# A Variable Neighborhood Search Algorithm for the Long-term Preventive Maintenance Scheduling Problem

Roberto D. Aquino, Jonatas B. C. Chagas and Marcone J. F. Souza
*Instituto de Ciências Exatas e Biológicas, Departamento de Computação, Universidade Federal de Ouro Preto, Ouro Preto, Brazil*

Keywords: Long-term Maintenance Programming, Scheduling, Variable Neighborhood Search, Combinatorial Optimization, Meta-heuristic.

Abstract: In this work we propose a Variable Neighborhood Search (VNS) approach for the long-term maintenance programming of an iron ore processing plant of a company in Brazil. The problem is a complex maintenance programming where we have to assign the machine preventive programming orders to the available work teams over a 52-week planning. In order to evaluate our solution we developed a general mixed integer programming model and used the numerical results as the benchmark. The proposed VNS approach improved most of the instances leading to new benchmarks.

## 1 INTRODUCTION

Preventive maintenance of industrial machines and equipment is an effective way to keep the production system in good condition. In fact, this type of maintenance seeks to correct failures even before they occur, thus avoiding losses and risks of interruption of production. This practice consists in periodically performed maintenance on the basis of technician's experience or according to specifications in the manufacturer's manual.

Problems related to maintenance optimization have become very attractive, arousing the interest of several researchers. (Simões et al., 2011) made an review on maintenance performance measurement published in 67 journals between 1969 and 2009. (Sharma et al., 2011) also made an extensive review on this area, which were reviewed 104 articles starting from early 1960s. According to these last authors, maintenance optimization could have several optimization criteria such as maintenance cost rate, profitability, plant utilization, performance efficiency and work safety. Although there are some articles on optimization of maintenance cost, they are all focused on the maintenance strategy such as Reliability Centered Maintenance (RCM), Total Productive Maintenance (TPM) and Plant Asset Management (PAM). All these techniques have several trade-offs that have to be balanced to give an optimal solution.

(Yamayee et al., 1983) proposed a mathematical formulation for optimal preventive maintenance scheduling problem and used dynamic programming as a framework to solve it. The problem was to schedule 21 maintenance orders with different capacities and different costs.

(Yao et al., 2004) proposed a mixed integer programming model for the short-term preventive maintenance scheduling for 29 maintenance orders distributed on 11 different tools associated with a time window. The objective is to maximize the overall tool availability and minimize unavailability during the periods when a significant amount of work is expected.

(Adhikary et al., 2016) present a multi-objective genetic algorithm for a preventive maintenance scheduling problem that involves the maximization of availability and minimization of maintenance cost for a continuous operating series system. The results reached by the authors shows that the genetic algorithm can improve the availability along with profound reduction of the maintenance cost.

This work has its focus on the problem of assigning the machine preventive maintenance orders to the work teams over a specific time period. The goal is to maximize the number of orders performed and minimize the number of work teams necessary to execute them.

In real scenarios we have to take into account specific constraints related to the maintenance order programming. Most of these constraints are directly re-

Aquino, R., Chagas, J. and Souza, M.
A Variable Neighborhood Search Algorithm for the Long-term Preventive Maintenance Scheduling Problem.
DOI: 10.5220/0006689703030310
In *Proceedings of the 20th International Conference on Enterprise Information Systems (ICEIS 2018)*, pages 303-310
ISBN: 978-989-758-298-1

lated to the scheduling problems, widely studied in the literature (see (Brucker, 2007)). The work team (group of workers that have to work together) cannot perform more than one maintenance order at the same time. Moreover, each machine cannot have more than one preventive maintenance being executed at the same time. Each preventive maintenance also have a time window limit that must be obeyed. Besides above mentioned constraints, each preventive maintenance has to be performed by a team that has the ability to perform it. Furthermore, there exists usually more than one team that is able to execute the same preventive maintenance.

At the studied company, the long-term maintenance programming is made by the engineering team using a specific maintenance software. It is a general maintenance software that manages all preventive and corrective maintenance activities such as work order, inventory, and material management. It also has the data of all maintenance procedures performed and all the maintenance plans. The maintenance plan for each machine includes all the preventive maintenance that have to be performed, the frequency that each preventive maintenance has to be performed, the expected time to complete the preventive maintenance, the team which is able to perform the preventive maintenance and the materials and tools that are necessary. From this database it is built a 52-weeks preventive maintenance plan, which consists of assigning the preventive maintenance to the teams. The problem is that the maintenance plan is unfeasible and it has to be rescheduled. To do it, the maintenance team works in extra hours, but even so, some preventive maintenance are not allocated.

