Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 957108, 8 pages
http://dx.doi.org/10.1155/2014/957108

Research Article

A Bicriteria Approach Identifying Nondominated Portfolios

Javier Pereira,[1] Broderick Crawford,[2,3] Fernando Paredes,[1] and Ricardo Soto[2,4]

[1] *Escuela de Ingeniería Industrial, Universidad Diego Portales, 8370179 Santiago, Chile*
[2] *Pontificia Universidad Católica de Valparaíso, 2362807 Valparaíso, Chile*
[3] *Universidad Finis Terrae, 7500000 Santiago, Chile*
[4] *Universidad Autónoma de Chile, 7500000 Santiago, Chile*

Correspondence should be addressed to Broderick Crawford; broderick.crawford@ucv.cl

Received 27 February 2014; Accepted 18 June 2014; Published 3 July 2014

Academic Editor: Mohammad Khodabakhshi

We explore a portfolio constructive model, formulated in terms of satisfaction of a given set of technical requirements, with the minimum number of projects and minimum redundancy. An algorithm issued from robust portfolio modeling is adapted to a vector model, modifying the dominance condition as convenient, in order to find the set of nondominated portfolios, as solutions of a bicriteria integer linear programming problem. In order to improve the former algorithm, a process finding an optimal solution of a monocriteria version of this problem is proposed, which is further used as a first feasible solution aiding to find nondominated solutions more rapidly. Next, a sorting process is applied on the input data or information matrix, which is intended to prune nonfeasible solutions early in the constructive algorithm. Numerical examples show that the optimization and sorting processes both improve computational efficiency of the original algorithm. Their limits are also shown on certain complex instances.

1. Introduction

Markowitz provided one of the first comprehensive theoretical frameworks for the portfolio selection problem [1]. In his proposal, each portfolio is evaluated in terms of the expected return and risk. Then, the efficient set, or efficient frontier, corresponds to all portfolios with the largest expected return, given a level of risk. From this framework, the expected return is usually evaluated as the weighted sum of the expected return from each project in the portfolio, while the risk value is evaluated by the variance of the portfolio. Thus, the investor may select the portfolios in the efficient frontier that best match her/his needs.

In portfolio selection two vectors are defined [2]. First, the *investment proportion vector* corresponds to the proportion of money that the investor accepts to invest on each member of a set of securities (projects). The *criteria vector*, instead, contains the values of measures evaluating the portfolio. In this sense, an efficient portfolio, in terms of the first vector, is a nondominated portfolio, in the sense of the second one. A multicriteria portfolio selection problem supposes a criteria vector with three or more criteria [3], which is expected to be more difficult in terms of computing of nondominated portfolios. However, below, we show that depending on what it is to be taken into account as an evaluation measure, even with two criteria the generation of portfolios is a hard combinatorial problem.

Since the portfolio selection problem intrinsically incorporates business criteria, budget restrictions, and returns volatility [4], in the literature the problem is formulated as the maximization of the expected return, under uncertainty of returns. When the multicriteria version is considered, several utility functions need to be maximized, subject to constrains defining the feasible portfolios [1, 4, 5]. In this context, nondominated portfolios may be computed using multiobjective algorithms [6], evolutionary methods for multiobjective models [7, 8], or preference programming [9].

In this article, we focus on the portfolio generation process, when business, budget, or even volatility information is poor. Several stringent situations obligate to split the project portfolio selection process into at least two phases: technical and business concerns. We place ourselves in the first phase and formulate our problem in terms of satisfaction of a given set of technical requirements, with the minimum number

of projects and minimum redundancy. This is a particular viewpoint where the selection problem is stated. In fact, as explained below, we are concerned with the generation of interesting portfolios more than with the problem of choosing the best portfolio.

