Gest. Prod., São Carlos, v. 23, n. 1, p. 132-145, 2016
http://dx.doi.org/10.1590/0104-530X1432-14

A multi-objective approach to the scheduling problem with workers allocation

Uma abordagem multiobjetivo para o problema de sequenciamento e alocação de trabalhadores

Guido Pantuza Júnior[1]

Abstract: This paper addresses the scheduling problem with workers allocation (SPWA). In SPWA, the objective is to minimize the number of workers and the total time taken to perform all tasks (makespan). To this end, we propose the use of two different mathematical programming models and a VNS-based multi-objective heuristic. As the objectives are conflicting, the proposed methods generate a set of efficient solutions, and the manager chooses which solution should be adopted. The proposed methods achieved satisfactory results for the SPWA resolution, showing that it is possible to use a limited staff and finish all tasks in a timely manner, thereby utilizing the resources of a firm optimally.

Keywords: Multi-objective optimization. Epsilon-restricted. Method of weighted sums. VNS-Multi-objective. SPWA.

Resumo: *O presente trabalho trata do problema de sequenciamento e alocação de trabalhadores (SPWA). No SPWA, objetiva-se minimizar o número de trabalhadores e o tempo total gasto para executar todas as tarefas (makespan). Para tanto, propõem-se o uso de dois modelos diferentes de programação matemática e uma heurística VNS-Multiobjetivo baseada no método heurístico VNS. Como os objetivos são conflitantes entre si, os métodos propostos geram um conjunto de soluções eficientes, cabendo ao gestor escolher qual solução deve ser adotada. Os métodos propostos obtiveram resultados satisfatórios para a resolução do SPWA, demostrando que é possível utilizar um número reduzido de funcionários e terminar todas as tarefas em tempo hábil, utilizando, assim, os recursos de uma empresa de forma otimizada.*

Palavras-chave: *Otimização multiobjetivo. Épsilon-restrito. Método das somas ponderadas. VNS-Multiobjetivo. SPWA.*

1 Introduction

The world market is more and more competitive and demanding. Besides that, the world crisis and the threat of a recession in Europe put in danger the financial health of the companies. In this scenario, the companies need to reduce their productive costs. A way to reduce that cost is optimizing the production factors. Between a diversity of production factors, one of those, which has, a larger impact on the production costs is the workforce. This is motivated by the recent salary increases higher than the inflation rate, added to the social charges and benefits. So, the companies look for a reduction on the number of workers without affecting the delivering deadlines.

This problem faced by the companies is known as scheduling problem with worker allocation – SPWA. The problem consists in allocate the jobs to the workers in a company, minimizing the instant of ending of the last job executed (makespan) and the number of workers on it.

In general, most of the works found in the literature treats this problem as mono-objective. That means the multi-objective evaluation function is turned in a mono-objective evaluation function. That transformation, in general, is made assigning different weights to different objectives. Works as Abensur's (2012) uses this approach. He used the goal programming to treat of a multi-objective optimization problem. This way, he converted the multiple objectives in a single one.

But, the objectives are conflicting with each other, so, doesn't exist a single solution which optimize them all at the same time. In fact, the lower the number of workers is, bigger is the time needed to execute all the jobs, so as the reverse. So, adopting the mono-objective resolution method, we have

[1] Instituto Federal de Minas Gerais – IFMG, Av. Minas Gerais, 5391, Ouro Verde, Governador Valadares, MG, Brasil, e-mail: gpantuza@gmail.com

Received Apr. 30, 2014 - Accepted Oct. 13, 2014

Financial support: FAPEMIG and IFMG.

only one solution, which privileges an objective over the other.

Besides, in practice, the managers responsible for the workers designation and sequencing of jobs have to deal, daily, with changes in the number of workers available and of jobs which need to be executed. So, the managers need a flexible solution. It has to be capable to absorb the sudden changes on the parameters in a quick way, keeping the system optimized.

The multi-objective optimization looks for a set of efficient solutions, what makes the system more flexible, once it shows different solutions. The manager, when solves the problem, adopting this method, has at its service a lot of solutions, which one using a different number of workers. The available solutions, obtained through the multi-objective method, are closer to the reality, making the manager's job faster and more flexible. So, the manager can answer to an absence of a worker without need to solve the model again, saving time.

To the resolution of the multi-objective problems, the methods vary between exact and heuristic. To the exact approach, has certain of an optimum solution, but, to complex problems, usually the time spent is bigger than the available time for the decision-making. The using of heuristic methods is recommended when is searched for a good solution in a timely manner. However, this method doesn't guarantee the optimality of the solution for the problem. Starting from those remarks made in the industrial and services sector, primarily in small and medium-sized enterprises we observe that, usually, those companies have a number of workers and jobs reduced. So, this work proposes a resolution of multi-objective approach for the SPWA, using exact and heuristic methods. These should generate flexible solutions which may be implemented.

Between a diversity of exact methods for the resolution of a multi-objective problem, this work focus on the models of Mathematical Programming, using the variable weight method and the epsilon-restricted (ε-restricted) method. According to Coello et al. (2002), these methods are the most used. Besides, according to Tan et al. (2009), the SPWA is a NP-hard problem, so, for problems with bigger dimensions, is not possible to find the global optimum solution in a computational timely manner. So, a heuristic multi-objective model, based on the Variable Neighborhood Search – VNS algorithm is also proposed.

The present work is organized as follows. In section 2, it describes the problem studied. In section 3, the theoretical assumption. In section 4, we present the models of Mathematical Programming used. In section 5 is presented the VNS-Multi-objective algorithm proposed. The presentation of the test-problems and the results are done in the section 6. The conclusion is showed in the section 7.

2 The scheduling problem with worker allocation

This problem is composed by a set of *Workers*, and a set of *Jobs*. It consists in allocate the jobs to the worker and propose an execution sequence, respecting the qualification restrictions. It means a worker can only execute a job if he is qualified. Besides, each worker, to execute a same job, needs different times according to their skills. The objectives of the SPWA are to minimize the instant of ending of the last job executed (makespan) and minimize the total number of workers on it.

In this work, we consider the following restrictions: every job should be executed. We don't consider the workers transportation times or the time spent waiting in queue. The horizon on workers allocation planning is fixed. Each job is executed by only a single worker. The overworking is not considered, it means, we do not consider the workers will be overloaded with excess of work. Every job has the same amount of effort. The workers, who are used, have a minimum and maximum utilization rate. All workers have the same cost (salary). Each employee has a different skill, it means, each one execute the same job with a different time.