For solving this problem, we developed a general mixed integer programming model and also a heuristic approach, based on the Variable Neighborhood Search meta-heuristic (Mladenović and Hansen, 1997).

The main difference of our problem from the ones that we found on the literature is the objective function. Here we want to maximize the number of orders executed and try to minimize the number of teams necessary to execute them. The greatest challenge is the huge size of it. In our problem we have more than 30,000 maintenance orders to be executed.

The remaining of this paper is organized as follows. Section 2 shows the mathematical formulation, which formally describes the problem. Section 3 presents a heuristic approach, based on VNS, for solving the problem. The results are discussed in Section 4 and the conclusions are presented in Section 5.

## 2 PROBLEM DEFINITION AND MATHEMATICAL MODEL

The Long-term Preventive Maintenance Scheduling Problem (LTPMSP) can be formally described as follows. Let $E = \{1, 2, \cdots, Q\}$ be the set of $Q$ industrial machines which must be submitted to preventive maintenance. Let also $\mathscr{T} = \{1, 2, \cdots, N\}$ be the set of $N$ preventive maintenance, and let $\mathscr{W} = \{1, 2, \cdots, M\}$ be the set of $M$ available work teams and responsible to realize them. The processing time of the preventive maintenance $i$ is denoted by $P_i$. Each preventive maintenance $i \in \mathscr{T}$ must be realized in a single machine $E_i$ and just one work team is sufficiently capable to perform it, but two or more different maintenance may be associated with the same machine. However, only one preventive maintenance can be performed at a time on a specific machine.

In addition, each preventive maintenance has a time window, i.e, an interval $[e_i, l_i]$, where $e_i$ and $l_i$ correspond, respectively, to the earliest and the latest time available to perform the preventive maintenance $i$. Each preventive maintenance $i \in \mathscr{T}$ requires a work team with a specific ability, being that $\mathcal{W}_i \subseteq \mathscr{W}$ indicates the set of work teams with ability to perform it. If the preventive maintenance $i$ is not performed, there is a penalty cost $C_i$ that must be paid. Each work team $k \in \mathscr{W}$ can process at most one preventive maintenance at a time, and has availability to work by $h_k$ continuous hours. The set $\mathcal{T}_k \subseteq \mathscr{T}$ indicates the preventive maintenance that can be executed by the work team $k$.

The aim of the LTPMSP is to determine a scheduling plan to perform as many preventive maintenance as possible in order to minimize the number of work teams. In order to formulate the problem, we define five sets of decision variables, which are described as follows:

- $x_{ij}^k$: binary variable that gets 1 if maintenance $i$ is performed immediately before maintenance $j$ by the work team $k$; 0, otherwise;
- $y_{ik}$: binary variable that gets 1 if maintenance $i$ is performed by the work team $k$; 0, otherwise;
- $z_k$: binary variable that gets 1 if work team $k$ is used; 0, otherwise;
- $c_{ik}$: completion time of the maintenance $i$ when it is performed by the work team $k$;
- $r_{ij}$: binary variable that gets 1 if maintenance $i$ is performed before maintenance $j$ and 0, otherwise.

With these variables we can describe the LTPMSP by the Mixed-Integer Linear Programming (MILP) formulation expressed by Equations (1)-(17):

$$\min \sum_{k\in\mathscr{W}} z_k + \sum_{i\in\mathscr{T}} C_i \left(1 - \sum_{k\in\mathcal{W}_i} y_{ik}\right) \tag{1}$$

$$\sum_{k\in\mathcal{W}_i} y_{ik} \le 1 \qquad i \in \mathscr{T} \tag{2}$$

$$\sum_{i\in\mathfrak{T}_k\cup\{0\}\setminus\{j\}} x^k_{ij} = y_{jk} \qquad j \in \mathscr{T}, k \in \mathcal{W}_j \tag{3}$$

$$\sum_{j\in\mathfrak{T}_k} x^k_{0j} = z_k \qquad k \in \mathscr{W} \tag{4}$$

$$\sum_{i\in\mathfrak{T}_k\cup\{0\}\setminus\{l\}} x^k_{il} = \sum_{j\in\mathfrak{T}_k\cup\{0\}\setminus\{l\}} x^k_{lj} \qquad k \in \mathscr{W}, l \in \mathfrak{T}_k \tag{5}$$