Therefore, we place the problem in a very early phase of nondominated portfolios identification. In order to fix the ideas, a general multicriteria decision aiding (MCDA) framework is chosen [10], centering our attention on the problem formulation stage, which supposes the definition of the set of potential actions (i.e., alternatives, candidates, or decision subjects) [11]. Nevertheless, as given in many practical situations, we assume that actions are not well-defined elements at hand. Indeed, Simon was one of the first scientists proposing that actions do not come perfectly defined and represented in decision processes [12]. Thus, when actions are not given in advance, a searching process must be activated in order to discover or design them. In some situations, except for abstract or very elementary decision processes, a potential action must be constructed from objects available in a repertory of primitive elements. The project portfolio problem parallels this situation very well. Stipulated as a management activity, portfolio selection is the processes of conceiving portfolios from discrete, even interrelated, projects.

We propose an approach to support exploratory search of an analyst, aiding him/her to identify a restricted list of nondominated interesting portfolios from a projects set, satisfying the whole set of requirements. Observing that the most relevant projects in a portfolio do not necessarily correspond to those satisfying the largest number of requirements, but rather to those blending well with other projects, we propose an algorithm that provides a list of all potentially interesting portfolios, based on requirements coverage and minimal collective redundancy. The approach is applicable in situations where detailed business information is poor and nondominated interesting portfolios could be analyzed in further stages of development, when better information is available [13].

This article is organized as follows. In Section 2, the problem stated is modeled as a bicriteria integer linear program and a constructive procedure allowing exploring a set of projects in order to identify nondominated portfolios is defined. A proposition guaranteeing that such procedure converges is proved. Next, an algorithm grounded on the constructive approach is introduced in Section 3. An improvement to this process is proposed, realizing that the approach is sensitive to initial values obtained on one of criteria of the ILP model. In Section 4 our approach is tested and results are compared to those obtained in a previous work. Finally, Section 6 is dedicated to conclusions.

2. Model Formulation

Let us consider a manager involved in a portfolio selection and composition process, looking for different combinations of projects, satisfying some technical requirements. In addition, let us assume that she/he prefers to initially explore a restricted set of projects, only looking for some interesting portfolio alternatives. The small exploration of pieces reveals to her/him possibilities to further consider them with new business criteria (costs, benefits, profit, etc.) or market related features (expansion, collocation, territorial coverage, etc.), among others. Therefore, the problem consists of identifying the set of portfolios satisfying a given number of requirements defined by the analyst, knowing that a systematic exploration might be a very hard task, even not reasonable. In such a context, is it possible to aid the analyst in the construction of early interesting solutions, using limited information?

Methodologically, this is a decision-making activity unfolding the following phases [11].

2.1. Problem Formulation. The problem formulation is a statement defining the triplet $\Gamma = \langle A, V, \Pi \rangle$, where A is a set of actions, in our case portfolios, V is a set of points of view (e.g., dimensions) considered to evaluate elements of A, and Π is a statement defining what would be done with elements of A (selection, ranking, classification, etc.).

2.2. Evaluation Model. Formally, an evaluation model is a tuple $M = \langle A, G, U, R \rangle$, where G is a set of criteria, eventually derived from V, allowing the evaltation of elements of A in terms of each criteria; U models uncertainty regarding available information in $A \times G$; and R is an aggregation logic defining the way that the information concerning A and G is operated in order to obtain a global conclusion solving the problem Π. The evaluation model produces a process output Φ.

2.3. Recommendation. The output of the evaluation model is translated into the decision maker's language, verifying that it is technically sound and deployable in the decision maker's setting and processes.

In the problem formulation phase, A is frequently supposed to be a known fact or the result of a modeling task, as usually done in linear programming, for instance, when a set of feasible alternatives is modeled by linear restrictions. Under such assumption, elements of A become the matter of analysis, evaluation, and recommendation. Then, further phases may be applied in a regular way.

In our case, a portfolio is not known in advance and becomes an alternative once it has been conceived or designed as a project composite. We restrict ourselves to the problem of definition of a set of portfolios, considering that a pool of projects is given *a priori*. Then, a portfolio will be a subset of components covering a set of predefined requirements.