The SPWA can be considered as special case of the job shop scheduling problem. Such problem, in its general manner, is characterized by the allocation of jobs to same kind of resource needed for its execution. This resource, usually, is a work station or some kind of machine.

In literature, a diversity of works treats about the job shop scheduling problem, such as Silva & Rentes (2012), Tavares Neto & Godinho Filho (2013) and Mello & Ferreira (2014).

The work of Silva & Rentes (2012) shows off a new optimizing model of the job shop problem by a layout optimizing and applies in some companies of the metal mechanical sector. The goal of the model consists in develop alternatives that would be harmonic with concepts and principles of the lean production philosophy.

Tavares Neto & Godinho Filho (2013) utilizes a method that consists in two stages to deal with the job shop in an ambient of a machine with the possibility of outsourcing. On the first stage, they propose the jobs to be sequenced, using the SPT (Shortest Processing Time) rule. On the second stage is proposed an algorithm based on ACO (Ant Colony Optimization).

Mello & Ferreira (2014) deals with the job shop problem with the minimizing of the makespan. They use models of computer simulation of a manufacture system with two different scenarios. On the first one, they do not consider the utilization of alternative machines. On the second one, they adopt the insertion of alternative machines to execute the same jobs.

According to Osawa & Ida (2007), the main difference between SPWA and the classic job shop is the utilization of the workforce resource, considering different levels of qualification and a low rate of utilization. This low rate of qualification should be assigned to the decrease of the yield due to the tiredness, the environmental conditions of the workplace (temperature and lighting, for example), in addition to the labour law that guarantee the workers' right to rest during the working time.

In literature we found some works which deal with the SPWA. Iima & Sannomiya (2001) proposed an heuristic to the resolution of the SPWA, based on the Genetic Algorithm called Module Type Genetic Algorithm (MTGA). Osawa & Ida (2005) utilized a Genetic Algorithm, but they proposed a new selection method of the survivor population. Osawa & Ida (2007) adopted the algorithm proposed by Osawa & Ida (2005). However, they considered that each worker owns a different skill level for each used machine.

Every work previously quoted adopted mono-objective approaches. However, we found in the literature, a diversity of works which adopted multi-objective methods, most of them using metaheuristics to solve the job shop problem.

Ishibushi & Murata (1998) proposed a local search Genetic Algorithm. It consists in inserting non-dominated solutions on the population (set of solutions) from the proposed method, looking for a more adapted population. Suresh & Mohanasundaram (2004) proposed a multi-objective algorithm which they called as Pareto Archived Simulated Annealing (PASA). In this algorithm was used a new method of perturbation of the solutions, to find neighborhood solutions, called Segment-Randon Service (SRI).

Garcia et al. (2004) adopt an heuristic and an exact method. The heuristic method is based on a multi-objective evolutionary algorithm. The mathematical programming model is non-linear and the used resolution method is epsilon-restricted. Their results showed the both techniques reached closed solutions and are capable of avoiding local optimum. Iima (2005) proposed a Genetic Algorithm with a new way to select the most adapted population and a new mutation operator, based on probabilities distribution. Lei & Wu (2005) used a multi-objective evolutionary algorithm (CMOEA). This is based on the crowding measure. It uses the crowding distance to adjust the external population and attribute different fitness for the people. They consider two objectives: minimize the makespan and the total delay of the delivering deadline.

Xia & Wu (2005) used an evolutionary algorithm based on the particle swarm optimization. Such algorithm imitates the flying birds behavior and their ways of changing information. It combines such algorithm with the local search, using the Simulated Annealing algorithm. Quian et al. (2006), developed an algorithm based on the Memetic Algorithm, which is based in differential evolution called MADE. First, in such algorithm, a rule is used to convert the problem in a way that the differential evolution can be applied. In a second place, the mechanism of parallel evolution is applied to explore the neighborhood, executing a local search. Besides, the Pareto Dominance concept is used for the solutions' selection. They consider, as a goal, minimize the maximum delay.

Li et al. (2010) developed an hybrid multi-objective method which combines two metaheuristics: Tabu Search and Variable Neighborhood Search (VNS). This work proposes an Hybrid Tabu Search Algorithm (HTSA) considering three minimizing goals: the maximum conclusion time (makespan); the total time of work of the equipment and the work load of the machine. A neighborhood structure proposed combine two adaptations criteria, one for the local search method, other for the machines selection. Following, a job designation method is executed. Besides, a VNS algorithm is used adopting three neighborhood structures.

Ruiz et al. (2012) have based in a multi-objective genetic algorithm called *Elitist Non-Dominated Sorting Genetic Algorithm* (NSGA-II). They consider three goals: minimize the makespan, the energy costs and work accidents.

3 Theoretical assumptions

The SPWA is composed by two conflicting objectives (minimize the number of workers and the instant of ending of the last executed job). So, it does not exist a single solution which optimizes them all at the same time. As a result, to find viable solutions that optimize simultaneously all he objectives is the biggest challenge of the multi-objective optimization.

For authors like Zitzler (1999), Fonseca & Fleming (1995) and Arroyo (2002), for a resolution of this kind of problem we should search for a set of efficient solutions. In this case, the decision taking will be a responsibility of the manager, who can choose the solution which best fits to the needing among the efficient solutions. This criterion will be used by the responsible, or manager, for the decision taking, it means, he can consider among different conflictive solutions.

The set of efficient solutions is also known as Pareto-optimum. For definition, a set *S* of solutions is Pareto-optimum if does not exist another set *S** of viable solutions that can improve some objective without causing a worsens in, at least, another objective. In other words, a solution *S* belongs to the Pareto-optimum solutions set if does not exist a solution *S** which dominates *S*.

Considering a minimization problem, we have:

- *S* dominates *S** if and only if $S \leq S^*$ for all purposes;
- *S* and *S** are indifferent or have the same dominance degree if and only if *S* do not dominates *S** and *S** do not dominate *S*.

For the multi-objective problems, the conventional mono-objective optimization methods are not efficient (COELLO et al., 2002). Therefore, the search for new optimization methods which can win the great challenge of this kind of problem became necessary. A way to win this challenge is the utilization of exact multi-objective optimization methods. Those methods emerged from the need to find solutions with priorities, or weights, associated to the objectives. Among the exact classic methods used to solve that range of problems, we highlight: weighted sum method and epsilon-restricted (ε-restricted) method.

