolio risk (σ_P) on the horizontal axis and expected return (ER_P) on the vertical axis (Figure 1).

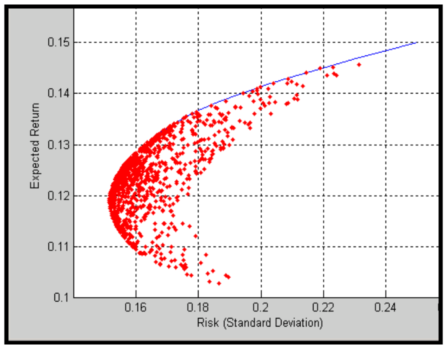

Figure 1. A typical Efficient Frontier

The Efficient Frontier, formed by the uppermost points set forth in the chart, presents all portfolios with maximum expected return for a given level of risk. A typical frontier is presented in Figure 1. Given how much risk the investor is willing to accept, the frontier shows the portfolio with the greatest expected return. On another perspective, it shows the portfolio with minimum risk for a given expected return. Thus, the Efficient Frontier comprises all projects that maximize the risk x return ratio.

The Expected Return yielded by a portfolio (ER_P) is represented by the weighted sum of the expected returns of its assets. For a portfolio consisting of *m* assets, ER_P can be calculated by equation (1), where w_i is the percentage of capital invested in asset *i* and μ_i is the expected return of the same asset.

$$ER_P = \sum_{i=1}^{m} w_i . \mu_i, \text{ where } \sum_{i=1}^{m} w_i = 100\% \qquad (1)$$

Portfolio risk (σ_P) is a function of the independent risks of its assets (σ_i), the proportion of capital invested in each asset, and the correlation (ρ_{ij}) among them. The risk of a given asset is usually estimated by the standard deviation of its observed returns over time. The correlation is a measure of the dependence between a pair of assets, indicating the strength and direction of the relationship between them. It is represented by a number in the [-1, +1] interval, where -1 represents two assets moving in opposite directions with similar strength while +1 represents two assets that tend to move in the same direction with similar strength. Correlation 0 (zero) means that no relation between the two assets can be inferred from the history of their observed returns. Optimal portfolios usually embed combinations of negatively-correlated assets, resulting in less risky portfolios since a negative impact on an asset is compensated by a positive impact on another one. Given the weights, the correlation, and the risks of its assets, the risk of a portfolio entailing *m* assets is calculated by equation (2).

$$\sigma_P = \sqrt{\sum_{i=1}^{m} w_i^2 . \sigma_i^2 + 2 . \sum_{i=1}^{m} \sum_{j=1, j \neq i}^{m} w_i . w_j . \sigma_i . \sigma_j . \rho_{ij}} \qquad (2)$$

When MPT is used to support project selection, two restrictions need to be considered. First, the proportion of capital invested on each asset in the financial market is an investor's discretion and can be changed at any time by converting assets to money (selling) or money to assets (buying). In a project portfolio setting, the proportion of capital invested in each project is dictated by the resources required to conduct the project and once a company is committed to a project such resources usually cannot be used for other purposes. Moreover, a project cannot be partially taken: it is either selected to comprise the company's portfolio or discarded.

Second, investors trading financial assets usually have data about these assets' performance in the past and can use this information to estimate risk and (with restrictions) expected returns. Projects are unique by definition and therefore there is no information available about their former performance. Thus, risk and return must be estimated according to expectations regarding their future cash flows and influences from uncertain factors upon them (opportunities to be explored and risks to be faced or countered). This is the basis for the project selection technique presented in the next section.

3. A PROJECT PORTFOLIO SELECTION TECHNIQUE

Costa et al. [2] present a technique to select projects to build a portfolio based on concepts underlying MPT and restrictions that must be taken into account when applying the theory to a project selection context. The technique depends on the following information to characterize candidate projects and risks that may affect them. Let P be the set of candidate projects, with $|P| \geq 1$ elements. Each project $p_i \in P$ is characterized by its development cost ($cost_i$) and the net present value of its expected cash flows (pv_i). Let R be the set of risks affecting the candidate projects, with $|R| \geq 1$ elements. Each risk $r_j \in R$ is described by its probability of occurrence ($prob_j$) and expected impact upon each project ($impact_{i,j}$). The impact of a risk upon a project may be positive (if the risk represents an opportunity) or negative (if the risk is a threat for the project). Risks affecting more than one project are especially important because they allow observing how these projects behave when exposed to the same uncertainties, providing the basis to measure dependency (correlation) among them. In software projects, examples of risks that may affect more than a single project include creeping user requirements, implementation of new technologies, human resources issues, support from senior management, and low productivity.