Finding of nondominated portfolios is formulated here as a bicriteria integer linear program. Let us consider

(i) $P = \{p_1, p_2, \ldots, p_n\}$ is the collection of projects;

(ii) $\wp = 2^P$ the set of possible portfolios, $a, b \in \wp$, $f_i : \wp \to R$;

(iii) $A = (a_{ji})_{m \times n}$; matrix of projects and requirements;

(iv) $a_{ji} \in \{0, 1\} : a_{ji} = 1$ if p_j covers the requirement r_i, $a_{ji} = 0$ otherwise.

(v) $x_j \in \{0,1\} : x_j = 1$ if p_j was included in a feasible solution, and $x_j = 0$ if not;

(vi) y_i times ith requirement is covered.

The following program is a model of the problem, where the whole set of nondominated portfolios simultaneously minimize the number of projects in a composite and the portfolio redundancy; that is, the number of times that requirements are of covered by more than one project. Consider

$$\begin{aligned} &\min \quad \sum_i y_i, \\ &\min \quad \sum_j x_j \\ &\text{St.} \quad \sum_j a_{ji} x_j = y_i, \quad \forall i \\ &\qquad y_i \geq 1, \quad \forall i \\ &\qquad x_j \in \{0,1\}, \quad \forall j. \end{aligned} \tag{1}$$

Inspecting this model we observe that taking into account only the objective ($\min \sum_j x_j$), this program corresponds to an instance of the set covering problem, well known to be NP-hard. In consequence, there are instances where this program cannot be solved by a traditional approach as the branch and bound method [14]. Instead, we propose an algorithm based on a constructive procedure adapted from a preference programming approach presented in [9], where an efficient algorithm aiding to find robust nondominated portfolios has been introduced. Such an algorithm was originally presented in the context of imperfect information regarding evaluation of projects on a number of continue value functions and their respective weights. Interestingly, the algorithm was also based on the progressive generation of nondominated portfolios, which could be split in two phases: generation of candidates and pruning.

Proposition 1 below guarantees a procedure for finding every nondominated solution in the Pareto front, grounded on the two-phase approach described.

Proposition 1. *Let $2 \leq k \leq n$, $P = \{p_1, p_2, \ldots, p_n\}$ be a set of projects and $a_k \subset P$ a portfolio having at most k projects. Equally, let $\eta(a_k)$ and $\mathrm{cov}(a_k)$ be defined as the redundancy and coverage levels of a_k, R the set of requirements to be satisfied, and N_n the set of nondominated portfolios 2^P in given the sets C_k, L_{k+1}, N_k, defined as follows:*

$$\begin{aligned} C_k &= \left\{ a_k \in 2^P \mid \eta(a_k) \leq \min_{a \in N_{k-1}} \eta(a) \right\}, \\ L_{k+1} &= \left\{ a_k \in C_k \mid \left| \mathrm{cov}\ (a_k) \right| < |R| \right\}, \\ N_k &= \left\{ a \mid \nexists\, a' \in N_{k-1} \cup (C_k \setminus L_{k+1}) \textit{ such that } a' \succ a \right\}. \end{aligned} \tag{2}$$

Then

$$N_n = \left\{ a \mid \nexists\, a' \in 2^P \textit{ such that } a' \succ a \right\}. \tag{3}$$

(1) $N_0 = \varnothing,\ L_1 = \{\varnothing\},\ n = MM$;
(2) **for** $(k = 1, n, k{+}{+})$ **do**
(3) $\quad C_k = \mathrm{Candidates}(L_k)$
(4) $\quad L_{k+1} = \{a \in C_k \mid |\mathrm{Cov}(a)| < |R|\}$
(5) $\quad N_k = \{a \mid \nexists\, a' \in N_{k-1} \cup (C_k \setminus L_{k+1})$ such that $a' \succ a\}$
(6) $\quad n = \min_{a \in N_k} \eta(a)$
(7) **end for**