The weighted sum method consists in a transformation of a multi-objective problem in a mono-objective problem through the attribution of weights for each objective. To reach the Pareto-optimum solutions, this problem must be solved iteratively. It means, for each iteration, new weights are attributed to different goals, being, usually, the sum of the weights equal 1.

According to Arroyo (2002), the main disadvantage of that method is it can't generate all the Pareto-optimum solutions when the objective space is not convex. The goal programming method, similar to the weighted sum method, also consists in the transformation of the multi-objective problem to a mono-objective problem. This happens through the attribution of weights for each goal. For that method, we consider that each goal has a different importance at the optimization represented through the weights. The bigger is the goal importance; bigger will be its weight.

The exact epsilon-restricted method was initially proposed by Ritzel et al. (1994). It consists in the optimization of the most important objective, represented for Equation 1, subjecting itself to the restrictions of other objectives, represented for Equation 2.

Considering, in a minimizing problem, f_1 as the most important objective, we have:

minimize

$$f_1(x) \tag{1}$$

Subjected to:

$$f_i(x) \leq \varepsilon_i \quad i = 2, 3, ..., q \tag{2}$$

where ε_i is the superior limit of the objective *i* and *q* is the number of objectives.

To build a Pareto-optimum set, even when the objective space is not convex, must vary only the superior limit. But, if this is not the proper limit, the subset of possible obtained solutions may be empty, it means, there is no viable solution.

Another way to solve multi-objective problems is through heuristic methods. Among a variety of works related to the multi-objective heuristic approach, we highlight as the most used the Genetic Algorithms, which are based on the evolution theory. On the multi-objective genetic algorithms, at each generation, or iteration, there is a set of individuals, or father-solutions. To create a new population (offspring-solutions), the genetic operators (so as crossover and mutation) are applied on those father-solutions. This way, we obtain a new population of solutions formed by father and offspring-solutions. At the end, of each iteration, the most able individuals survive and the rest is discarded. Among a variety of Multi-Objective Genetic Algorithm, we highlight: VEGA, MOGA, NPGA, SPEA and NSGA.

On the Vector Evaluated Genetic Algorithm (VEGA), proposed by Schaffer (1985), at each generation, a set of individuals that overcome the others according to one of the *n* goals is selected, until that *n* groups be formed. Then, the *n* groups are mixed together and the genetic operators are applied to create the following generation.

On the Multi-objective Genetic Algorithm (MOGA), proposed by Fonseca & Fleming (1993), each individual *i* is qualified in a level according to the number of individuals who that individual *i* dominates. Every non-dominated individuals are qualified at level 1. The fitness of each individual is attributed according to an interpolation between the best and the worst level. The final fitness attributed to all individuals from a same level is the same and equal to the fitness measure of the own level. This way, every individual on the same level are indifferent between themselves.

On the Niche Pareto Genetic Algorithm (NPGA), proposed by Horn et al. (1994), the selection of the individuals is given through a tournament based on the Pareto dominance concept. Two individuals are selected and compared with a subset of the solutions population, being selected to the next generation those not dominated.

On the Non-dominated Sorting Genetic Algorithm (NSGA), proposed by Srivivas & Deb (1995), the individuals are qualified in levels according to their dominance degree, such as the previous algorithms. However, it is attributed a fitness value to each individual, according to their level and their distance to the other solutions of the same level, the so called crowd distance. The selection is made through tournament using the fitness value until every vacancies to the next generation be filled.

On the Strength Pareto Evolutionary Algorithm (SPEA), proposed by Zitzler (1999), is used the selection based on the dominance relation to evaluate and select the solutions. To evaluate this dominance relation and qualify the individuals in dominance levels, the SPEA uses an additional set of the population. But, unlike the previous algorithms, which discard the non-selected individuals, it uses the non-dominated individuals of the population from the previous generation to determine the fitness of the individuals of the current population.

Despite being the most utilized, not always the Genetic Algorithms are the recommended ones (GUIMARÃES et al., 2007). Besides these, we also find other heuristics to multi-objective problems, such as the Multi-Objective Tabu Search procedures proposed by Arroyo (2002). Also the Tabu Search heuristics, proposed, initially, to mono-objective problems, other ones can also be used such as the VNS and *Variable Neighborhood Descent* (VND) heuristic methods.

The VNS heuristic method, exemplified by the Chart 1, proposed by Mladenovic & Hansen (1997), is a method that consists in explores the space of solutions by systematic changes of the neighborhood structures. In contrast to other metaheuristic based in local search methods, the VNS method does not follow a path. It explores different neighborhood of the current solution and focus the search to a new solution if and only if an improvement move is realized.

The VNS also includes a local search procedure to be applied on the current solution. In this work, we adopted the VND method, exemplified by the Chart 2 to make a local search, also used by the original VNS.

The VND is a technique used for the refining of initial solutions. This is made through an analysis of the feasible solutions region through systematic changes on the neighborhood structures of current solutions. This method accepts only the improvement solutions of the current solution, returning to the first structure when a better solution is found.

Among the countless VNS applied to the multi-objective optimization, we highlight Geiger (2004), so called MOVNS. In this procedure, the neighborhood structure and the solution to be explored are chosen randomly, changing at each iteration. A solution between the non-dominated is selected. From this solution, new neighborhood solutions are generated.

In Ottoni et al. (2011), the authors deal with the job shop scheduling problem, which *n* jobs should be processed in a single machine that can process a job at time. They proposed to incorporate a new intensification method to the MOVNS algorithm,

Chart 1. VNS Heuristic.

```
Procedure VNS (f(.), N(.), r, s);
1   Being $s_0$ a starting solution;
2   Being r the number of different neighborhood
    structures;
3   $s \leftarrow s_0$ ;
4   While (non-satisfied stopping criterion) do
5       $k \leftarrow 1$;
6       While ($k \le$ r) do
7           Generate any neighbor $s' \in N^{(K)}(s)$;
8           $s'' \leftarrow$ LocalSearch ($s'$);
9           if ($f(s'') < f(s)$) them
10          $s \leftarrow s''$;
11          $k \leftarrow 1$;
12          else
13              $k \leftarrow k + 1$;
14          End-if
15      End-while;
16  End-while;
17  Return s;
End VNS;
```

Chart 2. VND Heuristic.

```
Procedure VND (f(.), N(.), r, s);
1   Being r the number of different neighborhood
    structures;
2   $k \leftarrow 1$ ;
3   While ($k \le r$) do
4       Find a neighbor $s' \in N^{(K)}(s)$;
5       if ($f(s') < f(s)$) them
6           $s \leftarrow s'$;
7           $k \leftarrow 1$;
8       else
9           $k \leftarrow k + 1$;
10      End-if
11  End-while;
12  Return s;
End VND;
```

proposed by Geiger (2004). This intensification consists in a perturbation on the solution, generating new solutions.