$$c_{0k} = 0 \qquad k \in \mathscr{W} \tag{6}$$

$$c_{jk} \ge c_{ik} + P_j - M'_{ij}(1 - x^k_{ij}) \qquad k \in \mathscr{W}, i \in \mathfrak{T}_k \cup \{0\}, j \in \mathfrak{T}_k \tag{7}$$

$$c_{ik} \ge (e_i + P_i)\, y_{ik} \qquad k \in \mathscr{W}, i \in \mathfrak{T}_k \tag{8}$$

$$c_{ik} \le l_i \qquad k \in \mathscr{W}, i \in \mathfrak{T}_k \tag{9}$$

$$c_{jk'} \ge c_{ik} + P_j - M'_{ij}(1 - r_{ij}) \qquad k \in \mathscr{W}, k' \in \mathscr{W}, i \in \mathfrak{T}_k, j \in \mathfrak{T}_{k'}, \; | k \ne k', i < j, E_i = E_j \tag{10}$$

$$c_{jk'} \le c_{ik} - P_i + M''_{ij} r_{ij} \qquad k \in \mathscr{W}, k' \in \mathscr{W}, i \in \mathfrak{T}_k, j \in \mathfrak{T}_{k'}, \; | k \ne k', i < j, E_i = E_j \tag{11}$$

$$c_{ik} \le h_k \qquad k \in \mathscr{W}, i \in \mathfrak{T}_k \tag{12}$$

$$x^k_{ij} \in \{0,1\} \qquad k \in \mathscr{W}, i \in \mathfrak{T}_k \cup \{0\}, j \in \mathfrak{T}_k \cup \{0\} \tag{13}$$

$$y_{ik} \in \{0,1\} \qquad k \in \mathscr{W}, i \in \mathfrak{T}_k \cup \{0\} \tag{14}$$

$$z_k \in \{0,1\} \qquad k \in \mathscr{W} \tag{15}$$

$$c_{ik} \ge 0 \qquad k \in \mathscr{W}, i \in \mathfrak{T}_k \tag{16}$$

$$r_{ij} \in \{0,1\} \qquad i \in \mathscr{T}, j \in \mathscr{T} \mid i < j, E_i = E_j \tag{17}$$

aa We define a fictitious maintenance 0 that precedes immediately the first maintenance and follows immediately the last maintenance performed by each work team. The completion time of this maintenance is 0 for all work teams, as imposed by the constraint (6).

The objective function (1) minimizes the total number of work teams, while maximizes the number of maintenance performed through the minimization of the penalties. Constraint (2) ensures that each maintenance is performed by at most one work team. Constraint (3) guarantees if a work team $k$ performs a maintenance $j$, that maintenance must be contained in the maintenance schedule of work team $k$. By constraint (4), if at least one maintenance is assigned to the work team $k$, this work team is used. Constraint (5) ensures the continuity of the maintenance schedule of each work team. Constraint (7) calculates the completion time of all preventive maintenance. The constraints (8) and (9) force that all maintenance is performed in their respective time windows. The constraints (10) and (11) ensure that two or more maintenance are not performed at the same time on the same machine. Note that these last two constraints can be applied only between different work crews because there will be no overlap of execution of maintenance by the same work team, it being ensured by the predecessor constraints. The constraint (12) ensures that the number of working hours of work teams are not exceeded. And, finally, the constraints (13) to (17) define the scope and domain of the decision variables.

The constants $M'_{ij}$ and $M''_{ij}$ in Equations (7), (10) and (11) can be any sufficiently large number that is greater or equal to $l_i + P_j$ and $l_j + P_i$, respectively.

## 3 HEURISTIC APPROACH

As showed in results section (Section 4), the MILP formulation for the LTPMSP presents a high complexity and solving it by this method requires excessive computational time. Therefore, in this section we describe a heuristic approach based on the meta-heuristic Variable Neighborhood Search (VNS) in order to obtain high quality solutions with short computational time. The proposed heuristic and its components are detailed below.