Based on the former information, the project selection technique creates all alternative portfolios that can be formed by combining subsets of the candidate projects and whose cost is under a limit established by the company (the amount of capital available for investments). The magnitude of the number of alternative portfolios is 2^m, being *m* the number of candidate projects. Next, all possible risk scenarios that may affect the portfolios are created by combining subsets of formerly identified risks. These scenarios can vary from the occurrence of no risk to all risks occurring simultaneously. Given *n* risks, the total number of scenarios is 2^n. Each scenario is characterized by its probability of occurrence and its impact upon each project. The probability of a given scenario S is calculated by multiplying the probability of occurrence of all risks participating in the scenario, times one minus the probability of all other risks (equation 3). The impact of a scenario S upon a project p_i is the sum of the impacts of all risks comprising the scenario upon that project (equation 4).

$$prob(S) = \prod_{r_j \in S} prob_j \, . \prod_{r_j \notin S} (1 - prob_j) \qquad (3)$$

$$impact(S, p_i) = \sum_{r_j \in S} impact_{i,j} \qquad (4)$$

The technique follows by calculating risk-adjusted project data. At the financial market, the inputs for MPT are the historical time series of observed returns over time for each asset. Risk (standard deviation), expected return (mean), and correlations among assets can be calculated from these series, allowing for the computation of portfolio risk and expected return by means of equations (1) and (2). The observed return time series for an asset is formed due to the passage of time and the changing perceptions of market agents (banks, investors, and companies) regarding the asset's future price. Since projects are unique, historical time series on their returns do not exist and other means must be sought to estimate project risk, return, and correlation. The project selection technique suggests analyzing the frequency of occurrence of risk scenarios and their impact upon the return of candidate projects. The

Expected Return (ER_i) of a candidate project is calculated as the weighted average return that each risk scenario yields for the project, where weights are given by the scenario's probability of occurrence. Similarly, the risk of each project (σ_i) is calculated by the weighted standard deviation of the return yielded for the project on each scenario. Finally, the correlation ($\rho_{i,j}$) between two projects is calculated using the Spearman Rank Order Coefficient upon their pair-wise weighted returns for the same scenarios.

Next, risk-adjusted project information can be aggregated at the portfolio level. The cost of a portfolio PT (C_{PT}) is calculated by summing up the development cost of each project comprising the portfolio. The expected return of a portfolio (ER_{PT}) is calculated by summing up the expected return of each project comprising the portfolio. Portfolio risk (σ_{PT}) is calculated by equation (5), which is derived from equation (2) and takes into account only those *n* projects comprising the portfolio. Weights were removed from the equation because projects taking part on the portfolio have weight equal to 1 (they cannot be partially undertaken), while other projects have weight equal to zero.

$$\sigma_{PT} = \sqrt{\sum_{i=1}^{n} \sigma_i^2 + 2.\sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \sigma_i.\sigma_j.\rho_{ij}} \quad (5)$$

Finally, the technique creates the portfolio chart (such as Figure 1) from the set of pairs (ER_{PT}, σ_{PT}) for alternative portfolios whose cost is below the investment budget and depicts the Efficient Frontier. Thus, it shows the decision-maker which portfolios represent the highest return for a given risk or the lowest risk for a given return. This limits the choices of the decision-maker, since choosing portfolios which are not part of the frontier is not a rational, optimal decision according to MPT tenets.

The project selection technique considers only two variables: risk and return. Despite the importance of these variables, the decision of which portfolio will be undertaken by the company may be influenced by other factors, like risk appetite, the company´s strategic goals, development cost, or type of cash flow that a company seeks to develop and/or expend. External factors, such as legal constraints and political moves, may also influence the decision, but they are out of the scope of the proposed technique.

4. PROJECT PORTFOLIO SELECTION AS A MULTIOBJECTIVE PROBLEM

Project portfolio selection is a bi-objective problem where two incomparable measures (risk and return) define the most effective portfolios. Risk must be minimized, while expected return must be maximized. Therefore, we are interested in the portfolio which yields maximum return for a given level of risk or, on the opposite perspective, which incurs minimum risk to yield a certain return. The most effective portfolios form a curve disposed in the risk x return plane. A decision about which among these portfolios will be undertaken by the company depends on the decision-makers willingness to accept more risk in exchange for more return.

A bi-objective search to select the most effective portfolios must look for the Pareto-optimal set of subsets of P maximizing return and minimizing risk. Under Pareto optimality, one solution is better than another if it improves at least one of the individual objectives and does not decrease the remaining ones [9] [12]. These are known as *non-dominated solutions*, since no solution in the Pareto-optimal set can be said better than any other solution in the same set for all required objectives. Therefore, a bi-objective search algorithm supporting the technique presented in Section 3 yields a set of Pareto-optimal solutions PT*, each representing a portfolio comprised of projects pertaining to P.

We have addressed the optimization problem using the NSGA-II algorithm [1]. NSGA-II is a multiobjective genetic algorithm based on a ranking procedure which classifies candidate solutions according to their dominance. Non-dominated solutions are assigned a rank of 1; solutions dominated only by non-dominated solutions are assigned a rank of 2; solutions dominated only by the former are assigned a rank of 3, and so on. The algorithm evolves a population over a number of generations, applying crossover, mutation, and selection upon candidate solutions. The selection process prioritizes low-ranking solutions and, when a subset of solutions having the same rank must be selected, a density measure allows selecting candidate solutions covering the search space as uniformly as possible.