Arroyo et al. (2011) also deal with the job shop scheduling problem, considering a time window machine with setup times. The authors consider two goals: the first one is to minimize the jobs delay or advance and the second is to minimize the total flow. To solve the problem, was proposed a new MOVNS algorithm combined with a perturbation process, different of the proposed by Ottoni et al. (2011).

4 Mathematical programming models

4.1 *Epsilon*-restricted

To solve the SPWA using the epsilon-restricted method, we adopted, as the main objective, the instant of ending of the last executed job. The number of workers is considered a secondary goal, so, at each execution of the proposed mathematical model, the maximum number of workers who can be used (ε) is reduced. For this model, we consider the following entry parameters.

Job: Set of jobs.

Worker: Set of workers.

T_{ij}: Time needed for the worker *j* to finish the job *i*.

ε: Maximum amount of workers.

H: Planning horizon.

Tu_i: Maximum utilization rate of the worker *i*.

Tl_i: Minimum utilization rate of the worker *i*.

The following decision variable are:

σ: Instant of ending of the last executed job.

$$x_{ij}:\begin{cases}1 \text{ If the workeriexecute the job } j.\\ 0 \text{ Otherwise.}\end{cases}$$

$$y_i:\begin{cases}1 \text{ If the worker } i \text{ is used.}\\ 0 \text{ Otherwise.}\end{cases}$$

The mathematical programming model proposed, relative to the SPWA, is presented by the Equations 3 to 10:

Objective function:

The Equation 3 looks to minimize the instant of ending of the last job.

$$\min\ f = \sigma \qquad (3)$$

Subjected to restrictions:

The Equation 4 ensure that every job *j* will be executed only once for a single *i* worker.

$$\sum_{i \in \text{Worker}} x_{ij} = 1 \qquad \forall\, j \in Job \qquad (4)$$

The Equation 5 stipulates the maximum amount of workers *i* which can be used. The ε value is reduced at each iteration of the proposed model.

$$\sum_{i \in \text{Worker}} y_i \le \varepsilon \qquad (5)$$

The Equation 6 determine the instant of ending of the last job *j* executed by the operator *i*.

$$\sum_{j \in \text{Job}} T_{ij} x_{ij} \le \sigma \qquad \forall\, i \in Worker \qquad (6)$$

The Equation 7 define if the worker *i* are executing some job *j*.

$$\frac{\sum_{j \in Job} x_{ij}}{|\text{Job}|} \le y_i \qquad \forall\, i \in Worker \qquad (7)$$

The Equations 8 and 9 define the gap for the using rate of the worker *i* who are being used.

$$\frac{\sum_{j \in Job} x_{ij} T_{ij}}{H} \le y_i Tu_i \qquad \forall\, i \in Worker \qquad (8)$$

$$\frac{\sum_{j \in Job} x_{ij} T_{ij}}{H} \le y_i Tl_i \qquad \forall\, i \in Worker \qquad (9)$$

The Equations 10 to 12 assure the domain of the decision variables.

$$x_{ij} \in \{0,1\} \qquad \forall\, i \in Worker \text{ e } \forall\, j \in Job \qquad (10)$$

$$y_i \in \{0,1\} \qquad \forall\, i \in Worker \qquad (11)$$

$$\sigma \in \mathbb{R}^+ \qquad (12)$$

4.2 Weighted sum method

To solve the SPWA using the weighted sum method, we adopted, as a goal, the instant of ending of the last executed job (makespan) and the number of workers used on it. On this method, for each iteration k, the value of the penalty (w_k) of the objective function is changed at each iteration. Initially, it assumes the value 1 and for each iteration its value is decreased according to the Equation 13. In this equation, for each iteration, the new w_k value is equal to the value of the previous iteration penalty, w_{k-1}, minus the value of the division of 1 for the number of adopted solutions. This way, for each iteration, we change the space of searching, privileging some goal.

$$w_k = w_{k-1} - \frac{1}{NSol} \quad (13)$$

Being:
w_k: Penalty of the objective function on the iteration k.
$NSol$: Number of solutions.

The decision variables and parameters are the same as the previous model, except for the withdrawal of the (maximum amount of workers) parameter and for the inclusion of the parameter w (objective penalty). The proposed mathematical programming model for the weighted sum method is similar to the method on the 4.1 section, but they are different for the substitution of the Equation 3, regarding to the objective function; for the Equation 14 and for the exclusion of the restriction showed on the Equation 5.

Objective function

The Equation 14 looks to minimize the instant of ending of the last job.

$$\min\ f = w_k \sigma + (1 - w_k) \sum_{i \in \text{Worker}} y_i \quad (14)$$

5 Proposed algorithm

To a variety of problems, find the global optimum solution of big dimension problems can be impracticable. To problems of this nature, like SPWA, the use of exact methods become a lot restrict. This is the reason for what uncountable works concentrates efforts in using heuristics to solve problems this level of complexity. Heuristics can be defined as a technique which search for good solutions, it means, close to the global optimum.

So, we also presented a multi-objective heuristic model based on the VNS algorithm, exemplified by the Chart 1 on the section 3.

The proposed algorithm, called Multi-objective-VNS (VNSM), exemplified by the Chart 3, starts its execution from a set of S_0 initial solutions. This set is generated through a partially greed procedure, using a binary contest. The number of solutions (*NSol)* of the S_0 set was defined in an empirical way, considering the complexity of the problem.

After this step, each solution $s_l \in S_0$ is evaluated according to an evaluation function with variable weights. The weight value of each objective vary from each iteration, this way, from each iteration, an objective owns bigger or lower importance to determine a solution.

Following, the first solution, s_1, of the set S_0 is selected, and selects a s_1' neighbor on the neighborhood $N^{(1)}(s_1)$ of the current solution s_1. This neighbor, s_1', is subjected to a local search procedure, returning the s_1" solution. If the local-optimum solution, s_1", is better than the current s_1, the search continues in s_1", restarting from the first $N^{(1)}(s_1")$ neighborhood structure.