Algorithm 1

Proof. Let us proceed by induction. Thus, assume that N_{k-1} contains the whole set of nondominated portfolios bringing together 1, 2, or $k-1$ size projects. By construction, C_k contains the whole set of feasible and unfeasible project portfolios improving, or at least equaling, the minimum redundancy value computed in the $k-1$ stage. Any portfolio worsen this value is not included in this collection. Therefore, $(C_k \setminus L_{k+1}) = \{a \in C_k \mid \mathrm{cov}(a) = |R|\}$. Let M_k be defined as

$$\begin{aligned} M_k = &\left\{ a \in N_{k-1} \mid \nexists\, a' \in (C_k \setminus L_{k+1}), \eta(a) = \eta\left(a'\right) \right\} \\ &\cup \left\{ a \in (C_k \setminus L_{k+1}) \nexists\, a' \in N_{k-1}, \eta(a) \geq \eta\left(a'\right) \right\}. \end{aligned} \tag{4}$$

Then, it is clear that $\forall a' \in (N_{k-1} \cup (C_k \setminus L_{k+1})) \setminus M_k$ and $\forall a \in M_k$, $a \succ a'$. In consequence, $N_k = M_k$ is set of nondominated portfolios discovered until the stage k. In the nth round, N_n will contain the whole set of nondominated portfolios. □

In the next section, we present an algorithm based on Proposition 1. Further, an improvement is introduced noticing that the progressive construction and pruning of project portfolios may be accelerated when a convenient upper bound is set for η and a convenient sorting is applied on the information matrix.

3. Algorithm

The main purpose of the algorithm presented in this section is the identification of the whole set of nondominated portfolios. Different algorithms and approaches have been proposed for this task [14]. However, we focus on a solution provided by [9], which we have adapted as a strategy for the program (1). It is based on the candidates building and pruning processes implementing a constructive generation of project portfolios. In this algorithm, in order to generate candidates potentially selected as feasible solutions, any portfolio having a redundancy greater or equal than the minimal redundancy, found at the current level of the procedure, is pruned. Next, potential feasible solutions are compared to nondominated candidates, which have been found in the precedent iteration. The algorithm is defined in Algorithm 1.

The function *candidates* (Algorithm 2) is the generation module in this approach. It is interesting because we could change it in order to alter the behavior of the algorithm [15]. Notice that an initial redundancy value is set at MM, a number big enough that will be modified the first time

```
(1) L = ∅
(2) for a ∈ C_k do
(3)    a = a ∪ {p_k}
(4)    if η(a) ≤ η then
(5)       L = L ∪ {a}
(6)    end if
(7) end for
(8) Return L
```

Algorithm 2: Function candidates (C_k).

a feasible solution is found (i.e., a portfolio covering R). Actually, the algorithm progressively generates nonfeasible candidates until a feasible solution is found, which sets the first values for the minimal redundancy (η) and the portfolio size. However, such behavior implies that, depending on the coverage structure of components over requirements, these initial values could be identified after a very expensive searching process.

Indeed, the procedure could be improved if a convenient initial value for η was known at the very start of the algorithm. Then, let us consider the following program:

$$\begin{aligned} \min \quad & \mu \\ \text{St.} \quad & \sum_i y_i \leq \mu \\ & \sum_j a_{ji} x_j = y_i, \quad \forall i \\ & y_i \geq 1, \quad \forall i \\ & x_j \in \{0,1\}, \quad \forall j. \end{aligned} \tag{5}$$

In this case, we enforce the monocriteria version of the bicriteria ILP program, where its optimal solution does not necessarily solve the original problem, but it gives a good upper bound for the redundancy and the portfolio size. In what follows a comparison between the situation with and without an upper bound is analyzed. Results for both programs are presented in Section 4.