The NSGA-II algorithm was programmed to maximize returns and minimize risks. We have used the JMetal framework [11] and its implementation of the NSGA-II algorithm in the experiments designed to evaluate the effectiveness of the algorithm while searching solutions for the project selection problem. The crossover operator uses single point crossover with 90% crossover probability. The mutation operator uses uniform mutation with 1% probability. Binary tournament is used as selection strategy. Population size was set as two times the number of projects. The maximum number of fitness function evaluations was set as 100 times the square of the number of projects. Each candidate solution represents a potential portfolio and was encoded as a sequence of bits, one for each available candidate project. The bit for a given project indicates whether the project is part of the portfolio represented in the solution.

5. EVALUATING THE SEARCH-BASED APPROACH FOR PROJECT SELECTION

In this section we present two experimental studies conducted to evaluate the search-based approach to the technique presented in Section 3. First, we present the problem instances selected for the evaluation. Next, we present an experimental analysis designed to find the best parameter settings to run the NSGA-II algorithm. Finally, we present a comparison between the genetic algorithm and a multiobjective random search applied to the same instances.

5.1 Problem Instances

We have analyzed the behavior of the NSGA-II multiobjective genetic algorithm applied to the project selection technique using four real-world instances. These instances were provided by a Brazilian company acting in the distribution of electric energy and depict an excerpt of the candidate projects that were available to form the company's project portfolio for 2011. The instances also conveyed information about the risks that could affect the organization's business goals and the candidate projects. As required by the company that provided the data, we cannot disclose information about the projects and their risks. In fact, even the instances we have received contained obfuscated information about the name of the projects and risks, disclosing only value and cost data required for our computations.

To run the study, 250 projects were selected from the 556 eligible projects that could receive investments from the company. These projects were grouped into four categories, each representing an instance used in the experiment: (a) new buildings, installation and restoration works, consisting of 25 out of the 66 projects identified in this category; (b) maintenance, improvements, and upgrade projects, with 50 out of the 315 projects identified for this category; (c) R&D projects, consisting of 75 projects; and (d) new ventures and investments, with a total of 100 projects.

Projects in categories (a) and (b) were selected according to the date they were registered in the information system that supports the executive board on investment decisions, that is, older projects were chosen up to the desired number to compose each category. The number of projects in these categories was limited to allow experimenting with a set of instances that vary in size, thus depicting how the proposed algorithm behaves in different scenarios. The registration date-based criterion was used to avoid bias (cost, present value, risk exposure) in the selected projects. Instances representing categories (c) and (d) were selected to evaluate whether the optimization process would be able to find good solutions even in large scenarios.

Each category represents a cost center in the company and has a separate investment budget to implement a project portfolio for the period. As we have not selected all projects identified by the organization (556 projects), we have used a proportional amount of the budget available for each category, as follows (actual values cannot be published):

(a) The 25 selected projects are worth of approximately 38% of the cost of running the 66 projects in this category. Therefore, the constrained budget for this group was estimated at 38% of the budget available for projects comprising this category;

(b) The 50 selected projects are worth of 16% of the total cost of running the 315 projects in this category. Similarly to the first group, the constrained budget for this group was fixed at 16% of the available budget;

(c) We have selected all projects from the third category, so the full budget was considered;

(d) We have also selected all projects from the fourth category, so the full budget was considered.

Individual costs and estimated present value for future cash flows to be generated by each candidate project have been provided by employees and consultants working for the company but cannot be disclosed. One hundred and fourteen (114) uncertain events that might affect the business conducted by the organization were identified using questionnaires and interviews with employees and project stakeholders. Such risk identification process was performed before we have requested the data and without the participation of researchers involved in the present work. From these, 106 risks were directly related to the selected candidate projects. The probability of occurrence and the total impact of each risk were also identified. Based on this information, consultants calculated the impact of each individual risk upon each project. Since we were not interested in the effect of an increasing number of risks affecting the projects, we selected the 10 most important risks according to project exposure (that is, the risks with higher exposures) to perform the evaluation.

5.2 Parameter Settings

Parameter settings were configured according to the results of an experimental evaluation which used the instance comprised of 25 projects subjected to 10 risks. The NSGA-II algorithm was executed to find solutions for this instance under several distinct configurations of crossover probability, mutation probability, and population size. Since the budget of fitness function evaluations is calculated according to population size, changing the last parameter also affected the budget available for the algorithm.

Five distinct crossover probabilities were tested (60%, 70%, 80%, 90%, and 100%), along with five distinct mutation probabilities (1%, 2%, 3%, 4%, and 5%) and four population-size factors (50%, 100%, 150%, and 200%). The base population was set as the number of projects in the instance and the population-size factor was applied upon this number, thus testing the effects of halving the population, using the base population size, and increasing the population by 50% and 100%. All combinations of parameters were evaluated to identify the best settings to run the NSGA-II algorithm for the project selection technique. A total of a hundred distinct combinations were tested. To account for the variation inherent in stochastic heuristic algorithm, NSGA-II was executed 30 times for each configuration. Hereafter, we will call each execution of a given configuration a *running cycle*.

Each running cycle for a given configuration yielded a Pareto front comprised of a set of solutions ($PF_{c,m,f,i}$), where *c* represents the crossover probability used in the configuration under analysis, *m* represents the mutation probability for that configuration, *f* represents the population size factor, and *i* represents the cycle number. After running all cycles for a given configuration, a best front for that configuration was built by joining the fronts yielded by each cycle and removing dominated solutions ($PF_{c,m,f}$). Finally, after running all configurations, the best fronts for each configuration were merged to create the best front for the instance (PF_{best}), again removing dominated solutions. Each vertex of every front represents a portfolio and is described by two objectives – expected return and portfolio risk.

