Elsevier Editorial System(tm) for Computers & Operations Research
Manuscript Draft

Manuscript Number: COR-D-13-00081R1

Title: Multi-objective Metaheuristic Algorithms for the Resource-constrained Project Scheduling Problem with Precedence Relations

Article Type: Research Paper

Keywords: Project Management; Resource constrained project scheduling; Multi-objective optimization; Metaheuristics.

Corresponding Author: Mr. Helton Cristiano Gomes,

Corresponding Author's Institution:

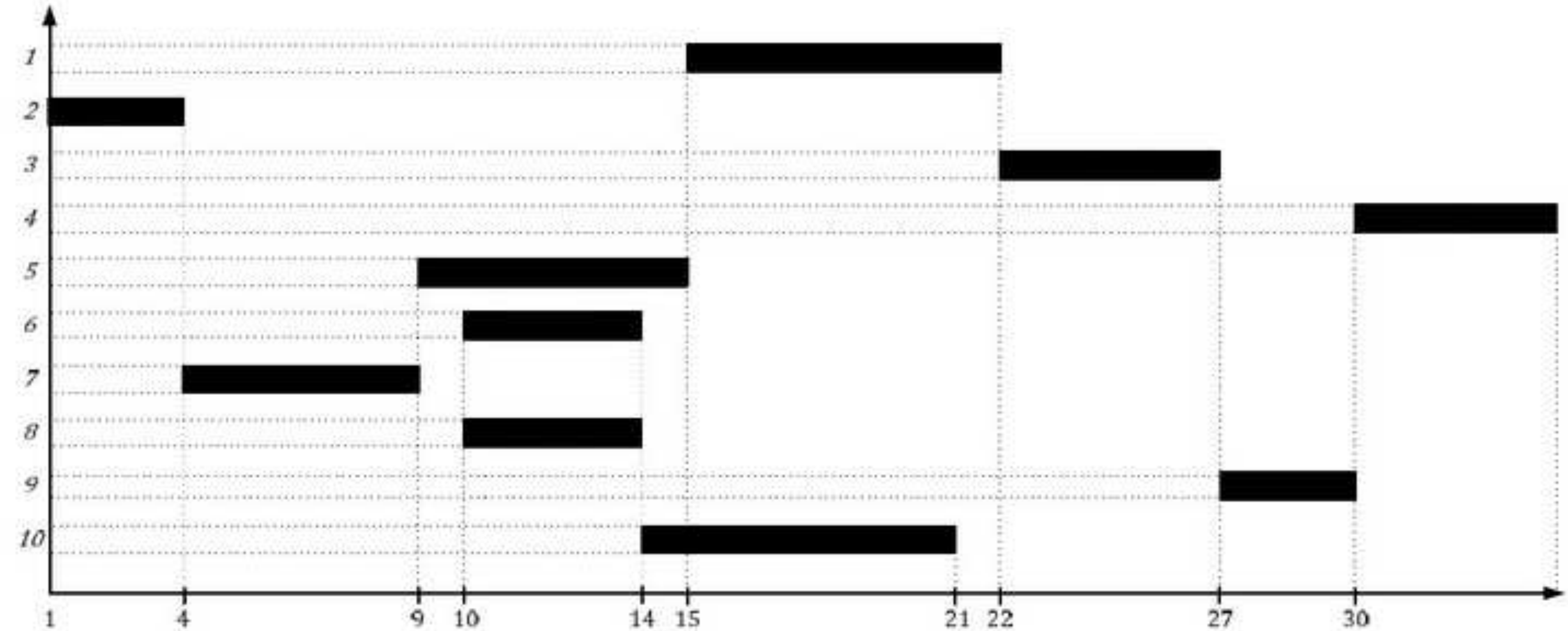
First Author: Helton Cristiano Gomes

Order of Authors: Helton Cristiano Gomes; Francisco A Neves; Marcone J Souza

Abstract: This study addresses the resource-constrained project scheduling problem with precedence relations, and aims at minimizing two criteria: the makespan and the total weighted start time of the activities. To solve the problem, five multi-objective metaheuristic algorithms are analyzed, based on Multi-objective GRASP (MOG), Multi-objective Variable Neighborhood Search (MOVNS) and Pareto Iterated Local Search (PILS) methods. The proposed algorithms use strategies based on the concept of Pareto Dominance to search for solutions and determine the set of non-dominated solutions. The solutions obtained by the algorithms, from a set of instances adapted from the literature, are compared using four multi-objective performance measures: distance metrics, hypervolume indicator, epsilon metric and error ratio. The computational tests have indicated an algorithm based on MOVNS as the most efficient one, compared to the distance metrics; also, a combined feature of MOG and MOVNS appears to be superior compared to the hypervolume and epsilon metrics and one based on PILS compared to the error ratio. Statistical experiments have shown a significant difference between some proposed algorithms compared to the distance metrics, epsilon metric and error ratio. However, significant difference between the proposed algorithms with respect to hypervolume indicator was not observed.

Fig 1

Click here to download high resolution image



Cover Letter

Computers and Operations Research

Dear Editor:

The study "Multi-objective Metaheuristic Algorithms for the Resource-constrained Project Scheduling Problem with Precedence Relations" arose as a requirement for obtaining a doctor degree in Civil Engineering in the Federal University of Ouro Preto, by the author Helton Cristiano Gomes. This author seeks to disseminate tools of Operations Research in the Civil Engineering, in order to reduce the duration and cost of projects, thereby improving the productive process in the civil construction.

Most sincerely,

Helton Cristiano Gomes

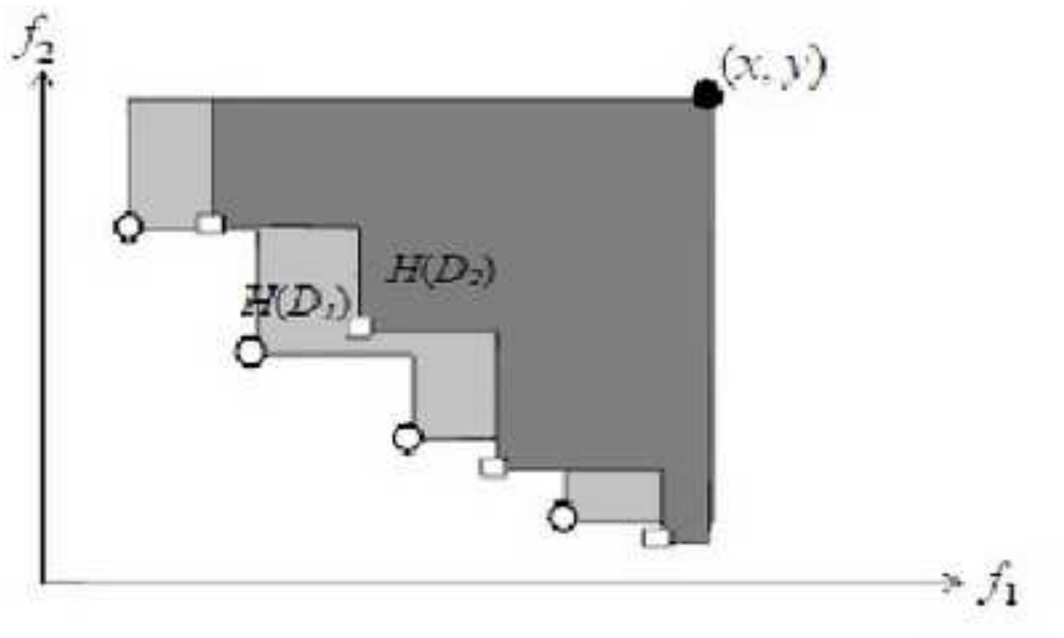
Corresponding author: Helton Cristiano Gomes

E-mail: helton.gomes@gmail.com
Full postal address:
35 Bourbon St, Jardim Ouro Verde
Zip Code: 38840-000
Carmo do Paranaíba – MG – Brazil
Phone numbers:
+55 31 87481302
+55 31 35522244 (home)
+55 34 38515658 (home)
+55 34 38559320 (work)

Highlights

- The Project Scheduling Problem is addressed as a multi-objective optimization problem;
- To solve the problem, five algorithms are analyzed and compared using four metrics;
- The *MOVNS_I* is superior on the majority of instances, regarding the distance metrics;
- The *GMOVNS* is superior regarding the hypervolume indicator and the *epsilon* metric;
- The algorithm *PILS* is superior regarding the error ratio.

Fig 2
Click here to download high resolution image



Multi-objective Metaheuristic Algorithms for the Resource-constrained Project Scheduling Problem with Precedence Relations

Helton Cristiano Gomes[a]*, Francisco de Assis das Neves[a], Marcone Jamilson Freitas Souza[b]

[a]*Departamento de Engenharia Civil*
[b]*Departamento de Ciência da Computação*
Universidade Federal de Ouro Preto
Campus Universitário - Morro do Cruzeiro
Ouro Preto – 35400-000 – Minas Gerais – Brazil

E-mail addresses: helton.gomes@gmail.com (H. C. Gomes), fassis@em.ufop.br (F. A. das Neves), marcone@iceb.ufop.br (M. J. F. Souza).

*Corresponding author:
E-mail: helton.gomes@gmail.com
Full postal address:
35 Bourbon St, Jardim Ouro Verde
Zip Code: 38840-000
Carmo do Paranaíba – MG – Brazil
Phone numbers:
+55 31 87481302
+55 31 35522244 (home)
+55 34 38515658 (home)
+55 34 38559320 (work)

ABSTRACT

This study addresses the resource-constrained project scheduling problem with precedence relations, and aims at minimizing two criteria: the makespan and the total weighted start time of the activities. To solve the problem, five multi-objective metaheuristic algorithms are analyzed, based on Multi-objective GRASP (*MOG*), Multi-objective Variable Neighborhood Search (*MOVNS*) and Pareto Iterated Local Search (*PILS*) methods. The proposed algorithms use strategies based on the concept of Pareto Dominance to search for solutions and determine the set of non-dominated solutions. The solutions obtained by the algorithms, from a set of instances adapted from the literature, are compared using four multi-objective performance measures: distance metrics, hypervolume indicator, *epsilon* metric and error ratio. The computational tests have indicated an algorithm based on *MOVNS* as the most efficient one, compared to the distance metrics; also, a combined feature of *MOG* and *MOVNS* appears to be superior compared to the hypervolume and *epsilon* metrics and one based on *PILS* compared to the error ratio. Statistical experiments have shown a significant difference between some proposed algorithms compared to the distance metrics, *epsilon* metric and error ratio. However, significant difference between the proposed algorithms with respect to hypervolume indicator was not observed.

Keywords**:** Project Management; Resource constrained project scheduling; Multi-objective optimization; Metaheuristics.

1. Introduction

Scheduling problems have been broadly studied in literature. Among those, the project scheduling (PSP) has been prominent. According to Oguz and Bala [1], the PSP is an important problem and it is challenging for those responsible for project management and for researchers in the related field. As said by the authors, one of the reasons for its importance is that it is a common problem in a great number of real situations of decision making, such as problems that originate in the project management of civil construction. The PSP is challenging, theoretically, for belonging to the class of NP-hard combinatorial optimization problems [2]. Thomas and Salhi [3], for example, state that the optimal solution of the PSP is hard to determine, especially for large-scale problems with resource and precedence constraints.