4. Results

In order to know if differences exist between the original algorithm and the version with an upper bound, both applied on model (1), a group of instances has been defined. An instance corresponds to a set of m projects, n requirements, and an information matrix $A^T_{m\times n} = (a_{ji})$. According to exploratory results [15], the instances are tested against different ratio of zeroes, or density, in A: 50%, 75%, and 80%. For each instance and expected ratio of zeroes, thirty random matrices have been filled using a Montecarlo process, agreeing to the given density value. In consequence, each instance has been simulated thirty times and the average time to solution, measured in milliseconds, has been calculated. A MacBook Pro I5, 8 Gb RAM, and 2.3 Ghz was used for the experiments.

Average time to find the Pareto front for the algorithm with the upper bound (WUB) and the algorithm without that bound (NUB) is presented in Table 1. We compare these results with those found with an Apriori-based algorithm [13, 14]. In this algorithm, which we call AP, the pruning rules are the same: minimum number of objects and redundancy. Only average time less or equal to 1 minute is reported, which emphasizes cases where both or one of algorithms respond in a very short period of time. Cases where no entry is shown for an instance mean that the respective model is not capable of solving the problem in such time.

When a particular simulation of the WUB algorithm is considered, the initial value of redundancy is selected as the optimal value obtained in the respective simulation run in NUB. In this way, we enforce the WUB model to present its best behavior. As expected, the more requirements or number of projects increase, the more time to solution rises. Results show that the WUB model outperforms the others, except in one case (underlined). Density δ (the ratio of zeroes) is an important condition for algorithms. NUB and AP are very sensitive to that feature, but it allows us to consider how this density acts on the algorithm performance. We hypothesize that distribution of zeroes in the information matrix may be important, because it determines the way that the first feasible solution is found.

In order to know how the distribution of zeroes impacts algorithms, a sorting process is applied on the information matrices for every case of the experiment as follows.

(i) Requirements (columns) are sorted from left to right, according the number of times they are covered.

(ii) Projects (rows) are sorted in descending order according to the number of requirements they cover.

Therefore, two processes are added to the analysis: the WUB and NUB algorithms, where an early sorting process is applied to the current information matrix, named WUB/Ord and NUB/Ord, respectively. In Table 2, results for WUB and the new processes are compared.

It is observed that WUB/Ord performs better than other algorithms. This suggests that sorting could have good effects on results. In Table 3, WUB/Ord is compared to other two versions of the Apriori algorithm, the first with a sorting process and the upper limit for redundancy (WAP/Ord) and the second with the sorting process but without the upper limit (NAP/Ord).

These results show that WUB/Ord is not the best method in all cases. For example, it does not solve some instances solved by the Apriori based processes. Conversely, WUB/Ord may outperform these processes in different situations. In general, we conclude that the distribution of zeroes in the information matrix is a critical issue. Additionally, the sorting process is a good strategy, but it does not scale in front of sparse structures (e.g., $\delta = 0.8$).

Table 1: Time in (ms) to find solutions as a function of 0's density (δ).

Project	Req.	$\delta = 0.5$			$\delta = 0.75$			$\delta = 0.8$		
		WUB	NUB	AP	WUB	NUB	AP	WUB	NUB	AP
25	4	0	2	9	0	43	3	0	173	2
	8	0	3	4	0	2	1	0	5	58377
	16	2	4	9	5	15	22747		746	
	32	9	15	322	7533	9341				
	64	20	93	189						
50	4	1	2	28	0	788	8	0	11292	7
	8	0	2	26	0	37	10985	0	12075	
	16	3	7	24	0	809		0	281	
	32	57	112	4756						
	64	278	548							
100	4	3	16	214	1786	25970	102			85
	8	0	4	25	0	420	5203			
	16	0	8	1502	0	3218		0	9889	
	32	419	631	46459						
	64	7736	13434							
200	4	23	67	2397			562			162
	8	0	3	98			14274			
	16	0	52	4503	1	367		0	162	
	32	729	1983							
	64									

Table 2: Performance of algorithms in (ms), including sorting process WUB/Ord and NUB/Ord.