We have selected the parameter settings from the configuration whose best front ($PF_{c,m,f}$) was closest to PF_{best}. We have used the generational distance quality indicator to compute the distance between two Pareto fronts. Generational distance represents the distance between $PF_{c,m,f}$ and PF_{best} calculating the Euclidean distance between each solution pertaining to $PF_{c,m,f}$ and the closest solution composing PF_{best}. Lower numbers are preferable, since they indicate that a given Pareto front is closer to the best front. Generational distances are defined in the [0, +∞[interval.

Tables 1 to 4 show means and standard deviations for generational distances collected after running each configuration. Each table represents a given population size factor. Table rows represent values for crossover probability, while table columns represent values for mutation probability. Generational distance values are presented in table cells in 1/1,000 scale (that is, actual values are obtained by dividing values shown in the tables by 1,000).

Table 1 – Generational distances for population factor = 50%

	1%	**2%**	**3%**	**4%**	**5%**
60%	5.96 ± 2.4	6.13 ± 3.1	6.75 ± 3.3	6.33 ± 3.7	6.80 ± 3.6
70%	7.35 ± 4.2	5.70 ±2.7	5.09 ± 2.8	6.47 ± 2.5	6.34 ± 2.2
80%	6.14 ± 2.8	5.64 ± 2.8	6.67 ± 3.0	5.85 ± 2.77	6.75 ± 3.5
90%	5.56 ± 2.2	6.40 ± 3.3	7.25 ± 3.2	5.79 ± 2.6	5.92 ± 3.0
100%	6.58 ± 2.9	6.02 ± 3.3	5.91 ± 2.8	5.26 ± 2.5	5.69 ± 3.0

Table 2 – Generational distances for population factor = 100%

	1%	**2%**	**3%**	**4%**	**5%**
60%	1.39 ± 0.7	1.36 ± 0.6	1.18 ± 0.4	1.37 ± 0.5	1.29 ± 0.5
70%	1.12 ± 0.3	1.24 ± 0.5	1.33 ± 0.5	1.34 ± 0.6	1.35 ± 0.4
80%	1.24 ± 0.4	1.26 ± 0.4	1.23 ± 0.5	1.24 ± 0.4	1.29 ± 0.5
90%	1.20 ± 0.5	1.34 ± 0.6	1.35 ± 0.5	1.26 ± 0.5	1.09 ± 0.4
100%	1.10 ± 0.4	1.29 ± 0.4	1.17 ± 0.4	1.12 ± 0.5	1.14 ± 0.4

Table 3 – Generational distances for population factor = 150%

	1%	**2%**	**3%**	**4%**	**5%**
60%	0.67 ± 0.2	0.70 ± 0.2	0.61 ± 0.1	0.64 ± 0.2	0.62 ± 0.1
70%	0.62 ± 0.1	0.68 ± 0.2	0.55 ± 0.2	0.60 ± 0.1	0.60 ± 0.2
80%	0.62 ± 0.1	0.63 ± 0.1	0.57 ± 0.2	0.63 ± 0.2	0.56 ± 0.1
90%	0.57 ± 0.2	0.67 ± 0.2	0.56 ± 0.2	0.63 ± 0.2	0.51 ± 0.1
100%	0.56 ± 0.2	0.60 ± 0.1	0.66 ± 0.2	0.61 ± 0.2	0.63 ± 0.1

Table 4 – Generational distances for population factor = 200%

	1%	**2%**	**3%**	**4%**	**5%**
60%	0.41 ± 0.11	0.41 ± 0.12	0.42 ± 0.14	0.40 ± 0.12	0.39 ± 0.14
70%	0.37 ± 0.10	0.39 ± 0.12	**0.36 ± 0.12**	0.38 ± 0.08	0.39 ± 0.10
80%	0.38 ± 0.10	**0.35 ± 0.12**	**0.34 ± 0.10**	**0.36 ± 0.13**	0.39 ± 0.10
90%	0.31 ± 0.08	0.38 ± 0.11	0.39 ± 0.12	0.37 ± 0.10	**0.35 ± 0.11**
100%	**0.33 ± 0.08**	**0.33 ± 0.09**	**0.35 ± 0.11**	**0.35 ± 0.09**	**0.35 ± 0.07**

As shown in Tables 1 to 4, the smallest average generational distance was observed under the configuration using 90% crossover probability, 1% mutation probability, and 200% population size factor (hereafter called *base configuration*). The base configuration is represented in the grey cell in Table 4. All values on Tables 1 to 3 are significantly different from the base configuration with at least 95% confidence, according to a non-parametric Wilcoxon-Mann-Whitney statistical test. Bold face values on Table 4 are not significantly different from the base configuration with 95% confidence. The *p-values* yielded by the statistical test while comparing these configurations to the base one, along with effect-sizes, are presented in Table 5.