Despite several authors like Slowinski [4], Martínez-Irano *et al.* [5] and Ballestín and Blanco [6] consider that the resolution of the PSP involve several and conflicting objectives, few studies have been developed using this approach. According to Ballestín and Blanco [6], the number of possible multi-objective formulations for the PSP is very large, due to the countless objectives found in literature. These can be combined in several forms, thus generating new problems. Among the objectives that project managers are most interested in, according to Ballestín and Blanco [6], we can emphasize the following:

- minimization of the project makespan;
- minimization of the project earliness or lateness;
- minimization of the total project costs;
- minimization of the resources availability costs;
- minimization of the total weighted start time of the activities;
- minimization of the number of tardy activities;
- maximization of the project net present value.

According to Martínez-Irano *et al*. [5], the multi-objective formulation of a problem is particularly important when the objectives are conflicting, i.e., when the objectives may be opposed to one another.

In this work, the PSP with resource and precedence constraints (RCPSPRP) is addressed as a multi-objective optimization problem. Two conflicting objectives are considered in the problem: the makespan minimization and the minimization of the total weighted start time of the activities.

Several multi-objective optimization methods can be found in literature to solve this class of problems. Such methods can be basically divided into two groups: the classic and the metaheuristic methods. The classic methods consist of transforming the objective function vector into a scalar objective function, as it is the case of the Weighted Criteria and the Global Criterion methods. In this case the problem is treated as a mono-objective problem. The metaheuristic methods use metaheuristics to generate and analyze several solutions, as well as to obtain a set of non-dominated solutions. Literature revisions about the multi-objective metaheuristic methods, as published by Jones *et al.* [7], show the Multi-objective Tabu Search (MOTS) [8], the Pareto Simulated Annealing (PSA) [9], the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [10] and the Strength Pareto Evolutionary Algorithm II (SPEA-II) [11] as the most used. Due to the computational complexity of the RCPSPRP, according to Thomas and Salhi [3], the metaheuristic methods appear as the best form to solve it. According to Ballestín and Blanco [6], there are still few works that propose efficient methods for solving the multi-objective RCPSPRP.

According to Ballestín and Blanco [6], Slowinski [4] was the first author to explicitly represent the RCPSPRP as a multi-objective optimization problem. In the last years, some authors have addressed the RCPSPRP this way, as is the case of Viana and Sousa [12], Abbasi *et al*. [13], Kazemi and Tavakkoli-Moghaddam [14], Hamm *et al*. [15], Geyer [16], Ballestín and Blanco [6], among others.

Slowinski [4] applied the multi-objective linear programming to solve the RCPSPRP, allowing activities preemption. Renewable and non-renewable resources were considered. Makespan and costs minimization were choosing as objectives. Also, goal programming and fuzzy logic applications to the multi-objective RCPSPRP were discussed.

The PSA and MOTS algorithms were implemented by Viana and Sousa [12] to solve the multi-objective PSP considering renewable and non-renewable resources. Three minimizing criteria were used: makespan, mean weighted lateness of activities and sum of the violation of resource availability. The distance metrics were used to assess the algorithms efficiency.

Abbasi *et al*. [13] studied the multi-objective RCPSPRP considering only one renewable resource. Two objectives, makespan minimization and robustness maximization, were used. The authors incorporated these two objectives in a linear objective function and applied the Simulated Annealing metaheuristic to generate different solutions to the problem.

Kazemi and Tavakkoli-Moghaddam [14] presented a mathematical model for the multi-objective RCPSPRP considering positive and negative cash flows. The maximization of net present value and makespan minimization were considered as objectives. The NSGA-II was used to solve the problem.

Hamm *et al*. [15] have proposed an adaptation of the PSA for the multi-objective RCPSPRP but do not presented applications. According to authors, the differential of their algorithm is the rule of acceptance of new solutions, which depends on current temperature and of the dominance status of the neighbor solutions.

Geyer [16] has proposed a methodology based on the Genetic Algorithm metaheuristic for the multi-objective RCPSPRP. The author took into account economic and environmental objectives, as well as the preferences of the decision maker (project manager).

Ballestín and Blanco [6] have presented theoretical and practical fundamentals of multi-objective optimization applied to the RCPSPRP. A comparison between the PSA, NSGA-II and SPEA-II was presented when the makespan and resources availability costs minimizations were considered as objectives. Also, a study of seven multi-objective performance measures applied to the problem and their disadvantages was presented.

Recently, new metaheuristic methods have arisen in literature. The main examples are the Multi-objective GRASP (*MOG*) [17], Multi-objective Variable Neighborhood Search (*MOVNS*) [18] and Pareto Iterated Local Search (*PILS*) [19]. Such methods have been applied successfully in several types of problems, as have reported in [20], [21], [22], [23] and [24].

Due to the success of using these new methods, variations of the *MOG*, *MOVNS* and *PILS* are analyzed in this study to solve the RCPSPRP. For this, five algorithms were implemented: a *MOG*, a *MOVNS*, a *MOG* using *VNS* as local search, named *GMOVNS*, a *MOVNS* with an intensification procedure based on [24], named *MOVNS_I*, and a *PILS*. To assess the efficiency of the implemented algorithms, the results obtained through the use of instances adapted from literature were compared through four multi-objective performance measures: distance metrics, hypervolume indicator, *epsilon* metric, and error ratio. Statistic experiments were also carried out aiming at verifying, if there is a significant difference between the algorithms regarding the used performance measures.

From our knowledge, no article was found in literature using these new multi-objective metaheuristic methods to solve the problem addressed in this paper. Furthermore, in terms of algorithms, no work was found using *VNS* as local search for the *MOG*, as was done in the *GMOVNS*.

The rest of this paper is organized as following: in Section 2 the characteristics of the problem addressed in this study are described and in Section 3 the concepts of the multi-objective optimization are presented. In Section 4 the aforementioned multi-objective metaheuristic algorithms are described, while in Section 5 the characteristics of the instances, as well as the performance measures used to assess and compare the algorithms, are laid out. In Section 5 the results of the conducted tests are presented and analyzed. The last section concludes the work.

2. Problem Statement

The RCPSPRP consists of, given a set $A = \{1, ..., n\}$, with n activities, and, another $R = \{1, ..., m\}$, with m renewable resources with predefined availabilities B_k, determining the start time of execution (s_i) of each one of the n activities, assuring that the resource level and the precedence relation are not violated. The execution of each activity $i \in A$ has a duration (processing time) pre-determined p_i, a weight c_i and demand b_{ik} units of each resource $k \in R$.

The precedence relations determine that some activities need to be conducted in a particular sequence; that is, an activity cannot start while its precedent activities have not been finished.

Two objectives have been considered in the formulation used for the problem, the makespan minimization ($f_1(s)$) and the minimization of the total weighted start time of the activities ($f_2(s)$). The values of $f_1(s)$ and $f_2(s)$ are given by the Equations (1) and (2), where n+1 is an artificial activity ($p_{n+1} = c_{n+1} = 0$, $b_{n+1,k} = 0\ \forall k$) that represents the last one to be concluded and s_{n+1} represents the project's finishing time.

$$f_1(s) = Min\ s_{n+1} \tag{1}$$

$$f_2(s) = Min\ \sum_{i=1}^{n} \frac{c_i}{s_i} \tag{2}$$

The choice of such objectives was based on the fact these are conflicting. The $f_2(s)$ represents the modified minimization of the total weighted start time of the activities. This objective was modified to become conflicting with $f_1(s)$. While in the objective $f_1(s)$ the activities must be initiated as early as possible in the objective $f_2(s)$ is the opposite.

3. Some Definitions of Multi-objective Optimization

For the best understanding of the developed algorithms the definition of some concepts of multi-objective optimization are primarily necessary.

Definition 1 – Pareto Dominance

Given the feasible solutions s and s', it is found:

1º) if $f_k(s) \leq f_k(s')$ for all $k = 1, 2, ..., l$ and $f_j(s) < f_j(s')$ for any j, s will be a solution that dominates s';

2º) if $f_k(s') \leq f_k(s)$ for all $k = 1, 2, ..., l$ and $f_j(s') < f_j(s)$ for any j, s will be a solution dominated by s';

3º) if $f_j(s) < f_j(s')$ for any j e $f_i(s) > f_i(s')$ for any i, s and s' are stated non-dominated or indifferent.

Definition 2 – Pareto Optimality

A feasible solution s is named Pareto-optimal (or efficient) if there is no other feasible solution s' suck that s' dominates s, that is, a solution s' such as $f_k(s') \leq f_k(s)$ for all $k = 1, 2, ..., l$ and $f_j(s') < f_j(s)$ for any j.

The set of all Pareto-optimal solutions is termed Pareto-optimal front and as a result of the defined concepts, all the solutions that belong to the Pareto-optimal front are non-dominated (indifferent).

In all the algorithms proposed in this work the criterion of the Pareto Dominance was used, as described in this section, to assess the solutions generated along with its iterations and to determine the set of non-dominated solutions, denoted by D^*, to be returned by the algorithms.

4. Methodology

In this section the multi-objective algorithms proposed to solve of the RCPSPRP are described. In the first three sub-sections the common components of the five algorithms are presented, such as the representation of a solution, the generation of an initial solution and the neighborhood structures.

4.1. Representation of a Solution

A solution for the RCPSPRP is represented by a list $s = \{s_1, s_2,..., s_n\}$, where s_i indicates the start time of the execution of the activity i.

To illustrate, let us consider the instance given in Table 1. The instance has ten activities (named from 1 to 10) and two renewable resources (1 and 2). The availabilities of the resources are, respectively, 5 and 3 units. In this table, for each activity i, the duration p_i, the weight c_i, the demand for the resources 1 and 2 (b_{i1} and b_{i2}) and the successors activities are presented. The instance presented on Table 1 is an adaptation of Koné *et al.* [25].

Table 1: Data for an instance with 10 activities

Activities	1	2	3	4	5	6	7	8	9	10
p_i	7	3	5	5	6	4	5	4	3	7
c_i	200	300	500	100	600	200	500	300	300	200
b_{i1}	0	2	3	3	2	1	1	1	1	3
b_{i2}	2	1	3	2	1	0	3	1	1	1
Successors	3	6, 7	4, 9	11	1	1	5, 8	10	4	9

An example of a feasible solution, not necessarily optimal, for the presented instance is the list $s = \{15, 1, 22, 30, 9, 10, 4, 10, 27, 14\}$. The *Gantt* chart representing the described solution for the instance is presented in Fig. 1.