Otherwise, the search continues from the next $N^{(K+1)}(s_1)$ neighborhood structure. This step is repeated until we have a maximum number of iterations without an improvement, *It_SM,* empirically defined. At the end of this stage, the better found solution is added to the set of final solutions *S*.

After the end of the previous stage, the procedure return to its begins, selection the next solution s_1 of the set S_0, and all the previous procedure repeats. The algorithm is repeated for all solutions s_i of the initial solutions set $S_{0.}$

5.1 Representing a solution

A solution can be represented by a (s_l) [*Worker*×(*Job*+1)] matrix. This way, the moves of the neighborhood structures become more simple and natural. So, the algorithm becomes less complex and the evaluation of the solutions become easier. The set of *S* solutions is formed by the union of all *NSol* (number of solutions) matrix s_l.

The Table 1 exemplifies a possible s_l solution. The first column shows the workers. The "Useful" column shows if the worker i realizes a job j. The columns "Jobs" shows the jobs allocated for

Chart 3. Multiobjective-VNS heuristic.

```
Procedure VNSM ($f(.)$, $N(.)$, $r$, $S_0$)
1  Being $S_0$ a set of initial solutions;
2  Being $r$ the number of different neighborhood structures;
3  for ($l < NSol$) do
4      $s \leftarrow s_l \in S_0$;
5      $it \leftarrow 0$;
6      While ($it < It\_SM$) do
7          k ← 1;
8          While ($k \leq r$) do
9              Generates any neighbor $s_l' \in N^{(K)}(s_l)$;
10             $s" \leftarrow$ LocalSearch ($s'$);
11             if ($f(s_l") < f(s_l)$) them
12                 $s_l \leftarrow s_l"$;
13                 $k \leftarrow 1$;
14             else
15                 $k \leftarrow k + 1$;
16                 $it \leftarrow it+1$;
17             End-if
18         End-while;
19     End-while;
20     $s_l \in S$;
21 End-for
22 Return $S$;
End VNSM
```

Table 1. Representation of a s_l solution.

		Useful	Jobs				
Worker	**1**	1	1	3	4	0	0
	2	0	0	0	0	0	0
	3	0	0	0	0	0	0
	4	1	2	5	0	0	0

each worker and its respective sequence. On the example of the Table 1, the value 1 of the first line (Worker 1) and "Useful" column shows the worker 1 is being used. The values of the other columns show the worker 1 will execute the jobs 1, 2 and 4, this order. The value 0 on the second line (worker 2), column 'Useful" shows the worker 2 is idle. Soon it will not execute any job. For the worker 4 we know he will execute jobs 2 and 5, this order.

5.2 Generation of the initial solution

The initial solution generates *NSol* matrix s_l through a partially greed procedure using a binary tournament. For each matrix solution s_l, at each iteration of the procedure, a job *j* is selected. Besides, two workers are also chosen randomly. The worker, who finish the *j* selected job in a smaller time, is chosen. For example, for the job 1 is chosen the workers number 2 and 4. The worker 2 finishes the job 1 in three hours and the worker 4 in two hours. In this case, the job *j* will be executed by the worker 4.

This procedure is executed until all jobs be attributed to some worker *i* for all the matrix $s_l \in S$.

5.3 Evaluation of a solution

A $s_l \in S$ solution is evaluated following the evaluation function $f(s_l)$. This function tries to minimize the number of workers and the makespan through penalties. The evaluation function $f(s_l)$ is exemplified by the Equation 15.

$$\min f(s_l) = \frac{w_l}{\Delta^{\sigma}}\sigma(s_l) + \left(\frac{1-w_l}{\Delta^{\tau}}\right)\tau(s_l) \qquad \forall s_l \in S \quad (15)$$

Being:
w_l: Penalty of the solution s_l.
σ: makespan.
τ: Number of workers used.
Δ^{σ}: Correction factor for the σ parameter.
Δ^{τ}: Correction factor for the τ parameter.

The value of w_l is changed at each iteration of the VNSM algorithm, for each solution s_l evaluated. Initially, it assumes the value 1, and at each iteration, for each solution s_l, its values decreased following the Equation 16. In this equation, at each iteration, a new s_l value is equal to the penalty value of the previous iteration, w_{l-1}, minus the value of the division of 1 for the number of solutions adopted. This way, at each iteration, changes the space of searching, privileging some objective.

$$w_l = w_{l-1} - \frac{1}{NSol} \qquad (16)$$

In the evaluation function, Equation 15, we also adopted correction factor Δ. This factor guarantees the decision variable τ e σ to be inside the same order of magnitude.

5.4 Neighborhood structures

To explore the space of problem's solutions, it means, to find a solution s_l', neighborhood of s_l, we used two movies, presented as follows:

Job Relocation Movement, $N^{RT}(s_l)$: this movement consists in picking up randomly a worker *i* who are being used. Following, the last job *j* is selected and relocated to another worker randomly picked. In Table 2, the worker 1 is selected and the job 4 is relocated to the worker 4.

Worker Movement, $N^{T}(s_l)$: this movement consists in picking up randomly a worker *i* who are being used and relocate all of your jobs to the other workers, randomly. In Table 3, the worker 1 is selected, the job 3 is relocated to the worker 4 and the job 1 is relocated to the worker 2.

5.5 Local search

The local search is applied to all solutions $s_l \in S$. It consists in applying the VND (see Chart 3). Initially, we consider a set of $r = 2$, distinct neighborhood, each one defined by one of the kind of movements defined in section 5.4. In the sequence, starting from the solution s_l', certain amount of neighbors s_l" of the first neighborhood are analyzed.

Following, we go to the second neighborhood structure. Again, a determined amount of neighbors is analyzed. The s_l" neighbor who shows the biggest improvement related to the s_l', according the evaluation function on the section 5.3, is chosen and we finish the local search.

Table 2. Job Relocation Movement, $N^{RT}(s_l)$.

		Useful			Jobs		
Worker	**1**	1	1	3	4	0	0
	2	0	0	0	0	0	0
	3	0	0	0	0	0	0
	4	1	2	5	0	0	0
		Useful			**Jobs**		
Worker	**1**	1	1	3	0	0	0
	2	0	0	0	0	0	0
	3	0	0	0	0	0	0
	4	1	2	5	4	0	0

Table 3. Worker Movement, $N^{T}(s_l)$.