### 3.1 Solution Representation and Evaluation Function

A solution to the problem is represented by a permutation $\pi = \langle \pi_1, \pi_2, \ldots, \pi_N \rangle$ of the $N$ maintenance orders. The evaluation of the quality of a solution $\pi$ is done as follows. Consider a procedure where the preventive maintenance orders are sequentially allocated to the first available time slot that have a duration greater or equal of the preventive maintenance duration. For each preventive maintenance, the available time slot is built checking its time window and the non-allocated time of the teams. When all the allocation conditions are met, the preventive maintenance is allocated at the beginning of the available time slot; otherwise, a penalty is incurred.

In order to clarify the characteristics of this allocate procedure, consider an instance with 6 maintenance orders involving 3 machines, which should be executed by 3 teams. The maintenance orders 1, 2, 3 and 6 can be executed by the work teams 1 and 2, and the maintenance orders 4 and 5 by work team 2. The maintenance orders 1, 3 and 5 must be executed at machine 1, while the maintenance orders 2 and 6 at machine 2 and maintenance order 3 at machine 3. The time windows of the maintenance orders start in the following time: 0, 2, 3, 2, 3, 4; and end in the following time: 4, 7, 9, 6, 8, 7. The processing times of the preventive maintenance are the following: 1, 2, 3, 1, 2, 1. The penalty for not performing a maintenance order is the following: 20, 30, 40, 20, 30, 40. Figure 1 shows an allocation to the permutation $\pi = \langle 1,2,3,4,5,6 \rangle$. The time window of each maintenance order is represented by a horizontal bar, and the time and duration are represented by the filled part. This allocation has a cost of 3.

### 3.2 Initial Solution and Neighborhood Structure

The initial solution is obtained choosing any random permutation $\pi$ of $N$ maintenance orders.

In order to explore the solution space, we define a simple neighborhood structure that consists of exchanging two positions of the permutation $\pi$. Figure 2 shows a solution $s$ and one of its neighbors $s'$ for instance with 6 maintenance orders, while Figure 3

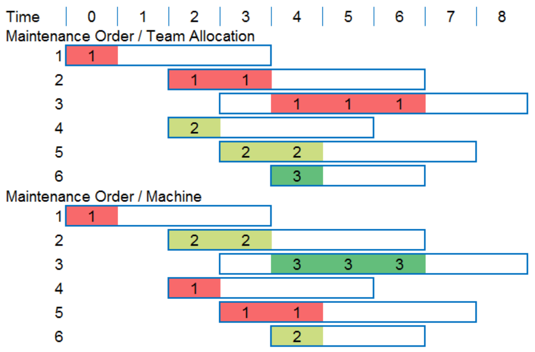

Figure 1: Initial allocation example.

$s = \langle 1, 2, \mathbf{3}, 4, 5, \mathbf{6} \rangle \qquad s' = \langle 1, 2, \mathbf{6}, 4, 5, \mathbf{3} \rangle$

Figure 2: A solution $s$ and a neighbor $s'$.

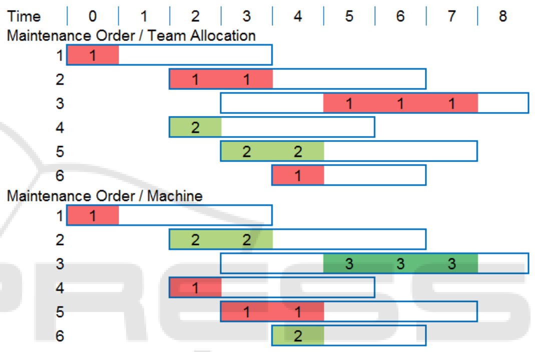

Figure 3: Allocation after swap of preventive maintenance orders 3 and 6.

shows the allocation from $s'$. This allocation is the optimal solution with cost 2.

We represent all neighbors of a solution $s$ are represented by $\mathcal{N}(s)$.

### 3.3 Variable Neighborhood Search

Variable Neighborhood Search (VNS), proposed by Mladenović & Hansen (Mladenović and Hansen, 1997) (see e.g. (Hansen and Mladenović, 2014) for a recent description), is a method that explores the solutions through systematic changes on the neighborhood structures. For each neighborhood, it searches for a local optimum.