P-values closer to zero indicate stronger confidence that the results being compared are statistically different. Effect-size measures, such as the non-parametric Vargha and Delaney's A_{12} statistics [13] used in our analysis, assess the magnitude of improvement in a pair-wise comparison. Given a measure M for observations collected after applying treatments A and B, A_{12} measures the probability that treatment A yields higher M values than B. If both treatments are equivalent, then $A_{12} = 0.5$. Otherwise, A_{12} indicates the frequency of improvement, e.g., $A_{12} = 0.7$ denotes that higher results would be obtained 70% of the time with A. In Table 5, an effect-size of 0.38 denotes that the referred configuration will be able to yield smaller generational distances than the base configuration in 38% of its executions, while the base configuration will yield better values 62% of the time. Means, standard deviations, *p-values*, and effect sizes were calculated using the R Statistical Computing system[1] v2.12.2.

Table 5 – P-values and effect-sizes for the selected configurations

	P-Value	**Effect-Size**
PF (70%, 3%, 200%)	0.07	0.38
PF (80%, 2%, 200%)	0.20	0.41
PF (80%, 3%, 200%)	0.22	0.42
PF (80%, 4%, 200%)	0.16	0.40
PF (90%, 5%, 200%)	0.11	0.39
PF (100%, 1%, 200%)	0.59	0.47
PF (100%, 2%, 200%)	0.40	0.44
PF (100%, 3%, 200%)	0.37	0.44
PF (100%, 4%, 200%)	0.11	0.39
PF (100%, 5%, 200%)	0.54	0.37

While some configurations using 80% and 100% crossover probability could also be considered good settings for the search algorithm, effect-size measures add evidence that the base configuration represents the best parameter settings (at least for the instance under analysis). Therefore, this configuration was used in the experiment reported in the next section and is suggested for further applications of the proposed approach.

We can also observe from Tables 1 to 4 that population-size factor seems to be the most important parameter among those selected for the analysis. The percentile difference between the maximum and minimum generational distances for all configurations using the same population-size factor (intra-treatment variation) varies from 28% (Table 2) to 45% (Table 1). On the other hand, the percentile difference between the maximum and minimum overall distances (extra treatment variation) varies up to 2,250%.

[1] http://www.R-project.org

5.3 Comparison with a Simpler Search

To evaluate whether a complex search procedure, such as the NSGA-II algorithm, is required to find good solutions for the project selection problem in scenarios of varying sizes, we designed and executed an experimental study to compare the heuristic search with a simpler, non-systematic search procedure. The study compared the efficiency and effectiveness of both searches using the instances described in Section 5.1.

Two configurations were tested for each instance. The first one, hereafter called GA, used the NSGA-II algorithm with the parameter settings and fitness evaluation budget described in Section 4. The second configuration, hereafter referred to as RS, used a multiobjective random search with the same fitness evaluation budget given to the NSGA-II algorithm.

The multiobjective random search is a random search that uses an archive of non-dominated solutions to build a Pareto front taking into account more than one objective. The algorithm is essentially a loop where a solution is randomly generated in each step and compared to the solutions in the archive for domination. Solutions dominated by the new one are removed from the archive and the new solution is introduced if it is not dominated by any of the former ones. The search procedure continues until it consumes its budget of fitness function evaluations.

To properly account for the randomness inherent in heuristic search procedures, each configuration was executed 30 times for all instances. For each pair of configuration and instance, each running cycle yielded a Pareto front comprised of a finite set of solutions (PF_i). After running all cycles for a given instance and configuration, a best front for that pair was built by joining the fronts yielded by each cycle and removing dominated solutions (PF_{GA} and PF_{RS}). Finally, PF_{GA} and PF_{RS} were merged to create the best front for the instance at hand (PF_{best}), again removing dominated solutions. Each vertex of the Pareto fronts represents a portfolio and is described by two objectives – the expected return and the risk incurred by the portfolio.

To evaluate the efficiency of a configuration, we have collected the execution time for each cycle, configuration, and instance. In this context, execution time means the wall-clock time required to run the cycle. Lower values are preferred, since they indicate that the configuration under analysis consumes less processing power to find solutions for an instance. To evaluate the effectiveness of a configuration, we have collected the generational distance and error ratio for each cycle, configuration, and instance. Generational distance was already introduced in Section 5.2. Error ratio is calculated by one less the count of solutions in PF_i which also pertain to the best front (PF_{best}) divided by the count of solutions in PF_i. Lower numbers are preferred, since they indicate that a cycle's front has more solutions pertaining to the best front. Error ratios are defined in the [0, 1] interval.

After collecting execution time, generational distance, and error ratio data, configurations were compared in a per instance basis, e.g., results yielded by GA for the instance with 25 projects were compared to those presented by RS for the same instance. Smaller execution times for a given configuration indicate that it is more efficient than the other. Smaller error ratios and generational distances for a configuration denote that it yields more effective results than the second one. These values were subjected to a non-parametric Wilcoxon-Mann-Whitney test to ascertain if there was statistically significant difference between the configurations.

The following tables present means and standard deviations of the measures above for each instance/configuration over 30 cycles. They also present the *p-value* for the non-parametric test and the Vargha and Delaney's A_{12} effect-size measure.