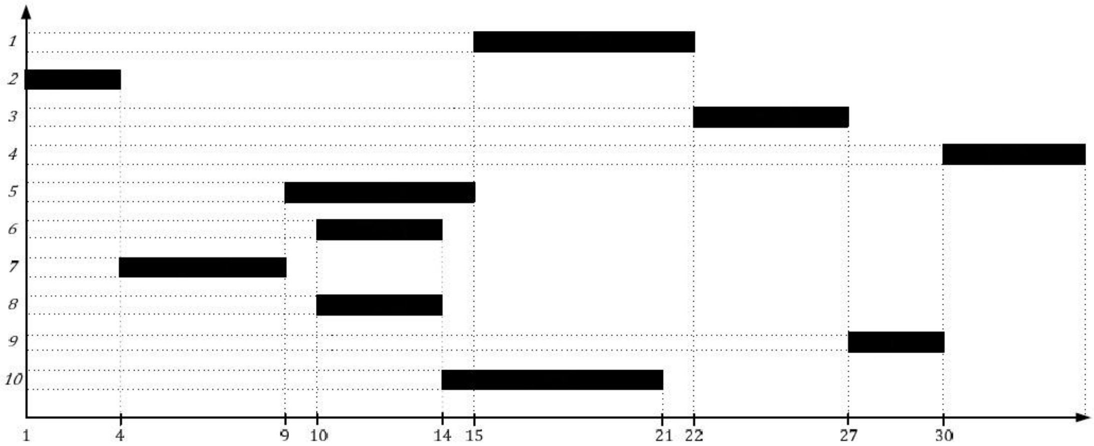


Fig. 1. *Gantt* chart for the presented solution

In the presented solution it is observed that the activity 2 is the first to be executed ($s_2 = 1$) and the activity 4 is the last ($s_4 = 30$). For this solution the objective functions values are: $f_1(s) = 35$ e $f_2(s) = 606.44$.

4.2. Initial Solution Generation

The proposed multi-objective algorithms start from an initial set of non-dominated solutions generated through a priority rule based scheduling heuristic. According to Kolisch [26], usually, this heuristic is composed of a priority rule and a schedule generation scheme for the determination of feasible sequencing.

For the generation of the initial set of non-dominated solutions the serial schedule generation scheme (S-SGS) proposed by Kelley [27] was used. In S-SGS, activities in an activity list L are scheduled in the order in which they appear in L; they are scheduled at the earliest clock time at which the required resources become available. An activity list L is a precedence feasible list of all activities of the given project [32]. If more than one activity can be assigned at a certain clock time, the activity to be scheduled is selected based on a priority rule. In the S-SGS used, three different types of priority rules were used as mentioned later:

(1) Lower duration: a solution s is generated by sequencing activities in non-decreasing order of the value of its duration;
(2) Bigger number of successors activities: a solution s is generated by sequencing activities in non-increasing order of its numbers of successors activities;
(3) Lower weight: a solution s is generated by sequencing activities in non-decreasing order of the value of its weight.

4.3. Neighborhood Structures

Local search methods usually use a neighborhood search to explore the space of feasible solutions of the addressed problem. The methods begin with a solution s, and generate a neighborhood of this solution. Such neighborhood is obtained by applying simple changes on solution s.

The algorithms developed in this paper use two neighborhood structures: exchange and insertion. For a given solution (sequence) s, the neighborhood structures are described below:

(1) *Exchange Neighborhood* ($N_1(s)$): the neighbors of s are generated by interchanging two activities in the sequence. The size of neighborhood $N_1(s)$ is $n(n - 1)/2$.

(2) *Insertion Neighborhood* ($N_2(s)$): the neighbors of s are generated by inserting one activity in another position of the sequence. The size of neighborhood $N_2(s)$ is $(n - 1)^2$.

By using the described two neighborhood structures, infeasible solutions can be generated due to resource constraints and precedence relations, but only the feasible solutions generated are considered and assessed by the algorithms.

4.4. Multi-objective Metaheuristic Algorithms for the RCPSPRP

4.4.1. MOG Algorithm

The Multi-objective GRASP (*MOG*) is a multi-objective optimization algorithm based on the metaheuristic Greedy Randomized Adaptive Search Procedure (GRASP) proposed by Feo and Resende [28]. The *MOG* version proposed in this work, based on Reynolds and Iglesia [17], is presented in the Algorithm 1.

Algorithm 1: *MOG*

Input: MOG_{max}, θ
Output: D^*
$D^* \leftarrow \phi$;
For (Iter = 1 to MOG_{max}) **do**
  $s \leftarrow$ *Construction_MOG*(s, θ, D^*);
  $s \leftarrow$ *LocalSearch_MOG*(s, D^*);
End_for;
  Return D^*;

As in the method proposed by Feo and Resende [28], the *MOG* is composed of two phases: construction and local search. In each one of the MOG_{max} iterations of Algorithm 1, a solution s is generated in the construction phase through an adaptation of S-SGS. This adaptation consists of the insertion of a randomization rate (θ) to the method, being the *greedy function*, a characteristic of GRASP, based on the priority rules described in Section 4.2. The pseudo-code of the procedure *Construction_MOG* is presented in Algorithm 1.1.

Algorithm 1.1: *Construction_MOG*

Input: s, θ, D^*
Output: s
$s \leftarrow \phi$;
Initialize the candidate list *CL*;
Determine randomly the value $\theta \in [0, 1]$;
Determine randomly a priority rule;
While ($CL \neq \phi$) **do**
  Determine *RCL* with the first θ % elements of *CL* which are based on the selected priority rule;
  Select randomly an element $t \in RCL$;
  $s \leftarrow s \cup \{t\}$;
  Update *CL*;
End_while;
$D^* \leftarrow$ non-dominated solutions of $D^* \cup \{s\}$;
Return s;

In Algorithm 1.1 the construction of a solution *s* starts with the generation of a list of activities *CL* that are candidates to be included in the sequencing. The *CL* is determined by the available activities to the execution, at the time instant considered, and with its precedent activities already being sequenced. From the *CL*, the value of θ, will define the restricted candidates list (*RCL*), where the *greedy function* is determined by the priority rule selected in Section 4.2, that is, the activity that has the biggest priority will be the one that will bring the biggest benefit by being included in the sequencing. Once the *RCL* is defined, an activity $t \in RCL$ is randomly selected and inserted in *s*, thus being the *CL* updated. Finally, the solution *s* generated is assessed to be part or not of *D**. Aiming at the generation of different solutions over the Pareto front, the value of $\theta \in [0, 1]$ and the priority rule to be used are randomly determined by each *Construction_MOG* procedure call.

In the local search phase, the solution *s* generated by the Algorithm 1.1 is modified by the exchange movement ($N_1(s)$), described in Section 4.3, in a way that new solutions are generated. The pseudo-code of the procedure *LocalSearch_MOG* is presented in Algorithm 1.2.

Algorithm 1.2: *LocalSearch_MOG*

```
Input: s, D*
Output: D*
Determine randomly a neighbor solution s´∈ N1(s);
For (each neighbor s´´∈ N1(s´)) do
   D* ← non-dominated solutions of D* ∪ {s´´};
End_for;
Return D*;
```

The Algorithm 1.2 starts with a random determination of a solution $s' \in N_1(s)$. Then the *D** set is updated through the evaluation of all the neighbors solutions $s'' \in N_1(s')$.

4.4.2. MOVNS Algorithm

The Multi-objective Variable Neighborhood Search (*MOVNS*) is an algorithm of multi-objective optimization presented by Geiger [18]. Its structure is based on metaheuristic Variable Neighborhood Search (VNS), delineated by Mladenovic and Hansen [29]. In Algorithm 2 the proposed version of *MOVNS*, based on Ottoni *et al.* [24], is presented.

Algorithm 2: *MOVNS*

```
Input: r, StoppingCriterion
Output: D*
{s1, s2, s3} ← solutions (sequencing) constructed by using 3 different priority rules;
D* ← non-dominated solutions of {s1, s2, s3};
While (StoppingCriterion = False) do
   Select randomly an unvisited solution s∈ D*;
   Mark(s) ← True;
   Determine randomly a neighborhood structure Ni ∈ {N1,...,Nr};
   Determine randomly a solution s´∈ Ni(s);
   For (each neighbor s´´∈ Ni(s´)) do
      D* ← non-dominated solutions of D* ∪ {s´´};
   End_for;
        If (all the solutions of D* are marked as visited) then
           All marks must be removed;
        End_if;
     End_while;
     Return D*;
```

Algorithm 2 starts with the generation of three solutions (s_1, s_2, s_3) using the S-SGS described in Section 4.2. Each of these was attained using a different priority rule. These solutions are, then, inter-assessed and, the non-dominated ones are stored in the D^* set. Accordingly with Geiger [18], from each local search iteration a non-visited solution $s \in D^*$ is randomly selected and marked as visited ($Mark(s) \leftarrow True$). A neighborhood structure $N_i \in \{N_{1,\ \dots,} N_r\}$ is also randomly selected. Two neighborhood structures (r = 2) were used in Algorithm 2, as described in Section 4.3. After that, a solution $s' \in N_i(s)$ is randomly determined and the set D^* is updated through the assessment of all neighbors solutions $s'' \in N_i(s')$. Finally, it is checked whether all solutions belonging to D^* are marked as visited. If they are, the marking is removed from all solutions. This procedure is repeated until the stopping criterion is fulfilled.

4.4.3. GMOVNS algorithm

The *GMOVNS* proposed in this study is a hybrid algorithm that combines *MOG* features with *MOVNS* features, described in Sections 4.4.1 and 4.4.2, respectively. The algorithm follows the structure described in Algorithm 1, but has modifications on the construction and on local search phases. The pseudo-code of *GMOVNS* is presented in Algorithm 3.

Algorithm 3: *GMOVNS*

Input: $GMOVNS_{max}$, θ, β
Output: D^*
$D^* \leftarrow \phi$;
For (Iter = 1 to $GMOVNS_{max}$) **do**
  $D_1 \leftarrow Construction\_GMOVNS(\theta, \beta, D_1)$;
  $D_1 \leftarrow LocalSearch\_GMOVNS(D_1, D^*, r)$;
End_for;
Return D^*;

As it is observed in Algorithm 3, the *GMOVNS* - just like the *MOG* - is composed of two phases: construction and local search. Algorithm 3.1 describes the *Construction_GMOVNS* procedure, in which a set of non-dominated solutions D_1 is generated on each algorithm iteration.

Algorithm 3.1: *Construction_GMOVNS*

Input: θ, β
Output: D_1
$D_1 \leftarrow \phi$;
For (Iter = 1 to β) **do**
  $s \leftarrow \phi$;
  Initialize the candidate list *CL;*
  Determine randomly the value $\theta \in [0, 1]$;
  Determine randomly a priority rule;
  While ($CL \neq \phi$) **do**
    Let *RCL* be a list with the θ % first elements of *CL* based on the selected priority rule;
    Select randomly an element $t \in RCL$;
    $s \leftarrow s \cup \{t\}$;
    Update *CL;*
  End_while;
  $D_1 \leftarrow$ non-dominated solutions of $D_1 \cup \{s\}$;

End_for;
Return D_1;

In each one of the $GMOVNS_{max}$ iterations of Algorithm 3, β solutions are generated during the construction phase described in Algorithm 3.1. These solutions are assessed and the non-dominated ones are stored in the D_1 set. All the solutions of this phase are generated through the same adaptation of S-SGS used in *MOG*. For the different solutions to be generated, a value for $\theta \in [0, 1]$ and a priority rule are randomly determined during the construction of each solution.

In the local search phase of the *GMOVNS*, the metaheuristic VNS was proposed with two neighborhood structures, described in Section 4.3 and used in Algorithm 2. The VNS is better capable of exploring the space of feasible solutions to this problem due to its systematic swap of the neighborhood structure. With this, the quality of set D^* can be improved. The pseudo-code of the procedure *LocalSearch_GMOVNS* is presented in Algorithm 3.2.