Algorithm 1 describes the pseudo-code of VNS for solving the LTMPSP. The algorithm's initial solution (line 1 of Algorithm 1) is generated randomly according to Section 3.2 and it is considered as the best solution so far. While maximum neighborhood structure are not achieved (line 12), the algorithm progressively shakes (see SHAKE procedure, shown in Figure 4) the current best solution $s$ (line 4) and applies a local search (line 5). This local search (see

LOCAL-SEARCH procedure, shown in Figure 5) consists of applying a descent method using a best improvement strategy. If a better solution is achieved (line 6), it is accepted and the neighborhood structure is returned to 1 (lines 7 and 8, respectively). Otherwise, the neighborhood structure is incremented by 1 in line 10. When the loop is interrupted either by time limit or the maximum number of neighborhood structure, the best solution is returned (line 13).

**Algorithm 1:** VNS.

```
1  $s \leftarrow$ initial solution
2  $k \leftarrow 1$
3  repeat
4      $s' \leftarrow$ SHAKE$(s,k)$
5      $s'' \leftarrow$ LOCAL-SEARCH$(s')$
6      if $f(s'') - f(s) < 0$ then
7          $s \leftarrow s'$
8          $k \leftarrow 1$
9      else
10         $k \leftarrow k+1$
11     end
12 until $k > k_{max}$;
13 return $s$
```

```
procedure SHAKE ($s$, $k$)
    $s' \leftarrow s$
    $i \leftarrow 1$
    while $i \leq k$ do
        select a random neighbor solution $s'' \in \mathcal{N}(s')$
        $s' \leftarrow s''$
        $i \leftarrow i+1$
    end
    return $s'$
end procedure
```

Figure 4: SHAKE procedure.

```
procedure LOCAL-SEARCH ($s$)
    $s' \leftarrow s$
    repeat
        get the best neighbor solution $s'' \in \mathcal{N}(s')$
        if $f(s'') < f(s')$ then
            $s' \leftarrow s''$
        end
    until $f(s'') \geq f(s')$;
    return $s'$
end procedure
```

Figure 5: LOCAL-SEARCH procedure.

## 4 COMPUTATIONAL EXPERIMENTS

Our proposed MILP formulation was coded in C++ language using the Concert Technology Library of CPLEX 12.5 Academic Version, with default settings, except for the runtime that was limited to 1 hour. The VNS algorithm was coded in C++ language and executed sequentially. All the experiments were performed on an Intel Core i5-4440 CPU @ 3.10GHz x 4 computer, 8GB RAM, Ubuntu 14.04 LTS 64 bits.

Our VNS algorithm (Algorithm 1) has only one parameter ($k_{max}$) that was empirically determined and set to $N$, i.e, the number of preventive maintenance orders.

The real instance of the studied iron ore processing plant consists of 33,484 preventive maintenance orders involving 1,032 industrial machines to be allocated to 145 work teams. As the MILP formulation is not able to solve real-size instances, we have created 100 different instances in order to compare the VNS algorithm to the MILP. As reported in Table 1, these instances, which are sub-instances of the real instance, differ from each other by the number of preventive maintenance orders, number of industrial machines and number of work teams.

We now present the results obtained by our approaches for the LTPMSP. Notice that since there is no literature regarding this problem, we are only going to compare the results of our approaches.

In Table 1, the four first columns describe the instances, where the columns *ID*, *Q*, *N* and *M* inform, respectively, the identifier number of each instance, the number of industrial machines, the number of preventive maintenance orders and the number of available work teams. The results obtained by the MILP formulation are described in the columns *Obj*, *#T*, *#P*, *Opt*, *Gap* and *Time(s)*. The column *Obj* shows the solution value obtained at the end of the computation, *#T* informs the number of work teams used to perform the *#P* preventive maintenance orders, the column *Opt* shows if each instance was solved to optimality (indicated by an asterisk), the column *Gap* shows the relative gap between upper bound ($Obj$) and *LB* computed as ($Obj$ - $LB$)/$Obj$, where *LB* is the lower bound value at the end of the computation and the column *Time(s)* informs the processing time in seconds spent by the MILP formulation. In the last five columns, we present the VNS results. The column *Obj* shows the best solution value found after 10 executions of the VNS algorithm, while the column $\sigma$ informs the standard deviation of the all 10 solution's values. The columns *#T*, *#P* have the same meaning of the columns for the MILP formulation, which refer to the best solution found by the VNS algorithm. Finally, the column *Time(s)* shows the processing time in seconds required by the VNS.

Notice that in order to highlight the solution quality of each proposed method, best results reached in each instances are boldfaced.