Table 6 shows execution times (measured in seconds) collected after performing the experiment. Execution time for configuration GA is on average two times greater than under configuration RS, but this percentile is severely reduced for the largest instance. Nevertheless, NSGA-II consumes much more processing time than the random search to find its solutions. The *p-value* for the statistical test converges to zero for all instances, denoting that differences in execution time are significantly different with, at least, 99% confidence. Effect-size values show that NSGA-II will take more time to find its solutions in 100% of its runs for all but the smallest instance, on which about 6.8% of the random searches consume more processing time to run than the respective genetic algorithm.

Table 6 – Execution time analysis

	GA	RS	P-Value	Effect-Size
25P	1.7 ± 0.05	1.3 ± 0.19	< 0.001	93.2
50P	13.4 ± 0.03	5.2 ± 0.73	< 0.001	100.0
75P	46.1 ± 0.03	9.8 ± 0.16	< 0.001	100.0
100P	134.7 ±0.27	80.0 ± 12.7	< 0.001	100.0

Table 7 shows error ratios collected after running the experiment. As in the former table, it presents means and standard deviations for each instance's error ratio under configurations GA and RS over the 30 cycles, the *p-value* for the statistical test, and effect-size. Error ratio under configuration RS is, on average, 98% greater than under GA. For all but the smallest instance, no cycle running the random search contributed to PF_{best}. Since smaller values are preferred, the genetic algorithm seems to find more effective solutions (in terms of error ratio) than the random search. As in the former table, p-values converge to zero for all instances, denoting that differences in error ratio are statistically significant with at least 99% confidence. Effect-size A_{12} measures also converge to zero, indicating that the genetic algorithm will be able to yield solutions with less error ratio in 100% of its runs.

Table 7 – Error ratio analysis

	GA	RS	P-Value	Effect-Size
25P	0.45 ± 0.07	0.90 ± 0.03	< 0.001	0.0
50P	0.53 ± 0.06	1.0 ± 0.0	< 0.001	0.0
75P	0.41 ± 0.04	1.0 ± 0.0	< 0.001	0.0
100P	0.63 ± 0.05	1.0 ± 0.0	< 0.001	0.0

Table 8 shows generational distances collected after running the experiment. These values are represented in 1/1,000 scale (that is, actual values are obtained by dividing the values in the table by 1,000). Conclusions regarding generational distance are not as straightforward as those drawn for execution time and error ratio. Even with a larger error ratio, random search was able to find solutions with less generational distance than NSGA-II for the smallest instance. This implies that although the solutions found by random search were not in the best front, they were close to it. Lower generational distances, alongside with lower execution times, indicate that random search might be a feasible procedure for solving small instances of the project selection problem.

On the other hand, generational distance for larger instances under configuration RS is, on average, 9,702% greater than under GA. This large difference is due to difficulties in finding good solutions for the instance with 75 projects, which is the hardest instance in terms of available capital to invest in the portfolio. The amount of capital available for this kind of project is only 26% of the total amount required to fund all candidate projects in the instance. Thus, many randomly-generated portfolios were over budget (that is, unfeasible) and were not even tested for dominance. The genetic algorithm seems able to compensate for this restriction, finding significantly better portfolios than random search.

Table 8 – Generational distance analysis

	GA	RS	P-Value	Effect-Size
25P	0.33 ± 0.12	0.11 ± 0.03	0.99	100.0
50P	0.10 ± 0.02	1.00 ± 0.12	< 0.001	0.0
75P	0.13 ± 0.05	43.07 ± 4.33	< 0.001	0.0
100P	0.10 ± 0.02	4.96 ± 0.61	< 0.001	0.0

The *p-value* converges to zero for all instances except for the one with 25 projects, denoting that differences in generational distance are statistically significant with at least 99% confidence (they are significant in all cases, but the smallest instance is better served by the random search). The effect-size A_{12} shows that in 100% of the runs the genetic algorithm will yield solutions with smaller generational distance than the random search, except for the smallest instance in which the reverse is true.

The former data shows strong evidence in favor of the heuristic search, except for small instances with relatively large budgets to fund the project portfolio. The genetic algorithm outperforms the random search in finding solutions closer to the best Pareto front, though the random search was able to find a good approximation of this front for the smallest instance.

Nevertheless, one may argue that a fair comparison between the genetic algorithm and random search would give a much larger fitness evaluation budget to the later, allowing to compare strategies that consume roughly the same amount of resources (in this case, computer processing time). We have repeated the experiment described in the former paragraphs, though giving a fitness evaluation budget 8 times larger to random search (configuration RS8). By being allowed to consume this larger budget, all running cycles for RS8 took more processing time to run than the respective NSGA-II.

We observe improvements for both error ratio and generational distance if RS is compared to RS8. RS8 produced an average error ratio of 0.81 for the smallest instance, while error ratio remained equal to 1.0 for instances with more than 25 projects. On regard of generational distance, the average improvement across all instances was about 50%, topping in 80% for the smallest instance. However, these numbers are still far behind those produced by the genetic algorithm. An exception is generational distance for the smallest instance on which random search finds better results than GA regardless of using a larger budget. Thus, independent of using the same fitness evaluation budget or a similar amount of processing time, random search cannot keep up with results produced by the genetic algorithm.