Algorithm 3.2: *LocalSearch_GMOVNS*
Input: D_1, D^*, r, *StoppingCriterion*
Output: D^*
While (*StoppingCriterion = False*) **do**
  Select randomly an *unvisited* solution $s \in D_1$;
  $Mark(s) \leftarrow True$;
  Determine randomly a neighborhood structure $N_i \in \{N_{1,\ldots,}N_r\}$;
  Determine randomly a solution $s' \in N_i(s)$;
  For (*each neighbor* $s'' \in N_i(s')$) **do**
    $D_1 \leftarrow$ non-dominated solutions of $D_1 \cup \{s''\}$;
  End_for;
  If (*all the solutions of* D_1 *are marked as visited*) **then**
    All marks must be removed;
  End_if;
End_while;
$D^* \leftarrow$ non-dominated solutions of $D^* \cup D_1$;
Return D^*;

On each iteration of Algorithm 3.2 the solution s to be explored is determined randomly within the non-visited ones that belong to set D_1 generated in the construction phase. Then, a neighborhood structure $N_i \in \{N_{1,\ \ldots,}\ N_r\}$ and a neighbor solution $s' \in N_i(s)$ are chosen randomly. The D_1 set is then updated through the assessment of all the neighbors solutions $s'' \in N_i(s')$. Finally, it is checked, if all the solutions that belong to D_1 are marked as visited, and, if they are, the marking is removed from all solutions. This procedure is repeated until the stopping criterion is fulfilled. From D_1 on each iteration the D^* set is updated with the assessment of all solutions of $D^* \cup D_1$.

4.4.4. MOVNS_I Algorithm

Two variants of algorithm *MOVNS* are found in literature. One is proposed by Ottoni *et al*. [24] and another by Arroyo *et al*. [23]. These variants consist of adding an intensification procedure to the algorithm. The intensification of the search around the best solution is obtained, for example by the application of small perturbations on it. The *MOVNS* with intensification, denominated *MOVNS_I*, proposed in this work is based on the variant proposed by Ottoni *et al.* [24] and it is described in Algorithm 4.

Algorithm 4: *MOVNS_I*
Input: r, *StoppingCriterion*
Output: D^*

$\{s_1, s_2, s_3\} \leftarrow$ solutions (sequencing) constructed by using 3 different priority rules;
$D^* \leftarrow$ non-dominated solutions of $\{s_1, s_2, s_3\}$;
While (*StoppingCriterion* = *False*) **do**
  Select randomly an *unvisited* solution $s \in D^*$;
  $Mark(s) \leftarrow True$;
  Determine randomly a neighborhood structure $N_i \in \{N_{1,\ldots,} N_r\}$;
  Determine randomly a solution $s' \in N_i(s)$;
  For (*each neighbor* $s'' \in N_i(s')$) **do**
    $D^* \leftarrow$ non-dominated solutions of $D^* \cup \{s''\}$;
  End_for;
  If (*all the solutions of D* are marked as visited*) **then**
    All marks must be removed;
  End_if;
  Select randomly a solution $s \in D^*$;
  $D_1 \leftarrow$ ***INTENSIFICATION***(s, d);
  $D^* \leftarrow$ non-dominated solutions of $D^* \cup D_1$;
End_while;
Return D^*;

According to Ottoni *et al.* [24], the intensification procedure is composed by two stages: *destruction* and *reconstruction*, as presented in Algorithm 4.1.

Algorithm 4.1: *INTENSIFICATION*

Input: s, d
Output: D_1
$s_r \leftarrow \phi$;
$s_p \leftarrow s$;
Define randomly the weights w_1 and $w_2 \in [0, 1]$, such that $w_1 + w_2 = 1$;
For (i = 1 to d) **do**
  Let $s_p(j)$ the j-th activity of s_p randomly selected;
  Remove $s_p(j)$ from s_p;
  Insert $s_p(j)$ in s_r;
End_for;
For (i = 1 to (d – 1)) **do**
  $f_p^* \leftarrow \infty$;
  For (j = 1 to (n - d + i)) **do**
    $s' \leftarrow$ result of the insertion of the i-th activity from s_r in the j-th position from s_p;
    If ($f(s') < f_p^*$) **then**
      $s_p^* \leftarrow s'$;
      $f_p^* \leftarrow f(s')$;
    End_if;
  End_for;
  $s_p \leftarrow s_p^*$;
End_for;
For (j = 1 to n) **do**
  $s' \leftarrow$ result of the insertion of the last activity from s_r in the j-th position from s_p;
  $D_1 \leftarrow$ non-dominated solutions of $D_1 \cup \{s'\}$;
End_for;
Return D_1;

The intensification procedure starts with the *destruction* stage, in which d activities are removed from a solution $s \in D^*$ randomly selected. In out experiments, d was fixed at 4. This strategy results in the generation of a partial solution s_p, composed by $(n - d)$ activities, and of a set s_r with the d activities removed from s. Then the solution s is reconstructed inserting $(d - 1)$ activities of s_r in s_p. To do this, an activity belonging to s_r is inserted in all possible positions of s_p. The position that offers the best partial solution is selected. The assessment of the partial solutions is done through a weighted function given by the equation $f = w_1 f_1 + w_2 f_2$, where w_1 and w_2 are associated weights with the objective functions and $w_1 + w_2 = 1$. This procedure is made until $(d - 1)$ activities of s_r are inserted in s_p. Finally, the last activity of s_r is inserted in the partial solution s_p in all its possible positions. All solutions generated by this last insertion process are assessed and the non-dominated ones are stored in D_1.

After the intensification procedure, the set D^* is updated through the assessment of all $D^* \cup D_1$ solutions.

4.4.5. PILS Algorithm

The Pareto Iterated Local Search (*PILS*) is a multi-objective optimization algorithm proposed by Geiger [19]. It is based on metaheuristic Iterated Local Search (ILS) delineated by Lourenço *et al*. [30]. The basic pseudo-code of *PILS* is presented in Algorithm 5.

Algorithm 5: *PILS*

Input: r, *StoppingCriterion*
Output: D^*
Determine the initial set of non-dominated solutions D^*;
Select randomly a solution $s \in D^*$;
While (*StoppingCriterion* = *False*) **do**
  $i \leftarrow 1$;
  While ($i < r \wedge$ *StoppingCriterion* = *False*) **do**
    For (*each neighbor* $s' \in N_i(s)$) **do**
      $D^* \leftarrow$ non-dominated solutions of $D^* \cup \{s'\}$;
    End_for;
    If ($\exists\ s' \in N_i(s) |\ s'$ dominates s) **then**
      $s \leftarrow s'$;
      Rearrange the neighborhood structures $N_1, ..., N_r$ in some random order;
      $i \leftarrow 1$;
    End_if;
    Else
      i ++;
    End_else;
  End_while;
  Mark(s) $\leftarrow$ *True*;
  If ($\exists\ s' \in D^* /\ s'$ has not yet been visited) **then**
    $s \leftarrow s'$;
  End_if;
  Else
    Select randomly a solution $s' \in D^*$;
    $s'' \leftarrow$ ***PERTURBATION***(s');
    $s \leftarrow s''$;
  End_else;
End_while;
Return D^*;

Algorithm 5 starts with the generation of an initial set of non-dominated solutions D^*, using the procedure S-SGS and the priority rules from Section 4.2. After that, a solution $s \in D^*$ is randomly selected, that starts to be the current solution and all its neighborhood is explored. The neighborhood structures used are presented on Section 4.3 (r = 2). In case any neighbor solution $s' \in N_i(s)$ dominates the current solution s, then s' starts to be the new current solution, the neighborhood structures are then randomly reordered and the procedure returns to its first neighborhood structure of the new generated order. This procedure is repeated until all solutions belonging to D^* are visited, that is, until the algorithm arrives in a local optimum in the explored neighborhood. Once this is done, a solution $s' \in D^*$ is randomly selected on which a perturbation is applied. The objective on perturbation a solution is to explore other local optimums. The perturbation used here is proposed originally by Geiger [19] and works as follows: after the selection of solution $s' \in D^*$, one position $j \leq n-4$ is randomly determined along with four consecutive activities of s' on the positions j, j+1, j+2 and j+3. A solution s'' is then generated by applying the activities swap movement on positions j and j+3, and on the activities from positions j+1 and j+2. Thus, the activities before the activity on position j and those before the activity on position j+3, stay on the same position after the perturbation. After that the solution s'' starts to be the current solution and its neighborhood is explored. In case all neighbors solutions from the one generated by the perturbation are dominated by any solution that belongs to D^*, then the perturbation procedure is repeated. This procedure is repeated until the stopping criterion is fulfilled.

5. Computational Experiments

The five algorithms presented in this study were coded in C++ and executed on an AMD *Turion II Dual-Core* with a 2.20GHz and 4.0GB of RAM.

The algorithms were run with the same stopping criterion (*StoppingCriterion*) based on the limit of the generated solutions. In literature, this stopping criterion is extensively used for performance comparison of mono and multi-objective algorithms for the PSP, as illustrated in [31], [6], [32] and others. Several values are found in literature, but in this work the limit of generated solutions equal to 5000 was used as the stopping criterion for the algorithms.

In the execution of the *MOG* algorithm the value 100 for the $MOG_{\max}$ parameter was empirically defined. For the execution of *GMOVNS*, the value 10 to β and the value 100 to the $GMOVNS_{max}$ also were empirically defined.

5.1. Problems Instances

According to Viana and Sousa [12] the study of multi-objective RCPSPRP involves some difficulties, specially related to the availability of instances shown in literature. Several mono-objective problems can be found, like the Project Scheduling Problem Library (PSPLIB), developed by Kolisch and Sprecher [33], but nothing was found by the authors regarding multi-objective instances.

Due to this, 160 instances from the PSPLIB, available in [34], were used to test the algorithms. These instances have the numbers of activities n = 30, 60, 90 and 120. For each value of n, 40 instances were used, from which 4 different types of renewable resources are available. As the instances were used for the mono-objective RCPSPRP and they do not present associated weights to the activities. Thus, such weights were then generated randomly and uniformly distributed over the interval [1, 500].

Due to the fact the proposed algorithms using random choices, in the same way which [23] and [24], the five algorithms were run five times independently (replicates), with five different seeds randomly generated, for all the 160 instances. From the solutions attained on the five runs of an algorithm, a set of non-dominated solutions is determined for each instance.

5.2. Performance Measures

The comparison between non-dominated solution sets attained by multi-objective optimization algorithms is not a trivial task. Several performance measures (metrics) of multi-objective algorithms can be found in literature, such as in [35], [36], [37], [38] and [39].

In this work, to assess the quality of the non-dominated solutions attained by the proposed algorithms, four multi-objective performance measures were used: distance metrics, hypervolume indicator, *epsilon* metric and error ratio.