		Useful			Jobs		
Worker	**1**	1	1	3	0	0	0
	2	1	4	0	0	0	0
	3	0	0	0	0	0	0
	4	1	2	5	0	0	0
		Useful			**Jobs**		
Worker	**1**	0	0	0	0	0	0
	2	1	4	1	0	0	0
	3	0	0	0	0	0	0
	4	1	2	5	3	0	0

6 Results

To test the proposed models we used 4 test-problems. They can be found in http://www.4shared.com/zip/m7xmUfpi/Instance_-_SPWA.html. They are different between themselves by the number of workers and jobs. The Table 4 shows us some characteristics of the problems. The columns ***#Worker*** and ***#Job*** show, respectively, the amount of workers and the number of jobs which need to be executed.

The mathematical programming models, developed in Section 4, were implemented in LINGO 10.0 optimization app, interfacing with EXCEL 2010 spreadsheets. The VNSM algorithm proposed were developed in C language, using the *Bordland*'s C++ compiler *Builder 5.0*. All of those tests were realized in a *Pentium Core 2 Duo* PC, with, 4 GHz and 4 GB of RAM on *Windows 7* platform.

6.1 Results of the exact method

The set of solutions *S*, Pareto-optimum, found, using the ε-restricted method, for the Problem 1, is presented in Table 5. The *Epsilon* column shows the adopted value for ε, that represents the maximum amount of workers. The column *#Worker* shows the amount of used workers. The column *makespan* shows, in minutes, the instant of ending of the last job.

Table 4. Test-problems characteristics.

	#Worker	***#Job***
Problem 1	5	10
Problem 2	10	50
Problem 3	15	100
Problem 4	25	100

According to the Table 5, for the Problem 1, we can observe we obtained the same results for the two exact methods, both with 5 different solutions each. In solution 1, we adopted the value ε = 0. We will use 5 workers who will execute all jobs in 8 minutes. For the solution 5, we adopted the value ε = 4. We will use 1 worker who will execute all jobs in 43 minutes.

The Table 6 shows a set of Pareto-optimum solutions, using a ε-restricted method and the weighted sum, respectively, for the Problem 2. In this problem we adopted 10 different solutions. The value of ε varies from 0 to 9 (the maximum amount of available workers).

The Table 7 shows the results of the exact model for the Problem 3. The set of Pareto-optimum solutions is formed by 7 different solutions. The value of ε varies from 0 to 14, being reduced in two units for each solution.

Table 5. Results of the Exact Method for the Problem 1.

	Epsilon-Restricted				Weighted sum	
	Solution	**Epsilon - ε**	**#Worker**	***makespan* (min)**	**#Worker**	***makespan* (min)**
Problem 1	1	0	5	8	5	8
	2	1	4	10	4	10
	3	2	3	13	3	13
	4	3	2	20	2	20
	5	4	1	43	1	43

Table 6. Results Exact Method for Problem 2.

	Epsilon-Restricted				Weighted sum	
	Solution	**Epsilon - ε**	**#Worker**	***makespan* (min)**	**#Worker**	***makespan* (min)**
Instance 2	1	0	10	20	10	20
	2	1	9	22	9	22
	3	2	8	25	8	25
	4	3	7	28	7	28
	5	4	6	33	6	33
	6	5	5	39	5	39
	7	6	4	49	4	49
	8	7	3	65	3	65
	9	8	2	98	2	98
	10	9	1	215	1	215

Table 7. Results of the Exact Method for the Problem 3.

	Epsilon-Restricted				Weighted sum	
	Solution	***Epsilon - ε***	***#Worker***	***makespan* (min)**	***#Worker***	***makespan* (min)**
Instance 3	1	0	15	27	15	27
	2	2	13	31	13	31
	3	5	10	39	8	49
	4	7	8	49	7	58
	5	9	6	65	5	81
	6	12	3	130	4	107
	7	14	1	430	1	430

Table 8. Results of the Exact Method for the Problem 4.

	Epsilon-Restricted				Weighted sum	
	Solution	***Epsilon - ε***	***#Worker***	***makespan* (min)**	***#Worker***	***makespan* (min)**
Instance 4	1	0	25	16	25	16
	2	3	22	18	22	18
	3	6	19	22	19	22
	4	9	16	25	15	28
	5	12	13	31	11	37
	6	15	10	40	10	40
	8	18	7	57	8	49
	9	21	4	100	3	135
	10	24	1	433	1	433

The results, of the exact model for the Problem 4, are presented in Table 8. In this problem, we adopted 10 different solutions. The value of ε varies from 0 to 24, being reduced in 3 units for each solution.

For the Problem 4, unlike the other problems (1, 2 and 3), we couldn't find a set of Pareto-optimum solutions, due to the SPWA problem be NP-hard (TAN et al., 2009). It means, for problems with bigger dimensions, is not possible to find the global optimum solution in a computational timely. In this case, we adopted as the maximum execution time 900 seconds for each generated solution. For both exact methods, only for the solutions 1 and 10 (first and last solution), we could solve the problem and find a solution in less than 900 seconds, being: for the epsilon-restricted method, the execution time for the solution 1 is 41 seconds and for the solution 2, 24 seconds. For the weighted sum method, the execution time for the solution 1 is 39 seconds; for the solution 2, 28 seconds.

6.2 VNSM results

Initially, the proposed algorithm was subjected to a preliminary marathon of tests to calibrate the variety of existing parameters. Such parameters are: the number of iterations of each execution (*It_SM*), number of solutions (*NSol*), the penalty of the solution s_l (w_l) and the correction factors (Δ^{σ} e Δ^{τ}). After determining the parameters, the algorithm was subjected to a marathon of tests with 100 executions for each test problem.

The chart of the Figure 1 shows the obtained results after 100 executions of the VNSM and the exact methods. The x-axis represents the total time spent to execute all the jobs. The y-axis represents the amount of workers used. The chart shows the Best set of solutions found between the executions of the VNSM. In this chart we noticed that, for the test-problem 01, the VNS and the two exact methods (*epsilon*-restricted and weighted sum) obtained the same result. For the other test-problems, we observed the results of the exact methods were close and the VNS was able to find solutions close to the exact methods.