5.4 Threats to Validity

Wohlin et al [10] classify the threats to validity that may affect an experimental study into four categories: conclusion, construct, internal, and external threats. Barros and Dias-Neto [14] propose an extension of the framework for search-based experiments.

Conclusion threats are concerned with the relationship between the treatment and the outcome. In SBSE experiments, major conclusion threats include not accounting for random variation in the search, lack of good descriptive statistics and of a meaningful comparison baseline. These issues were addressed in this paper by running 30 experimental cycles for each instance/configuration, by presenting means and standard deviation for relevant measures collected after running the experiment, by comparing these values using a non-parametric test, and presenting effect-size measures.

Internal threats evaluate if a relationship between the treatment and the outcome in an experimental study is causal or the result of a factor upon which the researcher has no control. In SBSE experiments, major internal threats involve poor parameterization, lack of real-world problem instances, and not discussing code instrumentation and data collection procedure. These issues were addressed in this paper by presenting an experimental analysis which was conceived to find the most appropriate settings for the heuristic algorithm, by using four real-world instances (kindly provided by a private company), by describing the data collection procedure used to build the instances, and by using a well-known heuristic algorithms library in the implementation.

Construct threats are concerned with the relation between theory and observation, ensuring that the treatment reflects the construct of the cause and that the outcome reflects the construct of the effect. In SBSE experiments, major construct threats involve using invalid efficiency and effectiveness measures and not discussing the underlying model subjected to optimization. On regard of the model, the project selection technique that is supported by the approach proposed in this paper was presented and discussed in Sections 2 and 3. Regarding the measures selected for the experiment, using wall-clock time as efficiency measure is questionable if the experimental cycles are executed in different computers under distinct loads, but we took precautions to run all cycles in the same computer under similar system load. The effectiveness measures, error ratio and generational distance, were selected to allow comparing Pareto fronts in terms of their proximity to the approximated best front for each instance. Since we are interested on the ability of the algorithms to yield solutions closer to the best front, these seem reasonable quality indicators for our experiments.

Finally, external threats are concerned with the generalization of the observed results to a larger population, outside the sample instances used in the experiment. Major external threats to SBSE experiments include lacking a clear definition of target instances, lacking a clear instance selection strategy, and not having enough diversity in instance size and complexity. In our experiment we have used four instances of different sizes (in terms of the number of projects), though complexity was not directly addressed. In the future, we intend to improve the proposed approach to handle a growing number of risks, but this is out of the scope of the present paper. Finally, the instances used in Section 5.1 are protected by a non-disclosure agreement and cannot be made available for replications of the present study.

6. RELATED WORKS

The usage of heuristic search algorithms to support the application of MPT (Modern Portfolio Ttheory) in the context of financial asset portfolios has been addressed by many authors. Schaerf [15] compared three local search strategies (*Hill Climbing*, *Simulated Annealing*, and *Tabu Search*) on their ability to address the asset selection problem given constraints on the number of different assets to take part on the portfolio and on the maximum amount to be invested in any single asset. The author concluded that *Tabu Search* was the most effective alternative to find solutions for the problem. Lai et al. [16] proposed a two-stage asset selection procedure in which a genetic algorithm was used to identify high-quality assets (based on their history regarding four financial indicators) and then a second genetic algorithm is used to select the best combination of the selected assets (based on MPT). Lin and Liu [17] presented a heuristic strategy to address the asset selection problem given minimum lot restrictions, that is, the traded quantities of any asset are restricted to multiples of a given quantity. Bakar et al. [18] evaluated using a genetic algorithm to select portfolios using equities from the two most important economic sectors traded in Malaysia. Chang et al. [19] evaluated a genetic algorithm to select financial portfolios based on three risk measures: semi-variance, absolute standard deviation, and variance with asymmetry.

In relation to project portfolios, a few papers were found to take dependencies into account in the optimization process. However, the interpretation of the concept of dependency varies in each case. Bhattacharyya et al. [20] presented a project selection approach where projects can have dependencies in their outcomes (jointly-executed projects may yield different results than when executed together), techniques (leveraging a given technology used in two or more projects), resources (many projects sharing a limited resource pool), and risks (projects executing at the same time may increase their risk exposure). Fuzzy set theory is used to model dependencies, while both a single-objective and a multi-objective genetic algorithm were evaluated in the optimization (using an instance with only six projects). Fuzzy set theory is also used by Wang and Hwang [21], who proposed a real-option based hedge strategy to reduce the risk of a project portfolio.

Finally, regarding software projects, Kremmel et al. [22] presents a multiobjective genetic algorithm to support selecting software projects. The optimization takes into account potential revenue, strategic alignment, resource usage, risk and synergy. The approach was evaluated through an experimental study that used 50 projects and compared the proposed multiobjective search strategy to NSGA-II and SPEA2.

7. Conclusions

This paper proposed a multiobjective heuristic search approach to support a project portfolio selection technique that, in its original formulation, cannot be executed in feasible time for scenarios with more than 20 candidate projects. The technique is based on concepts of the Modern Portfolio Theory and was formally presented in Section 3. An experimental procedure to find suitable parameter settings for the heuristic algorithm selected to support the technique was designed and presented.