For each instance D_i is the non-dominated solutions set found by the algorithm i, for $i = 1, 2, \ldots, h$, and h is the number of assessed algorithms. From these sets a reference set, denoted by *Ref*, where $Ref = \{s \in D_1 \cup D_2 \cup \ldots \cup D_h |\ s$ is a non-dominated solution$\}$, is determined. The *Ref* set is the best known Pareto-optimal front. The performance of an algorithm is then measured in terms of the quality of the solution obtained by this algorithm regarding the solutions in *Ref*. Based on the *Ref* set, the definition of the used performance measures are presented as follows:

Distance metrics: measures the proximity between the solutions of set D_i and the solutions of set *Ref*. It also measures the solutions spreading on set D_i. The closer to zero the distances are, the better the quality of the solutions found by the algorithm will be. The formulas used to calculate the average (D_{av}) and maximum (D_{max}) distances from the D_i solutions compared to the *Ref* set are:

$$D_{av}(D_i) = 100 \times \frac{1}{|Ref|} \sum_{s \in Ref} \min_{s' \in D_i} d(s, s') \tag{3}$$

$$D_{\max}(D_i) = \max_{s \in Ref} \{ \min_{s' \in D_i} d(s, s') \} \times 100 \tag{4}$$

in which $|Ref|$ is the cardinality of set *Ref* and:

$$d(s, s') = \max \left\{ \frac{(f_1(s) - f_1(s'))}{\Delta_1}, \frac{(f_2(s) - f_2(s'))}{\Delta_2} \right\} \tag{5}$$

Δ_j is the difference between the biggest and the smallest value of the objective function f_j, considering the solutions of set *Ref*.

The distances D_{av} and D_{max} are broadly used as performance measure of multi-objective algorithms such as in [9], [12] and [24].

Hypervolume indicator: measures the covered or dominated area by set D_i. For the minimization of two objectives, a reference point (x, y) is used to limit this coverage, denoted by $H(D_i)$, where x and y are upper bounds for f_1 and f_2, respectively. A larger dominance area indicates that the solutions attained by the algorithm generated a good coverage on the Pareto-optimal front. The value of the hypervolume difference $(H^-(D_i))$ is calculated by the Equation (6):

$$H^-(D_i) = H(Ref) - H(D_i) \tag{6}$$

As $H(Ref) > H(D_i)$, the smaller the value of $H^-(D_i)$, the better the quality of set D_i will be. In Fig. 2, the covered area by the solution sets D_1 and D_2 are illustrated.

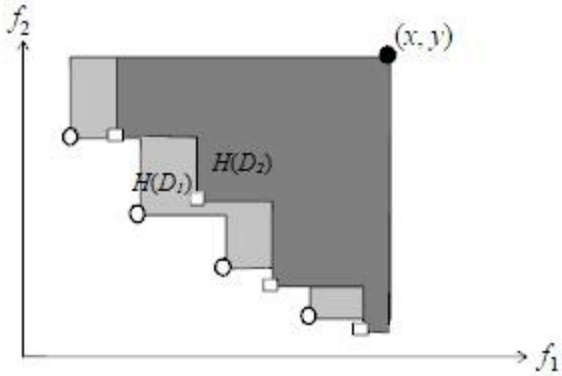


Fig. 2. Examples of areas covered by two sets of solutions

As it is shown on Fig. 2, $H(D_1) > H(D_2)$, therefore $H^{-}(D_1) < H^{-}(D_2)$, which indicates the solutions from the D_1 set are "better" than the ones from the D_2 set.

Epsilon metric: given a set D_i and $z^a = (z_1^a, \ldots, z_r^a)$ and $z^b = (z_1^b, \ldots, z_r^b)$, two solutions belonging to the sets D_i and *Ref*, respectively, the *epsilon* metric denoted by $I_\varepsilon^1(D_i)$, measures the maximum normalized distance from set D_i in relation to set *Ref*, and is calculated by Equation (7):

$$I_\varepsilon^1(D_i) = \max_{z^b \in Ref}\{\min_{z^a \in D_i}\{\max_{1 \le j \le r} \frac{z_j^a}{z_j^b}\}\} \tag{7}$$

Therefore, the quality of a non-dominated solutions set D_i attained by an algorithm to a determined instance is assessed in relation to set *Ref* and as $I_\varepsilon^1(D_i)$ measures the maximum distance of D_i in relation to *Ref*, thus a value close to zero of $I_\varepsilon^1(D_i)$ indicates a good quality of set D_i. To use the *epsilon* metric to assess a D_i set, the values of the objective functions must be normalized according to the following equation:

$$f_i^*(s) = \left(\frac{f_i(s) - f_i^{\min}}{f_i^{\max} - f_i^{\min}} \times 100\right) \tag{8}$$

where $f_i^{\min}$ and $f_i^{\max}$ are, respectively, the smallest and the biggest value found to the *i*-th objective considering the solutions belonging to set *Ref*. Hence, the values of the objective function $f_i^*(s)$ calculated by Equation (8) are in the interval [0, 100].

Error ratio: indicates the percentage of the solutions that belong to set D_i that don't belong to set *Ref*. The metric based on Veldhuizen [36] and denoted by TE_i, is calculated by Equation (9):

$$TE_i = \frac{|D_i| - |Ref \cap D_i|}{|D_i|} \times 100 \tag{9}$$

where $|D_i|$ corresponds to the cardinality of set D_i and $|Ref \cap D_i|$ to the number of reference solutions originating from the set D_i. According to Coello and Lamont [40], $TE_i = 0$ indicates that all solutions belonging to D_i are part of *Ref*. On the other hand, $TE_i = 100$ indicates that no solutions from D_i are part of *Ref*. Thus, the nearest to zero the value of the TE_i the better is the performance of the algorithm.

5.3. Computational Results

For each group of 40 instances of size *n*, Table 2 shows the average values (in seconds) of the computational time spent by each algorithm to obtain the non-dominated solutions sets.

Table 2: Average Computational Time

N	**Algorithm**				
	MOG	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
30	0.19	0.41	0.41	0.56	1.16
60	0.88	3.99	2.98	4.08	11.09
90	2.72	12.91	10.96	15.82	52.95
120	7.73	53.75	36.46	57.62	154.71

Table 2 shows that all algorithms presented low computational effort, i.e., obtained the sets of non-dominated solutions in an acceptable time.

Except Table 5, all following tables in this section presents, for each group of 40 instances of size *n*, the average values of the performance measure attained by each algorithm.

On Tables 3 and 4 the results attained by the algorithms in relation to the distance metrics are presented. On Table 3 the results regarding the average distance and on Table 4 the results regarding the maximum distance are presented.

Table 3: Distance Metrics Results – Average Distance (%)

n	**Algorithm**				
	MOG	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
30	15.90	18.29	13.89	3.68	6.20
60	92.82	19.15	19.07	13.64	5.83
90	14.59	14.50	15.14	6.52	9.81
120	32.69	16.87	26.44	3.49	12.57
Average	39.00	17.20	18.63	6.83	8.60

Table 4: Distance Metrics Results – Maximum Distance (%)

n	**Algorithm**				
	MOG	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
30	64.70	49.15	55.77	12.53	20.23
60	297.51	62.79	78.53	35.48	21.04
90	47.53	40.66	38.35	14.86	29.85
120	88.30	44.42	107.61	14.18	37.20
Average	124.51	49.26	70.06	19.26	27.08

Through Tables 3 and 4, it is verified that the *MOVNS_I* algorithm is the one that produces lower average values, that is, closer to zero, from the average and maximum distances to the majority set of instances. The *MOVNS_I* didn't attain lower average values to the set of instances with $n = 60$ only where *PILS* showed better results.

As presented in Section 5.2, the distance metrics measures the proximity between the solutions of a set D_i and the solutions of set *Ref*. Therefore, the higher the percentage of solutions of D_i in the *Ref* set, the lower tends to be the values of the distance metrics. The values of the distance metrics tend to be smaller, but those values also depend of the distance between D_i solutions and solutions belonging to *Ref* set obtained by other algorithms. For each group of 40 instances of size *n*, Table 5 shows average percentages of solutions obtained by the *MOVNS_I* and *PILS* algorithms which are part of *Ref* set.

Table 5: Average Percentages of Solutions of the *MOVNS_I* and *PILS* in the *Ref* Set

n	**Algorithm**		**Difference**
	MOVNS_I	*PILS*	
30	56.75	59.79	3.04
60	27.08	55.48	28.48
90	36.70	38.64	1.94
120	42.11	44.16	2.05

Table 5 shows that algorithms had presented very close values for the average percentage except for the set with n = 60. In this case the percentage difference was 28.48%. For the groups of instances in which the difference between the average percentages was small, the *MOVNS_I* algorithm had presented better results for D_{av} and D_{max}, even the *PILS* showing higher percentage. However, when the difference between these average percentages was large, as in the case of the instances set with n = 60, better values for the distances was obtained by the *PILS*. Therefore, the *MOVNS_I* had presented in most cases a better performance regarding the distance metrics.

On Table 6 the values attained by the proposed algorithms regarding the hypervolume indicator are presented.

Table 6: Hypervolume Indicator Results

n	**Algorithm**				
	MOG	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
30	927.60	485.50	360.20	386.90	369.10
60	1803.00	747.20	501.50	1249.30	739.00
90	2712.60	2420.60	2021.00	2275.60	2399.30
120	5358.00	5841.00	3654.40	3727.40	3881.40
Average	2700.30	2373.58	1634.28	1909.80	1847.20

Through Table 6 it is verified that the *GMOVNS* algorithm presented lower average values, compared with the other algorithms, from the hypervolume indicator for all sets of instances.

On Table 7 the results attained by the proposed algorithms are shown regarding the *epsilon* metric.

Table 7: *Epsilon* Metric Results

n	**Algorithm**				
	MOG	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
30	1.45	1.91	1.22	1.85	1.24
60	1.41	1.72	1.30	1.51	1.46
90	1.91	1.87	1.59	1.93	1.84
120	1.34	1.68	1.24	1.94	1.49
Average	1.53	1.80	1.34	1.81	1.51

Through Table 7 it is verified that the *GMOVNS* is the algorithm that produces lower average values for the *epsilon* metric for all sets of instances.

On Table 8, the values attained by the algorithms proposed regarding the error ratio are presented.

Table 8: Error Ratio Results (%)

n	**Algorithm**				
	MOG	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
30	77.14	40.74	59.60	43.25	40.21
60	86.13	74.70	66.89	72.92	44.52
90	89.67	62.13	75.10	63.30	61.36

120	91.42	90.50	67.09	57.89	55.84
Average	86.09	67.02	67.17	59.34	50.48

As it can be observed on Table 8, the *PILS* algorithms presented, in all sets of instances, a lower average value for the error ratio. This means that, based on error ratio, the algorithm *PILS* was superior to the others.

5.3.1. Analysis of the Results

Based on the average values of the computational time spent by each algorithm to obtain the non-dominated solutions sets, we can see that all the algorithms were computationally eficeintes, obtaining sets of solutions in an acceptable time. For all the instances sets, the *MOG* and *PILS* algorithms had presented the lowest and highest average computational time, respectively.

Results attained from the computational experiments, showed that the *GMOVNS* algorithm had best performance. The *GMOVNS* has generated better results for two of the four multi-objective performance measures assessed: hypervolume indicator and *epsilon*. This means that the *GMOVNS* algorithm produces a better coverage for the Pareto-optimal front and that the non-dominated solutions generated by this algorithm are closer to the *Ref* set.

Regarding the distance metrics, in general, the *MOVNS_I* algorithm has obtained the lowest average values for this metric. Therefore, the *MOVNS_I* has achieved better distributed solutions throughout the *Ref* set.