Project	Req.	$\delta = 0.5$			$\delta = 0.75$			$\delta = 0.8$		
		WUB	WUB /Ord	NUB /Ord	WUB	WUB /Ord	NUB /Ord	WUB	WUB /Ord	NUB /Ord
25	4	0	0	0	0	0	2	0	0	5
	8	0	0	1	0	0	0	0	0	1
	16	2	1	1	5	2	5			
	32	9	4	5	7533	1608	2068			
	64	20	10	36						
50	4	1	0	0	0	0	32	0	0	486
	8	0	0	0	0	0	11	0	0	2608
	16	3	2	2	0	0	231	0	18	78
	32	57	24	45						
	64	278	120	212						
100	4	3	2	1	1786	748	5948			
	8	0	0	1	0	0	19			
	16	0	1	18	0	0	725	0	0	2132
	32	419	151	252						
	64	7736	2733	4435						
200	4	23	11	11						
	8	0	0	1						
	16	0	0	21	1	120	372	0	0	51
	32	729	234	811						
	64									

Table 3: Comparison of WUB/Ord with Apriori sorting process WAP/Ord and NAP/Ord in (ms).

Project	Req.	$\delta = 0.5$			$\delta = 0.75$			$\delta = 0.8$		
		WUB /Ord	WAP /Ord	NAP /Ord	WUB /Ord	WAP /Ord	NAP /Ord	WUB /Ord	WAP /Ord	NAP /Ord
25	4	0	5	6	0	1	2	0	2	1
	8	0	1	1	0	0	1	0	1384	1374
	16	1	2	3	2	4196	4249			
	32	4	69	69	1608					
	64	10								
50	4	0	14		0	6	4	0	2	3
	8	0	2		0	6077	6085	0		
	16	2	2		0			18		
	32	24	651							
	64	120								
100	4	2	64	101	748	19	41		17	31
	8	0	9	13	0	6	10			
	16	1	21	32	0			0		
	32	151								
	64	2733								
200	4	11	875	1134		76	83		34	36
	8	0	33	52						
	16	0	55	94	120			0		
	32	234								
	64									

5. Discussion

Results show that performance of the algorithms depends on the number of projects, the number of requirements, and the distribution of zeroes in the information matrix of a given instance. A simple strategy to apply consists of having small size instances. This approach may be found in other studies. Actually, the portfolio composition problem has been modeled using multicriteria frameworks. The rationale of these methods consists of identifying interesting projects and next composing portfolios with them.

In [16] a two-stage combination of discrete and continuous multicriteria decision aid methods is proposed for mutual funds selection and composition. In the first stage, a multicriteria decision method is used to select the most promising mutual funds. In the second stage, a goal programming approach is applied in order to search for the best proportions of mutual funds in the final portfolio. Criteria used for selection and composition are grouped into three categories: criteria regarding expected outcome of investment in a mutual fund; criteria measuring risk to obtain an outcome; and criteria about efficiency of mutual funds. In [5] a two-level process is applied for selection and composition of portfolios from a set of projects. In the first level, the ELECTRE TRI decision aid method is used to sort projects according to given categories, for instance, good, average, and bad projects. In the second level, a portfolio on each category is identified as the list of projects satisfying specific constraints.

In [4], the selection and composition portfolio problem is also solved in a two-stage process but in a different way. First, a multicriteria decision analysis is applied to evaluate projects. Next, a knapsack optimization problem, where portfolio benefit is maximized, subject to a budget constraint, is solved to find an efficient portfolio. The optimization problem is included in an algorithm searching for the set of efficient portfolios.