We found evidence that a heuristic search is required for finding proper solutions for large instances or those characterized by a severely constrained investment budget. The NSGA-II algorithm outperformed random search both in terms of error ratio and generational distance effectiveness indicators for large instances. On the other hand, random search seems a feasible alternative for small instances or those with large budgets. Limitations of the present work which can be addressed by future research include adapting the heuristic search to deal with a large number of risks and repeating the experiment with more instances.

ACKNOWLEDGMENTS

The author would like to express his gratitude for FAPERJ and CNPq, the research agencies that supported this project.

REFERENCES

[1] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II", IEEE Transactions on Evolutionary Computing Vol. 6, No. 2, pp 182-197, April 2002.

[2] Costa, H.R., Barros, M.O., Rocha, A.R.C, "Software Project Portfolio Selection: a Modern Portfolio Theory Based Technique", Proceedings of the 22nd Software Engineering and Knowledge Engineering Conference, San Francisco, EUA, 2010.

[3] Killen, C.P., Hunt, R.A., Kleinschmidt, E.J., "Managing the New Product Development Project Portfolio: A Review of the Literature and Empirical Evidence". PICMET Proceedings, Portland, USA, August 2007.

[4] Levine, H.A., *Project portfolio management: A practical guide to selecting projects, managing portfolios, and maximizing benefits*. John Wiley & Sons, San Francisco, USA, 2005.

[5] Cooper, R.G., Edgett, S.J., Kleinschmidt, E.J., *Portfolio Management for New Products*. 2nd edition, Cambridge, MS, USA, 2001.

[6] Markowitz, H.M., "Portfolio Selection". Journal of Finance, Vol. 7, No 1, pp. 77-91, 1952.

[7] Dye, L. D., Pennypacker, J., *Project Portfolio Management: Selecting and Prioritizing Projects for Competitive Advantage*. Glen Mills, PA, 2003

[8] Ding, W., and Cao, R., "Methods for Selecting the Optimal Portfolio of Projects". IEEE International Conference on Service Operations and Logistics, and Informatics, 2008.

[9] M. Harman, S. A. Masouri, Y. Zhang, "Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications", Dept. of Computer Science, King's College London, Technical Report TR-09-03, April 2009.

[10] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, A. Wesslén, *Experimentation in Software Engineering*, Kluwer Academic Publishers, Norwell, MA, 2000

[11] J. J. Durillo, A. J. Nebro, F. Luna, B. Doronsoro, E. Alba, “JMetal: A Java Framework for Developing Multi-objective Optimization Metaheuristics”, TR ITI-2006-10, Dept. de Lenguajes y Ciencias de Computacion, Univ. Málaga, 2006

[12] Harman, M., Tratt, L., “Pareto Optimal Search Based Refactoring at the Design Level”, Proceedings of the GECCO’07, London, UK, 2007

[13] Arcuri, A., Briand, L., “A Practical Guide for Using Statistical Tests to Assess Randomized Algorithms in Software Engineering”, Proc. of the 33th International Conference on Software Engineering, ICSE’11, Hawaii, EUA, 2011

[14] Barros, M.O., Dias-Neto, A.C., “Threats to Validity in Search-based Software Engineering Empirical Studies”, Technical Report DIA/UNIRIO, No. 6, Rio de Janeiro, Brazil, 2011 (available at URL http://www.seer.unirio.br/index.php/monografiasppgi/article/ viewFile/1479/1307)

[15] Schaerf, A., “Local Search Techniques for Constrained Portfolio Selection Problems”. Computational Economics, v. 20, n. 3 (Dec), 2002, pp. 177-190.

[16] Lai, K.K.; Yu, L.; Wang, S.; Zhou, C., “A Double-Stage Genetic Optimization Algorithm for Portfolio Selection”, In Proceedings of the 13th International Conference Neural Information Processing (ICONIP 2006), Part III, v. 4234, 2006, pp. 928 – 937

[17] Lin, C.; Liu, Y., “Genetic Algorithms for Portfolio Selection Problems with Minimum Transaction Lots”, European Journal of Operational Research, v. 185, Issue 1, 2008, pp. 393-404

[18] Bakar, N.A., Adam, N.L., Hassan, S., “GA Based Portfolio Optimization Technique”, in Proceedings of the Conference on Scientific & Social Research (CSSR´09), Melaka, Malysia, 2009

[19] Chang, T.; Yang, S.; Chang, K., “Portfolio optimization problems in different risk measures using genetic algorithm”, Expert Systems with Applications: An International Journal, v. 36, Issue 7, 2009, pp. 10529-10537.

[20] Bhattacharyya, R., Kumarb, P., Kar, S., “Fuzzy R&D portfolio selection of interdependent projects”, Computers & Mathematics with Applications, 62, pp. 3857-3870, 2011

[21] Wang, J., Hwang, W.-L., “A fuzzy set approach for R&D portfolio selection using a real options valuation model”, Omega, 35, pp. 247 – 257, 2007

[22] Kremmel, T., Kubalík, J., Biffl, S., “Software project portfolio optimization with advanced multiobjective evolutionary algorithms”, Applied Soft Computing, Vol 11, 1, pp. 1416-1426, 2011