For all the instances sets, the *PILS* algorithm had obtained the better results for the error ratio. The *PILS* had presented, on average, the higher percentage of solutions belonging to the *Ref* set.

5.4. Statistical Analysis

The experiments that follow aim at verifying, if there is a significant difference between the algorithms proposed in this paper, concerning the multi-objective performance measures used. These experiments were conducted with the assistance of the Minitab® computational package on its 16th version. It is emphasized here that this experimentation enables the researchers to make inferences to the population of all instances.

To conduct the experiments, the statistical technique Analysis of Variance (ANOVA) was chosen, as described by Montgomery [41]. The interest is then to test the equality of the population means (μ) to the five implemented algorithms against the inequality of the means.

In the ANOVA application two hypotheses were tested:

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 \tag{1}$$

$$H_1 : \mu_i \neq \mu_j \text{ } \textit{for at least one pair } (i, j)\textit{, with } i, j = 1, 2, 3, 4, 5 \textit{ and } i \neq j \tag{2}$$

In the test (1)-(2), the null hypothesis (1) represents the equality of the population means hypothesis in relation to the analyzed multi-objective performance measure on the five algorithms, that is, it conjectures that there is no significant difference between these algorithms regarding the metric. The hypothesis (2), on the other hand, conjectures the opposite.

However, to apply the ANOVA, the sample data should be normally distributed in this case, and the population variances (σ^2) approximately equal between the factor levels, regarding the algorithms proposed here.

Although the test is based on the supposition that the sample data should be normally distributed, according to Kulinskaya *et al*. [42], this hypothesis is not critical when the sizes of the samples are at least 15 or 20. Once all the samples on this work have the equal size to 160 (number of instances used) for each algorithm, thus, the normality is not critical. Hence, the normality premise is verified for all the algorithms regarding all metrics. To use the ANOVA it is needed, then, the verification of only the variances proximity between the data from the algorithms regarding each metrics. For this, the following hypotheses were tested:

$$H_0 : \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \sigma_4^2 = \sigma_5^2 \tag{3}$$

$$H_1 : \sigma_i^2 \neq \sigma_j^2 \textit{ for at least one pair } (i, j)\textit{, with } i, j = 1, 2, 3, 4, 5 \textit{ and } i \neq j \tag{4}$$

In the test (3)-(4), the null hypothesis (3) represents the equality of the population variances hypothesis in relation to the analyzed multi-objective performance measure on the five algorithms. The hypothesis (4) conjectures the opposite.

By applying these hypothesis tests is possible to calculate a test statistic that allows us to accept or reject the null hypothesis. In *Statistical Inference* is usual to represent this test statistic for *p-value*. From the value of this test statistic and of a criterion for acceptance/rejection is possible to conclude, with a significance level α defined *a priori*, which of the hypotheses accept. That is, if α ≥ *p-value* rejects H_0. All the tests in this section have been executed with a significance level α = 0.05 (5%).

Nevertheless, the ANOVA does not tell us which pairs of algorithms present significant differences, that is, result in different means to each assessed metric. To answer this question the method of the Least Significant Difference (LSD), also known as the Fisher's method [41], is used.

All the tables of ANOVA results presented in this section show the calculated value of the *p-value*, the sample means, the sample standard deviations and the interval limits with 95% of confidence on the population means from the analyzed multi-objective performance measure, in accordance with each algorithm.

- *Distance Metrics*

For the distance metrics the hypothesis test (3)-(4) was used to verify the proximity of variances between the data of all algorithms. The *p-value* statistics calculated for this test was equal to 0.078 for the average distance, and 0.054 for the maximum distance. Once α < *p-value* for both distance metrics, the variance equality hypothesis is accepted between the population data to the five algorithms. Therefore, once the premise is verified, the ANOVA is applied to the concerning data from the metrics.

The application of ANOVA to the average distance data allowed us to calculate the values presented in Table 9.

Table 9: The results of ANOVA for the Average Distance

p-value	**Algorithm**				
0.025	*MOG*	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
Mean	39.0	17.2	18.6	6.8	8.6
Standard Deviation	103.4	29.1	31.6	10.7	10.7
IC (μ, 95%)	(5.9; 72.1)	(7.9; 26.5)	(8.5; 28.7)	(3.4; 10.2)	(5.2; 12.0)

According to the results on Table 9, *p-value* = 0.025. Therefore, it can be stated that the null hypothesis should be rejected, that is, as α ≥ *p-value*, there are enough statistical evidences to conclude that the average values regarding the average distance are different on each algorithm. By using the LSD method, it can be stated that there are statistical evidences showing that the average values, regarding the average distance, are different within the following algorithm pairs: *MOG* × *MOVNS_I* and *MOG* × *PILS*.

The application of ANOVA to the maximum distance data allowed us to calculate the values presented on Table 10.

Table 10: The results of ANOVA for the Maximum Distance

p-value	**Algorithm**				
0.002	*MOG*	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
Mean	124.5	49.2	70.1	19.3	27.1
Standard Deviation	311.4	59.3	137.1	19.9	28.1
IC (μ, 95%)	(24.9; 224.1)	(30.3; 68.2)	(26.2; 113.9)	(12.9; 25.7)	(18.1; 36.1)

According to the results on Table 10, *p-value* = 0.002. Therefore, it can be stated that the null hypothesis should be rejected, that is, as $\alpha \geq$ *p-value*, there are enough statistical evidences to conclude that the average values regarding the maximum distance are different on each algorithm. By using the LSD method, it can be stated that there are statistical evidences showing that the average values regarding the maximum distance, are different within the following algorithm pairs: *MOG* × *MOVNS*, *MOG* × *MOVNS_I* and *MOG* × *PILS*.

- *Hypervolume Indicator*

For the hypervolume indicator it was verified the proximity of variances between the data of all algorithms by the hypothesis test (3)-(4). The calculated *p-value* statistics was equal to 0.567 and, as $\alpha <$ *p-value*, the hypothesis of the variances equality between the population data on the five algorithms is accepted. Once verified the premise, the ANOVA is applied to the data of this metric.

The application of ANOVA to the hypervolume indicator data allowed us to calculate the values presented on Table 11.

Table 11: The results of ANOVA for the Hypervolume Indicator

p-value	**Algorithm**				
0.452	*MOG*	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
Mean	2700.0	2374.0	1634.0	1910.0	1847.0
Standard Deviation	3099.0	3159.0	2235.0	2729.0	2958.0
IC (μ, 95%)	(1709.2; 3691.4)	(1363.4; 3383.8)	(919.4; 2349.2)	(1037.0; 2782.6)	(901.1; 2793.3)

According to the results on Table 11, *p-value* = 0.452. Therefore, it can be stated that the null hypothesis should be accepted, that is, as $\alpha <$ *p-value*, there are enough statistical evidences to conclude, with a 5% significance level ($\alpha = 0.05$), that the average values regarding the hypervolume indicator equal within all algorithms.

- *Epsilon metric*

For the *epsilon* metric it was verified the proximity of variances between the data of all algorithms by the hypothesis test (3)-(4). The calculated *p-value* statistics was equal to 0.082 and, as $\alpha <$ *p-value*, the hypothesis of the variances equality between the population data on the five algorithms is accepted. Once verified the premise, the ANOVA is applied to the data of this metric.

The application of ANOVA to the *epsilon* metric data allowed us to calculate the values presented on Table 12.

Table 12: The results of ANOVA for the *Epsilon* Metric

p-value	**Algorithm**				
0.024	*MOG*	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
Mean	1.53	1.80	1.34	1.81	1.51
Standard	0.81	0.97	0.48	0.83	0.56

Deviation					
IC (μ, 95%)	(1.27; 1.79)	(1.48; 2.11)	(1.18; 1.49)	(1.54; 2.07)	(1.33; 1.68)

According to the results on Table 12, *p-value* = 0.024. Therefore, it can be stated that the null hypothesis should be rejected, that is, as $\alpha \geq$ *p-value*, there are enough statistical evidences to conclude that the average values regarding the *epsilon* metric are different between the algorithms. By using the LSD method, it can be stated that there are statistical evidences showing that the average values, regarding the *epsilon* metric, are different within the following algorithm pairs: *GMOVNS* × *MOVNS* and *GMOVNS* × *MOVNS_I*.

- *Error ratio*

For the error ratio the hypothesis test (3)-(4) was used to verify the proximity of variances between the data of all algorithms. The calculated *p-value* statistics was equal to 0.002 and, as $\alpha >$ *p-value*, the hypothesis of the variances equality between the population data on the five algorithms is rejected. Therefore, this premise is not verified, and consequently, the ANOVA cannot be applied to this metric's data. As a result, the Kruskal-Wallis non-parametric test [43] was used. The difference from ANOVA to the Kruskal-Wallis non-parametric test is that the later, instead of working with means, uses population medians (η). The test can be used to verify the medians equality of two or more populations and, applying to this work, tests the following hypothesis:

$$H_0 : \eta_1 = \eta_2 = \eta_3 = \eta_4 = \eta_5 \tag{5}$$

$$H_1 : \eta_i \neq \eta_j \text{ for at least one pair } (i, j), \text{ with } i, j = 1, 2, 3, 4, 5 \text{ and } i \neq j \tag{6}$$

In the test (5)-(6), the null hypothesis (5) represents the equality of the population medians hypothesis in relation to the error ratio on the five algorithms, that is, it conjectures that there is no significant difference between these algorithms regarding this metric. The hypothesis (6), on the other hand, conjectures the opposite.

For this test of hypothesis, the *p-value* statistics calculation brought the value 0.000. Once the significance level $\alpha = 0.05$ is adopted and $\alpha >$ *p-value*, the median equality between the population data on the five algorithms should be rejected. Hence, it is statistically concluded that the algorithms differ in error ratio. By comparing the pairs of algorithms, it can be stated that there are statistical evidences that the median values from the error ratio are different between: *MOG* × *MOVNS*, *MOG* × *GMOVNS*, *MOG* × *MOVNS_I*, *MOG* × *PILS*, *MOVNS* × *PILS* and *GMOVNS* × *PILS*.

6. Conclusions

This work addressed the resource-constrained project scheduling problem with precedence relations as a multi-objective optimization problem, having two optimization criteria that were tackled: the makespan minimization and the minimization of the total weighted start time of the activities.

To solve the problem, five algorithms were implemented: *MOG*, *MOVNS*, *MOG* using *VNS* as local search, denominated *GMOVNS*; *MOVNS* with intensification procedure based on Ottoni *et al*. [24], denominated *MOVNS_I*; and *PILS*.

The algorithms were tested in 160 instances adapted from literature, and compared using four multi-objective performance measures: distance, hypervolume, *epsilon* and error ratio. Based on the results attained from the computational experiments, we can see that all algorithms were computationally efficient, obtaining sets of non-dominated solutions in an acceptable time, and three conclusions were obtained: first, the *MOVNS_I* has shown to be superior than the other algorithms on the majority of instances, regarding the distance metrics;

second, the *GMOVNS* is superior regarding the hypervolume indicator and the *epsilon* metric; and third, the algorithm *PILS* is superior regarding the error ratio. Statistical experiments were conducted and have revealed that there is a significant difference between some proposed algorithms concerning the distance, *epsilon*, and error ratio metrics. However, significant difference between the proposed algorithms with respect to hypervolume indicator was not observed.