Several aspects of techniques mentioned above may be relevant for our purpose. For instance, following [16] or [5], a process may be applied to filter noninteresting projects and reduce the size of the instance to be analyzed. In addition, according to [4], the constructive algorithm could be implemented as a succession of optimization problems aiding to find the Pareto frontier. However, these procedures do not necessarily allow detecting the whole set of efficient solutions. As an illustration, let us consider Table 4, where the information matrix of an instance composed of eight projects and eight requirements is presented. This problem has three solutions, each one equaling a redundancy value of 3: p_1p_7, p_2p_7, and p_6p_8. In this case, any of the exhaustive algorithms we have proposed above allows identifying these portfolios, while an optimization problem finds just one of them.

Our main purpose is to identify interesting solutions, before expending resources and time for information concerning projects and portfolios. Thus, the problem we have established here consists of having a reduced number of portfolios and projects to which limited resources may be

Table 4: A sample instance of projects covering requirements.

	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8
p_1	1	1	1		1	1		
p_2	1		1	1	1	1		
p_3		1	1					
p_4		1		1		1	1	
p_5	1	1		1				
p_6		1	1	1	1	1	1	
p_7		1	1	1		1	1	1
p_8	1	1	1			1		1

destined for more exhaustive analysis. Indeed, we could classify interesting projects as follows [9]:

(i) core: which are projects present in every nondominated portfolio,

(ii) borderline: projects present in some of the nondominated portfolios,

(iii) exterior: projects excluded from any nondominated portfolio.

For instance, in the previous example, no project in nondominated portfolios is core but borderline. In general, we assume that the best is to use resources and time for better information on core and borderline projects. In other words, the portfolio problem posed by Markowitz could be applied on these interesting projects.

Efficiency is a critical issue for algorithms. As observed in Table 3, the WUB/Ord method works very well whether an information matrix is not sparse, or equivalently, it is filled at least at 50. In [9] 50 projects were analyzed using the original algorithm we adapted here. In that case, projects were evaluated with regard to four criteria, subject to a budget constraint. They showed that the resolution of this problem was time consuming, but if an initial interesting nondominated portfolio was introduced early in the algorithm, the time of resolution improved. This idea is used in the WUB/Ord method, finding the upper limit for the redundancy value, which confirms that such strategy works fine even for the discrete model proposed in this paper.

6. Conclusions

A portfolio constructive model has been proposed, formulated in terms of satisfaction of a given set of technical requirements, with the minimum number of projects and minimum redundancy. The approach aids to support exploratory search of an analyst, aiding him/her to identify a restricted list of nondominated interesting portfolios from a project set, satisfying the whole set of requirements. It is argued that the approach is applicable in situations where detailed business information regarding projects is poor or difficult to obtain, given resources available. Nondominated interesting portfolios could be analyzed in further stages of development, when better information is obtainable.

Resolution of an integer linear program allows finding a first optimal solution, used as a feasible result aiding to construct new nondominated solutions. Numerical examples show that this process improves the computational efficiency of an original algorithm. We have found that the ratio of zeroes in the information matrix appears as a critical issue because it determines that the time passing before the first feasible solution is found by the algorithm. Therefore, a sorting process is proposed, which improves time to find solutions.

Further research needs to be done in order to test pruning with different distributions of zero in the information matrix. Additional research also includes analytical and simulation studies concerning the effect of metaheuristics (or hybrid methods) [17] and hyperheuristics [18] on quality and time to solution.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

Broderick Crawford is supported by Grant CONICYT/FONDECYT/REGULAR/1140897. Ricardo Soto is supported by Grant CONICYT/FONDECYT/INICIACION/11130459. Javier Pereira and Fernando Paredes are supported by Grant CONICYT/FONDECYT/REGULAR/1130455.

References

[1] H. Markowitz, "Portfolio selection," *The Journal of Finance*, vol. 1952, pp. 77–91, 1952.

[2] R. Steuer, Y. Qi, and M. Hirschberger, "Portfolio selection in the presence of multiple criteria," in *Handbook of Financial Engineering*, vol. 18 of *Springer Optimization and Its Applications*, pp. 3–24, Springer, 2008.