The Table 9 shows the time spent, in seconds, by the used methods to solve the SPWA. For the VNSM, in 'Smaller' line, we have the smaller time spent among the 100 executions. The line 'Medium' shows the average time and in the line 'Bigger' we have the larger time spent by the proposed algorithm. We can observe the exact method of the weighted

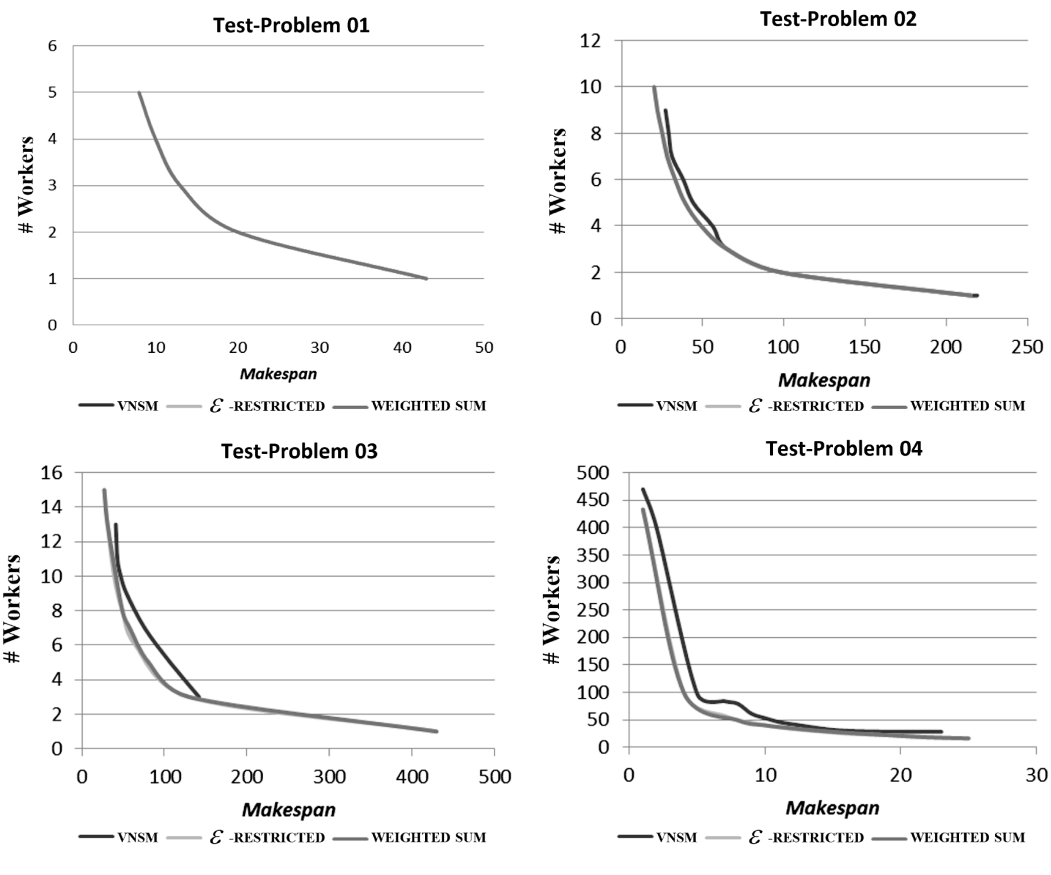

Figure 1. VNSM Results.

Table 9. Time of execution in seconds.

Method		Problem 1	Problem 2	Problem 3	Problem 4
***Epsilon*-restricted**		5	149	1972	6365
Weighted Sum		5	136	1861	5992
VNSM	**Smaller**	5.5	78.5	61.4	146.7
	Medium	5.8	101.8	146.8	211.7
	Bigger	6.8	136.5	265.7	277.3

sum, for all the problems, obtained a better or equal performance to epsilon-restricted.

7 Conclusions

This work showed two exact models of mathematical programming and a multi-objective algorithm to solve the scheduling problem with worker allocation – SPWA.

The mathematical programming models proposed were based in two classic methods of resolution of multi-objective problems: the ε-restricted multi-objective resolution method which is based in the optimization of the most important goal, subjecting to the restrictions of the other goals; and the weighted sum iterative method, which consists in the transformation of the multi-objective problem in a mono-objective one through the attribution of weights to each goal. The proposed multi-objective algorithm (VNSM) was based on the VNS heuristic combined with the local search VND algorithm.

The results show it's possible to optimize the amount of workers and maximum time of execution of the jobs by using a multi-objective approach. The both exact methods obtained similar results, seeing the weighted sum obtained a better performance about the computational time. The VNSM was capable to find solutions close to the exact methods, proving it is a good option, especially for more complex problems.

When presented a variety of solutions serving different goals, is provided to the manager alternatives for its decision-taking. This way, is possible to pick up the better solution which better adapts to the operational reality of the company.

Acknowledgement

The author is grateful to FAPEMIG and to IFMG for the financial support, indispensable for the achievement of this research project.

References

ABENSUR, E. O. Um modelo multiobjetivo de otimização aplicado ao processo de orçamento de capital. *Gestão & Produção*, v. 19, n. 4, p. 747-758, 2012. http://dx.doi.org/10.1590/S0104-530X2012000400007

ARROYO, J. E. C. *Heurísticas e metaheurísticas para otimização combinatória multiobjetivo*. 2002. 256 f. Tese (Doutorado)-Programa de Pós-Graduação em Engenharia Elétrica, Universidade Estadual de Campinas - Unicamp, Campinas, 2002.

ARROYO, J. E. C.; OTTONI, R. S.; OLIVEIRA, A. P. Multi-objective variable neighborhood search algorithms for a single machine scheduling problem with distinct due windows. *Electronic Notes in Theoretical Computer Science*, v. 281, n. 29, p. 5-19, 2011. http://dx.doi.org/10.1016/j.entcs.2011.11.022

COELLO, C. A. C; LAMONT, G. B.; VAN VELDHUIZEN, D. A. *Evolutionary algorithms for solving multi-objective problems*. Boston: Kluwer Academic Publishers, 2002. Relatório Técnico.

FONSECA, C. M.; FLEMING, P. J. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In: INTERNATIONAL CONFERENCE ON GENETIC ALGORITHMS, 5., 1993, San Mateo, USA. *Proceedings*... p. 416-423.

FONSECA, C. M.; FLEMING, P. J. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, v. 3, n. 1, p. 1-16, 1995. http://dx.doi.org/10.1162/evco.1995.3.1.1

GARCIA, J. M. C. et al. Hybrid heuristic and mathematical programming in oil pipelines networks. In: IEEE CONGRESS ON EVOLUTIONARY COMPUTATION, 2004. p. 1479-1486. v. 2.

GEIGER, M. J. Randomized variable neighborhood search for multi objective optimization. In: DESIGN AND EVALUATION OF ADVANCED HYBRID META-HEURISTICS -EU/ME, 2004, Nottingham, UK. *Proceedings*... p. 34-42.