References

[1] Oguz O, Bala H. A Comparative Study of Computational Procedures for the Resource Constrained Project Scheduling Problem. European Journal of Operational Research 1994;72:406–416.

[2] Garey MR, Jonhson DS. Computers and Intractability: A Guide to the Theory of NP-Completeness. New York: W. H. Freeman and Company; 1979.

[3] Thomas PR, Salhi S. A Tabu Search Approach for the Resource Constrained Project Scheduling Problem. Journal of Heuristics 1998;4:123-139.

[4] Slowinski R. Multi-objective Project Scheduling Under Multiple-category Resource Constraints. In: Advances in Project Scheduling, Amsterdam: Elsevier, 1989.

[5] Martínez-Irano M, Herrero JM, Sanchis J, Blasco X, Garcia-Nieto S. Applied Pareto Multi-objective Optimization by Stochastic Solvers. Engineering Applications of Artificial Intelligence 2009;22:455-465.

[6] Ballestín F, Blanco R. Theoretical and Practical Fundamentals for Multi-objective Optimization in Resource-constrained Project Scheduling Problems. Computers & Operations Research 2011;38:51-62.

[7] Jones DF, Mirrazavi SK, Tamiz M. Multi-objective Metaheuristics: an Overview of the Current State-of-art. European Journal of Operational Research 2002;137:1-19.

[8] Hansen MP. Tabu Search for Multi-objective Optimization: MOTS. In: Proceedings of the 13th International Conference on Multiple Criteria Decision Making, University of Cape Town; 1997, p. 6-10.

[9] Czyzak P, Jaszkiewicz A. Pareto Simulated Annealing – a Metaheuristic Technique for Multiple-objective Combinatorial Optimization. Journal of Multi-Criteria Decision Analysis 1998;7:34–47.

[10] Deb K, Agrawal S, Pratap A, Meyarivan T. A Fast Elitist Nondominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. In: Schoenauer M, editor. Parallel Problem Solving from Nature (PPSN VI). Berlin, Germany: Springer; 2000, p. 849-858.

[11] Zitzler E, Laumanns M, Thiele L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Computer Engineering and Networks Laboratory (TIK), Zurich, Switzerland; 2001.

[12] Viana A, Sousa JP. Using Metaheuristics in Multi-objective Resource Constrained Project Scheduling. European Journal of Operational Research 2000;120:359-374.

[13] Abbasi B, Shadrokh S, Arkat J. Bi-objective Resource-constrained Project Scheduling with Robustness and Makespan Criteria. Applied Mathematics and Computation, 2006;180:146-152.

[14] Kazemi FS, Tavakkoli-Moghaddan R. Solving a Multi-objective Multi-mode Resource-constrained Project Scheduling with Discounted Cash flows. In: Proceedings of the 6th International Management Conference, Tehran, Iran, 2008.

[15] Hamm M, Beibert U, König M. Simulation-Based Optimization of Construction Schedules by Using Pareto Simulated Annealing. In: Proceedings of the 18th International Conference on the Application of Computer: Science and Mathematics in Architecture and Civil Engineering, Weimar, Germany; 2009.

[16] Geyer P. Component-oriented Decomposition for Multidisciplinary Design Optimization in Building Design. Advanced Engineering Informatics 2009;23:12-31.

[17] Reynolds AP, Iglesia B. A Multi-objective GRASP for Partial Classification. Soft Computing 2009; 13:227-243.

[18] Geiger MJ. Randomized Variable Neighborhood Search for Multi-objective Optimization. 4th EU/ME: Design and Evaluation of Advanced Hybrid Meta-Heuristics, p. 34-42; 2008.
[19] Geiger MJ. Foundations of the Pareto Iterated Local Search Metaheuristic. In: Proceedings of the 18th MCDM, Chania, Greece, June; 2006.
[20] Arroyo JEC, Vieira PS, Vianna DS. A GRASP Algorithm for the Multi-criteria Minimum Spanning Tree Problem. Annals of Operations Research 2008;159(1):125-133.
[21] Minella G, Ruiz R, Ciavotta M. A Review and Evaluation of Multi-objective Algorithms for the Flowshop Scheduling Problem. *INFORMS* Journal on Computing 2010;20:451.
[22] Oliveira Junior PL, Arroyo JEC, Souza VAA. Heurísticas GRASP e ILS para o Problema No-wait Flowshop Scheduling Multiobjetivo. In: Proceedings of the 17th Simpósio Brasileiro de Pesquisa Operacional, Bento Gonçalves, Brasil; 2010.
[23] Arroyo JEC, Ottoni RS, Oliveira AP. Multi-objective Variable Neighborhood Search Algorithms for a Single Machine Scheduling Problem with Distinct Due Windows. Electronic Notes in Theoretical Computer Science 2011;281:5-19.
[24] Ottoni RS, Arroyo JEC, Santos A G. Algoritmo VNS Multiobjetivo para um Problema de Programação de Tarefas em uma Máquina com Janelas de Entrega. In: Proceedings of the 18th Simpósio Brasileiro de Pesquisa Operacional, Ubatuba, Brasil; 2011.
[25] Koné O, Artigues C, Lopez P, Mongeau M. Event-Based MILP Models for Resource-Constrained Project Scheduling Problems. Computers & Operations Research 2009;38:3-13.
[26] Kolisch R. Serial and Parallel Resource-Constrained Project Scheduling Methods Revisited: Theory and Computation. European Journal of Operational Research 1996;90:320-333.
[27] Kelley JE. The Critical Method: Resources Planning and Scheduling. In: Industrial Scheduling, New Jersey: Prentice-Hall; 1963, p. 347-365.
[28] Feo TA, Resende MGC. Greedy Randomized Adaptive Search Procedures. Journal of Global Optimization 1995;40:109-133.
[29] Mladenovic N, Hansen P. Variable Neighborhood Search. Computers & Operations Research 1997;24:1097-1100.
[30] Lourenço HR, Martin OC, Stüzle T. Iterated Local Search. In: Glover F, Kochenberger GA, editors. Handbook of Metaheuristics, Dordrecht, Netherlands: Kluwer Academic; 2003, p. 320-353.
[31] Deiranlou M, Jolai F. A New Efficient Genetic Algorithm for Project Scheduling under Resource Constraint. World Applied Science Journal 2009;7:987-997.
[32] Agarwal A, Colak S, Erenguc S. A Neurogenetic Approach for the Resource-constrained Project Scheduling Problem. Computers & Operations Research 2011;38:44-50.
[33] Kolisch R, Sprecher A. PSPLIB – A Project Scheduling Problem Library. European Journal of Operational Research 1996;96:205-216.
[34] PSPLIB. <http://129.187.106.231/psplib/>.
[35] Hansen M, Jaszkieweiz A. Evaluating the Quality of Approximations to the Non-dominated Set. Technical Report. Department of Mathematical Modeling, Technical University of Denmark; 1998.
[36] Veldhuizen D. Multi-objective Evolutionary Algorithms: Classifications, Analyses and New Innovations. Ph.D. Thesis, School of Engineering of the Air Force Institute of Technology, Dayton, Ohio; 1999.
[37] Zitzler E, Ded K, Thiele L. Comparison of Multi-objective Evolutionary Algorithms: Empirical Results. Evolutionary Computation 2000;8:173-195.
[38] Deb K, Jain S. Running Performance Metrics for Evolutionary Multi-objective Algorithms. Technical Report, Kanpur Genetic Algorithms Laboratory, Indian Institute of Technology Kanpur, India; 2002.
[39] Fonseca C, Knowles J, Thiele L, Zitzler E. A Tutorial on the Performance Assessment of Stochastic Multi-objective Optimizers. In: Proceedings of the 3th International Conference on Evolutionary Multi-criterion Optimization (EMO), Guanajuato, Mexico; 2005.

[40] Coello CA, Lamont GB. Applications of Multi-objective Evolutionary Algorithms. World Scientific Printers, Singapore: Pte. Ltd; 2004.
[41] Montgomery DC. Design and Analysis of Experiments. 7th ed. New York: John Wiley & Sons; 2009.
[42] Kulinskaya E, Staudte RG, Gao A. Power Approximations in Testing for Unequal Means in a One-way ANOVA Weighted for Unequal Variances. Communication in Statistics 2003;32(12):2353-2371.
[43] Montgomery DC, Runger CR. Estatística Aplicada e Probabilidade para Engenheiros. 5th ed. Brasil: LTC; 2012.

Detailed Response to Reviewers

Dear Reviewers,

We would like to thank for their relevant comments. They have contributed to the improvement of our work.

In the following pages we highlight changes made and answers for their comments.

Reviewers' comments:

Reviewer #1:

Even if the originality of the methods can be discussed, because it is a simple adaptation to the problem addressed for some of them, this is a good and well structured paper. The analysis of the experimental data is very Excellent.

(1) However a few turns of phrase sometimes complex and lengthy can be simplified.

The proposed changes in the text were relevant and, therefore, performed. It was made, also, a review in the text verifying paragraphs which can be simplified, reducing the turns of phrase. The review was made especially in the Statistical Analysis section.

Reviewer #2:

The authors applied five multi-objective metaheuristics: MOG, MOVNS, GMOVNS, MOVNS_I and PILS, to solve the resource-constrained project scheduling problem with precedence relations, and aimed to minimize two criteria: the makespan and the total weighted start time of the activities. The proposed algorithms use strategies based on the concept of Pareto Dominance to search for solutions and determine the set of non-dominated solutions. Four multi-objective performance measures: distance metrics, hypervolume indicator, epsilon metric and error ratio, were adopted to evaluate and compare the five heuristics.

The authors were appreciated for their efforts in conducting the set of computational tests to analyze the performances of the five heuristics for solving the multi-objective RCPSPRP. Nevertheless, this work lacks theoretical contributions to the existing literature, as the authors just applied or adapted several existing algorithms to solve an existing problem. Specific comments are given as follows.

(1) The authors did not provide an adequate literature review on multi-objective RCPSPRP. In this manuscript, the authors mentioned "According to Ballestín and Blanco [6], the number of possible multi-objective formulations for the PSP is very large, due to the countless objectives found in literature", but no specific references or examples were given. At least, the authors need to review solution algorithms to multi-objective RCPSPRP.

It was inserted in the manuscript (page 3, paragraphs 3-10) a literature review about the multi-objective RCPSPRP. This literature review presents some objectives and algorithms used in the resolution of the problem. The literature review is below.

According to Ballestín and Blanco [6], Slowinski [4] was the first author to explicitly represent the RCPSPRP as a multi-objective optimization problem. In the last years, some authors have addressed the RCPSPRP this way, as is the case of Viana and Sousa [12], Abbasi *et al*. [13], Kazemi and Tavakkoli-Moghaddam [14], Hamm *et al*. [15], Geyer [16], Ballestín and Blanco [6], among others.

Slowinski [4] applied the multi-objective linear programming to solve the RCPSPRP, allowing activities preemption. Renewable and non-renewable resources were considered. Makespan and costs minimization were choosing as objectives. Also, goal programming and fuzzy logic applications to the multi-objective RCPSPRP were discussed.