[3] J. Liesio, P. Mild, and A. Salo, "Robust portfolio modeling with incomplete cost information and project interdependencies," *European Journal of Operational Research*, vol. 190, no. 3, pp. 679–695, 2008.

[4] J. C. Lourenço, A. Morton, and C. A. Bana e Costa, "PROBE—a multicriteria decision support system for portfolio robustness evaluation," *Decision Support Systems*, vol. 54, no. 1, pp. 534–550, 2012.

[5] J. Zheng, O. Cailloux, and V. Mousseau, "Constrained multicriteria sorting method applied to portfolio selection," in *Proceedings of the 2nd International Conference on Algorithmic Decision Theory*, pp. 26–28, DIMACS, Rutgers University, New Jersey, NJ, USA, October 2011.

[6] M. Ehrgott and X. Gandibleux, "A survey and annotated bibliography of multiobjective combinatorial optimization," *OR Spektrum: Quantitative Approaches in Management*, vol. 22, no. 4, pp. 425–460, 2000.

[7] K. Doerner, W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer, "Pareto ant colony optimization: a metaheuristic approach to multiobjective portfolio selection," *Annals of Operations Research*, vol. 131, no. 1–4, pp. 79–99, 2004.

[8] K. Metaxiotis and K. Liagkouras, "Multiobjective evolutionary algorithms for portfolio management: a comprehensive literature review," *Expert Systems with Applications*, vol. 39, no. 14, pp. 11685–11698, 2012.

[9] J. Liesiö, P. Mild, and A. Salo, "Preference programming for robust portfolio modeling and project selection," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1488–1505, 2007.

[10] B. Roy, *Multicriteria Methodology for Decision Aiding*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[11] D. Bouyssou, T. Marchant, M. Pirlot, and P. Vincke, *Evaluation and Decision Models with Multiple Criteria*, vol. 86 of *International Series in Operations Research & Management Science*, Springer, New York, NY, USA, 2006.

[12] H. Simon, "A behavioral model of rational choice," *The Quarterly Journal of Economics*, vol. 69, no. 1, pp. 99–118, 1955.

[13] C. Lopez, H. Astudillo, and J. Pereira, "Towards robustness analysis of component-based systems built on imperfect information," in *Proceedings of the 1st International Workshop on Living with Uncertainties (IWLU '07) and 22nd International Conference on Automated Software Engineering (ASE '07)*, Atlanta, Ga, USA, November 2007.

[14] F. Paredes and J. Pereira, "A bi-criteria integer programming and algorithmic approach for software architecture alternatives modeling," in *Proceedings of the LIO-INFORMS Joint International Meeting*, Buenos Aires, Argentina, June 2010.

[15] F. Paredes, J. Pereira, and A. Candia, "Towards a requirementsbased bi-criteria approach to identify interesting project portfolios," in *Proceedings of the 21st International Conference on Multiple Criteria Decision Making*, pp. 13–17, Jyväskylä, Finland, June 2011.

[16] K. Pendaraki, C. Zopounidis, and M. Doumpos, "On the construction of mutual fund portfolios: a multicriteria methodology and an application to the Greek market of equity mutual funds," *European Journal of Operational Research*, vol. 163, no. 2, pp. 462–481, 2005.

[17] B. Crawford, R. Soto, E. Monfroy, C. Castro, W. Palma, and F. Paredes, "A hybrid soft computing approach for subset problems," *Mathematical Problems in Engineering*, vol. 2013, Article ID 716069, 12 pages, 2013.

[18] B. Crawford, R. Soto, E. Monfroy, W. Palma, C. Castro, and F. Paredes, "Parameter tuning of a choice-function based hyperheuristic using Particle Swarm Optimization," *Expert Systems with Applications*, vol. 40, no. 5, pp. 1690–1695, 2013.