GUIMARÃES, I. F.; PANTUZA, G.; SOUZA, M. J. F. Modelo de simulação computacional para validação dos resultados de alocação dinâmica de caminhões com atendimento de metas de qualidade e de produção em minas a céu aberto. In: SIMPÓSIO DE ENGENHARIA

DE PRODUÇÃO - SIMPEP, 14., 2007, Bauru. *Anais*... 11 p. CD-ROM.

HORN, J.; NAFPLIOTIS, N.; GOLDBERG, D. E. A niched pareto genetic algorithm for multiobjective optimization. In: IEEE CONFERENCE ON EVOLUTIONARY COMPUTATION, IEEE WORLD CONGRESS ON COMPUTATIONAL INTELLIGENCE, 1., 1994, Piscataway, USA. *Proceedings*... p. 82-87.

IIMA, H. Preposition of selection in a genetic algorithm for a job shop rescheduling problem. In: INTERNATIONAL CONFERENCE ON EVOLUTIONARY MULTICRITERION OPTIMIZATION, 3., 2005, Guanajuato, Mexico, 2005. *Proceedings*...

IIMA, H.; SANNOMIYA, N. Module type genetic algorithm for modified scheduling problems with worker allocation. In: AMERICAN CONTROL CONFERENCE, 2001, Arlington, VA. *Proceedings*...

ISHIBUSHI, H.; MURATA, T. A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*, v. 28, n. 3, p. 392-403, 1998.

LEI, D.; WU, Z. Crowding-measure-based multiobjective evolutionary algorithm for job shop scheduling. *International Journal of Advanced Manufacturing Tecnology*, v. 30, n. 1-2, p. 112-117, 2005. http://dx.doi.org/10.1007/s00170-005-0029-6

LI, J.; PAN, Q.; LIANG, Y. An effective hybrid tabu search algorithm for multi-objective flexible job-shop scheduling problems. *Computers & Industrial Engineering*, v. 59, p. 647-662, 2010. http://dx.doi.org/10.1016/j.cie.2010.07.014

MELLO, M. H.; FERREIRA, J. C. E. Avaliação de presença de recursos alternativos em plano de processos para melhorar o desempenho de sistemas manufatura. *Produção Online*, v. 14, n. 2, p. 648-678, 2014. http://dx.doi.org/10.14488/1676-1901.v14i2.1467

MLADENOVIC, N.; HANSEN, P. Variable neighborhood search. *Computers and Operations Research*, v. 24, n. 11, p. 1097-1100, 1997. http://dx.doi.org/10.1016/S0305-0548(97)00031-2

OTTONI, R. S.; ARROYO, J. E. C.; SANTOS, A. G. Algoritmo vns multi-objetivo para um problema de programação de tarefas em uma máquina com janelas de entrega. In: SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL - SBPO, XLIII., 2011, Ubatuba, SP. *Anais*...

OSAWA, A.; IDA, K. Scheduling problem with worker allocation using genetic algorithm. In: JAPAN-AUSTRALIA WORKSHOP ON INTELLIGENT AND EVOLUTIONARY SYSTEMS, 2005. p. 1-8.

OSAWA, A.; IDA, K. A solution method of scheduling problem with worker allocation by a genetic algorithm. *IEEJ Transactions on Electronics*, *Information and Systems*, v. 127, n. 5, p. 755-761, 2007. http://dx.doi.org/10.1541/ieejeiss.127.755

QUIAN, B. et al. Scheduling multi-objective job shops using a memetic algorithm based on differential evolution. *International Journal of Advanced Manufacturing Tecnology*, v. 35, n. 9-10, p. 1014-1027, 2006. http://dx.doi.org/10.1007/s00170-006-0787-9

RITZEL, B. J.; EHEART, J. W.; RANJITHAN, S. Using genetic algorithms to solve a multiple objective groundwater pollution containment problem. *Water Resources Research*, v. 30, n. 5, p. 1589-1603, 1994. http://dx.doi.org/10.1029/93WR03511

RUIZ, S.; CASTRILLÓN, O. D.; SARACHE, W. A. Una metodología multiobjetivo para optimizar un ambiente job shop. *Información Tecnológica*, v. 23, n. 1, p. 35-46, 2012. http://dx.doi.org/10.4067/S0718-07642012000100005

SCHAFFER, J. Multiple objective optimization with vector evaluated genetic algorithms. In: INTERNATIONAL CONFERENCE ON GENETIC ALGORITHMS, 1., 1985. *Proceedings*... p. 93-100. v. 1.

SILVA, A. L.; RENTES, A. F. Um modelo de projeto de layout para ambientes job shop com alta variedade de peças baseado nos conceitos da produção enxuta. *Gestão & Produção*, v. 19, n. 3, p. 531-541, 2012. http://dx.doi.org/10.1590/S0104-530X2012000300007

SRIVIVAS, N.; DEB, K. Multiobjective optimization using non dominated sorting in genetic algorithms. *Evolutionary Computation*, v. 2, n. 3, p. 221-248, 1995.

SURESH, R. K.; MOHANASUNDARAM, K. M. Pareto archived simulated annealing for job shop scheduling with multiple objectives. *International Journal of Advanced Manufacturing Technology*, v. 29, n. 1-2, p. 184-196, 2004. http://dx.doi.org/10.1007/s00170-004-2492-x

TAN, S.; WENG, W.; FUJIMURA, S. Scheduling of worker allocation in the manual labor environment with genetic algorithm. In: INTERNATIONAL MULTICONFERENCE OF ENGINEERS AND COMPUTER SCIENTISTS – IMECS, 2009, Hong Kong. *Proceedings*... v. 1.

TAVARES NETO, R. F.; GODINHO FILHO, M. Otimização por colônia de formigas para o problema de sequenciamento de tarefas em uma única máquina com terceirização permitida. *Gestão & Produção*, v. 20, n. 1, p. 76-86, 2013. http://dx.doi.org/10.1590/S0104-530X2013000100006

XIA, W.; WU, Z. An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. *Computers & Industrial Engineering*, v. 48, n. 2, p. 409-425, 2005. http://dx.doi.org/10.1016/j.cie.2005.01.018

ZITZLER, E. *Evolutionary algorithms for multiobjective optimization: methods and applications*. 1999. 122 f. Tese (Doutorado)-Federal Institute of Technology Zurich, Zurich, Swiss, 1999.