The PSA and MOTS algorithms were implemented by Viana and Sousa [12] to solve the multi-objective PSP considering renewable and non-renewable resources. Three minimizing criteria were used: makespan, mean weighted lateness of activities and sum of the violation of resource availability. The distance metrics were used to assess the algorithms efficiency.

Abbasi *et al*. [13] studied the multi-objective RCPSPRP considering only one renewable resource. Two objectives, makespan minimization and robustness maximization, were used. The authors incorporated these two objectives in a linear objective function and applied the Simulated Annealing metaheuristic to generate different solutions to the problem.

Kazemi and Tavakkoli-Moghaddam [14] presented a mathematical model for the multi-objective RCPSPRP considering positive and negative cash flows. The maximization of net present value and makespan minimization were considered as objectives. The NSGA-II was used to solve the problem.

Hamm *et al*. [15] have proposed an adaptation of the PSA for the multi-objective RCPSPRP but do not presented applications. According to authors, the differential of their algorithm is the rule of acceptance of new solutions, which depends on current temperature and of the dominance status of the neighbor solutions.

Geyer [16] has proposed a methodology based on the Genetic Algorithm metaheuristic for the multi-objective RCPSPRP. The author took into account economic and environmental objectives, as well as the preferences of the decision maker (project manager).

Ballestín and Blanco [6] have presented theoretical and practical fundamentals of multi-objective optimization applied to the RCPSPRP. A comparison between the PSA, NSGA-II and SPEA-II was presented when the makespan and resources availability costs minimizations were considered as objectives. Also, a study of seven multi-objective performance measures applied to the problem and their disadvantages was presented.

(2) In addition, the authors did not explain the reason why they aimed to minimize the makespan and total weighted start time of activities, despite that there were several objectives considered in the literature.

According to Martínez-Irano *et al*. [5], the multi-objective formulation of a problem is particularly important when the objectives are conflicting, i.e., when the objectives may be opposed to one another. (Page 2, paragraph 3)

Therefore, the choice of such objectives was based on the fact these are conflicting.

This justification was inserted in the manuscript (page 4, paragraph 7).

(3) While the authors aimed to minimize the total weighted start time, in Eq.(2), they actually took the sum inversed start times.

The objective $f_2(s)$ (Eq. 2) represents the modified minimization of the total weighted start time of the activities. This objective was modified to become conflicting with $f_1(s)$. While in the objective $f_1(s)$ the activities must be initiated as early as possible in the objective $f_2(s)$ is the opposite. (Page 4, paragraph 7)

(4) In section 4.3, the proposed two neighborhood structures: exchange and insertion may lead to infeasible solutions, due to resource constraints and precedence relations.

By using the proposed two neighborhood structures, infeasible solutions can be generated due to resource constraints and precedence relations, but only the feasible solutions generated are considered and assessed by the algorithms.

This justification was inserted in the manuscript (page 7, paragraph 1).

(5) The reference set approach described in Section 5.2 can only be used to determine the relative performances of the five algorithms. We cannot judge whether or not the problem is effectively and efficiently solved by the algorithms. The reference set are constructed using the solutions obtained by the algorithms under comparison. What if all there algorithms are not good. There lacks an absolute benchmark.

The ideal would be to compare the results obtained by the algorithms with the Pareto-optimal set. However, this set is not always known or available. In these cases, the Pareto approximation set of the union of sets obtained by the different algorithms is used as the reference set.

As there were no results in the literature for the multi-objective RCPSPRP with the same characteristics as studied in this work, the efficiency of the algorithms only can be assessed based on the reference set (*Ref*), which is the best known set of solutions to the problem. This procedure is used in most studies which deal multi-objective optimization, as is the case of Viana and Sousa [12], Arroyo *et al*. [20], Arroyo *et* al. [23], Ottoni *et* al. [24], among others.

According to Ballestín and Blanco [6], is necessary to be created exact algorithms capable of calculating the Pareto-optimal set for many important problems as the multi-objective RCPSPRP. The generated solutions would be used to compare and assess the sets of solutions obtained by metaheuristic algorithms.

(6) The four measures were used to determine the relative effectiveness of the five algorithms. What about the computational efficiency?

It was inserted in the manuscript (page 16, paragraphs 1-2) the results and comments regarding computational efficiency of the algorithms. The results and comments are below.

For each group of 40 instances of size *n*, Table 2 shows the average values (in seconds) of the computational time spent by each algorithm to obtain the non-dominated solutions sets.

Table 2: Average Computational Time

n	**Algorithm**				
	MOG	*MOVNS*	*GMOVNS*	*MOVNS_I*	*PILS*
30	0.19	0.41	0.41	0.56	1.16
60	0.88	3.99	2.98	4.08	11.09
90	2.72	12.91	10.96	15.82	52.95
120	7.73	53.75	36.46	57.62	154.71

Table 2 shows that all algorithms presented low computational effort, i.e., obtained the sets of non-dominated solutions in an acceptable time.

(7) The five algorithms were run ONLY five times for each instance. This is not enough to get meaningful results.

The choice of running the five algorithms only five times for each instance was based in the following papers:

- Arroyo JEC, Ottoni RS, Oliveira AP. Multi-objective Variable Neighborhood Search Algorithms for a Single Machine Scheduling Problem with Distinct Due Windows. Electronic Notes in Theoretical Computer Science 2011;281:5-19.

- Ottoni RS, Arroyo JEC, Santos A G. Algoritmo VNS Multiobjetivo para um Problema de Programação de Tarefas em uma Máquina com Janelas de Entrega. In: Proceedings of the 18th Simpósio Brasileiro de Pesquisa Operacional, Ubatuba, Brasil; 2011.

This choice was based on papers above where their authors had published several works regarding multi-objective metaheuristic methods using this procedure.

(8) For table 2 and table 3, the authors need to analyze and explain why MOVNS_I did not attain lower average values to the set of instances with n = 60, not just report the results.

Regarding this comment, the analysis below was inserted in the manuscript (page 16, paragraph 6, and page 17, paragraph 1).

As presented in Section 5.2, the distance metrics measures the proximity between the solutions of a set D_i and the solutions of set *Ref*. Therefore, the higher the percentage of solutions of D_i in the *Ref* set, the lower tends to be the values of the distance metrics. The values of the distance metrics tend to be smaller, but those values also depend of the distance between D_i solutions and solutions belonging to *Ref* set obtained by other algorithms. For each group of 40 instances of size n, Table 5 shows average percentages of solutions obtained by the *MOVNS_I* and *PILS* algorithms which are part of *Ref* set.

Table 5: Average Percentages of Solutions of the *MOVNS_I* and *PILS* in the *Ref* Set

n	**Algorithm**		**Difference**
	MOVNS_I	*PILS*	
30	56.75	59.79	3.04
60	27.08	55.48	28.48
90	36.70	38.64	1.94
120	42.11	44.16	2.05

Table 5 shows that algorithms had presented very close values for the average percentage except for the set with n = 60. In this case the percentage difference was 28.48%. For the groups of instances in which the difference between the average percentages was small, the *MOVNS_I* algorithm had presented better results for D_{av} and D_{max}, even the *PILS* showing higher percentage. However, when the difference between these average percentages was large, as in the case of the instances set with n = 60, better values for the distances was obtained by the *PILS*.

(9) In Section 5.4, the authors selected ANOVA to verify and compare solutions of algorithms. This approach requires some strong assumptions. There are other statistical approaches, such as response surface methods, that can be used in this regard.

There are other methods could be used, but due to statistical knowledge of the authors, we opted for use of the ANOVA. The using of the Minitab® computational package has assisted in the obtaining and analysis of results.

(10) In Section 5.4, the same equation (1), (3), (5), and (7) were used in different tests. This is really confusing. What is u?

The equations (1), (3), (5) and (7) represent the same hypothesis in different tests, i.e., each equation is related to a different multi-objective performance measures.

The Statistical Analysis section was reviewed and this equation was presented only once (page 18, paragraphs 8-9), facilitating the understanding of tests.

In equations (1), (3), (5) and (7), "μ" represent the population means. The definition of "μ" is found in the manuscript (page 18, paragraph 7, line 3).

(11) The authors definitely need to provide a section to discuss the implications and differences of the computational results.

It was inserted in the manuscript (page 18, paragraphs 2-5) a section that discusses implications and differences of the computational results. The section is below.

5.3.1. Analysis of the Results

Based on the average values of the computational time spent by each algorithm to obtain the non-dominated solutions sets, we can see that all the algorithms were computationally eficeintes, obtaining sets of solutions in an acceptable time. For all the instances sets, the *MOG* and *PILS* algorithms had presented the lowest and highest average computational time, respectively.

Results attained from the computational experiments, showed that the *GMOVNS* algorithm had best performance. The *GMOVNS* has generated better results for two of the four multi-objective performance measures assessed: hypervolume indicator and *epsilon*. This means that the *GMOVNS* algorithm produces a better coverage for the Pareto-optimal front and that the non-dominated solutions generated by this algorithm are closer to the *Ref* set.

Regarding the distance metrics, in general, the *MOVNS_I* algorithm has obtained the lowest average values for this metric. Therefore, the *MOVNS_I* has achieved better distributed solutions throughout the *Ref* set.

For all the instances sets, the *PILS* algorithm had obtained the better results for the error ratio. The *PILS* had presented, on average, the higher percentage of solutions belonging to the *Ref* set.

(12) In Section 6, the authors mentioned that "Statistical experiments were conducted and have revealed that there is a significant difference between the proposed algorithms concerning the distance, epsilon, and error ratio metrics", but this is inconsistent with the statistical result in Section 5.4, where the authors said "there are enough statistical evidences to conclude that the average values regarding the hypervolume indicator equal within all algorithms".

The statistical experiments have revealed that there is a significant difference between some proposed algorithms regarding to three multi-objective performance measures assessed:

- Distance metrics:

- Average: *MOG* × *MOVNS_I* and *MOG* × *PILS*;

- Maximum: *MOG* × *MOVNS*, *MOG* × *MOVNS_I* and *MOG* × *PILS*;

- *Epsilon*: *GMOVNS* × *MOVNS* and *GMOVNS* × *MOVNS_I*;

- Error ratio *MOG* × *MOVNS*, *MOG* × *GMOVNS*, *MOG* × *MOVNS_I*, *MOG* × *PILS*, *MOVNS* × *PILS* and *GMOVNS* × *PILS*.

According to statistical experiments, the algorithms only showed no significant difference with respect to hypervolume indicator, on a 5% significance level.

The multi-objective performance measures are used to quantitatively compare the algorithms with respect to characteristics of the sets of non-dominated solutions obtained by them. Each metric compare a different characteristic.

Regarding the distance metrics, for example, characteristics of the solutions sets assessed are the proximity between the solutions of the sets and the solutions of the *Ref* set and the distribution of the solutions throughout the set. In this case, algorithms pairs *MOG* × *MOVNS_I* and *MOG* × *PILS* showed different characteristics to the average distance and algorithms pairs *MOG* × *MOVNS*, *MOG* × *MOVNS_I* and *MOG* × *PILS* to the maximum distance.

Therefore, the algorithms can present or not differences in the characteristics of their solutions sets. The Statistical Analysis was performed to verify this.

Also, these results are based on statistical analysis considering a 5% significance level. If another significance level smaller than *p-value* is used in the tests, then other algorithms pairs could present significant difference with respect to assessed multi-objective performance measures.