The results show that, within a 1 hour time limit,

Table 1: Comparative analysis of the MILP formulation and the VNS algorithm.

| Instances | | | | MILP | | | | | | VNS | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | Q | N | M | Obj | #T | #P | Opt | Gap | Time(s) | Obj | #T | #P | σ | Time(s) |
| 001 | 2 | 20 | 2 | **434** | 2 | 18 | * | 0.00 | 0.3 | **434** | 2 | 18 | 0.0 | 0.2 |
| 002 | | | 3 | **219** | 3 | 19 | * | 0.00 | 7.2 | **219** | 3 | 19 | 0.0 | 0.2 |
| 003 | | | 4 | **219** | 3 | 19 | * | 0.00 | 82.0 | **219** | 3 | 19 | 0.0 | 0.2 |
| 004 | | | 5 | **219** | 3 | 19 | * | 0.00 | 85.5 | **219** | 3 | 19 | 0.0 | 0.2 |
| 005 | | | 10 | **219** | 3 | 19 | * | 0.00 | 28.8 | **219** | 3 | 19 | 0.0 | 0.3 |
| 006 | | 30 | 2 | **434** | 2 | 28 | * | 0.00 | 2.0 | **434** | 2 | 28 | 0.0 | 1.4 |
| 007 | | | 3 | **219** | 3 | 29 | * | 0.00 | 10.7 | **219** | 3 | 29 | 0.0 | 1.2 |
| 008 | | | 4 | **219** | 3 | 29 | * | 0.00 | 333.4 | **219** | 3 | 29 | 0.0 | 1.2 |
| 009 | | | 5 | 220 | 4 | 29 | | 0.01 | 1h | **219** | 3 | 29 | 0.0 | 1.5 |
| 010 | | | 10 | **219** | 3 | 29 | | 0.01 | 1h | **219** | 3 | 29 | 0.0 | 1.3 |
| 011 | | 40 | 2 | **434** | 2 | 38 | * | 0.00 | 57.2 | 650 | 2 | 37 | 68.3 | 4.8 |
| 012 | | | 3 | **219** | 3 | 39 | * | 0.00 | 186.3 | **219** | 3 | 39 | 0.0 | 4.6 |
| 013 | | | 4 | 220 | 4 | 39 | | 0.01 | 1h | **219** | 3 | 39 | 0.0 | 4.7 |
| 014 | | | 5 | 220 | 4 | 39 | | 0.99 | 1h | **219** | 3 | 39 | 0.0 | 4.8 |
| 015 | | | 10 | 221 | 5 | 39 | | 0.02 | 1h | **219** | 3 | 39 | 0.0 | 5.2 |
| 016 | | 60 | 2 | **434** | 2 | 58 | * | 0.00 | 67.7 | 650 | 2 | 57 | 0.0 | 26.7 |
| 017 | | | 3 | **219** | 3 | 59 | * | 0.00 | 1358.2 | **219** | 3 | 59 | 0.0 | 28.9 |
| 018 | | | 4 | 220 | 4 | 59 | | 1.00 | 1h | **219** | 3 | 59 | 0.0 | 29.9 |
| 019 | | | 5 | 221 | 5 | 59 | | 0.99 | 1h | **219** | 3 | 59 | 0.0 | 29.6 |
| 020 | | | 10 | 652 | 4 | 57 | | 1.00 | 1h | **219** | 3 | 59 | 0.0 | 30.8 |
| 021 | | 80 | 2 | **866** | 2 | 76 | | 0.50 | 1h | 1082 | 2 | 75 | 113.8 | 102.7 |
| 022 | | | 3 | **219** | 3 | 79 | | 0.99 | 1h | **219** | 3 | 79 | 111.5 | 109.5 |
| 023 | | | 4 | 436 | 4 | 78 | | 1.00 | 1h | **219** | 3 | 79 | 113.8 | 112.7 |
| 024 | | | 5 | 653 | 5 | 77 | | 1.00 | 1h | **219** | 3 | 79 | 111.5 | 119.9 |
| 025 | | | 10 | 653 | 5 | 77 | | 1.00 | 1h | **219** | 3 | 79 | 113.8 | 120.7 |
| 026 | 3 | 20 | 3 | **579** | 3 | 17 | * | 0.00 | 2.1 | **579** | 3 | 17 | 0.0 | 0.2 |
| 027 | | | 4 | **220** | 4 | 19 | * | 0.00 | 14.7 | **220** | 4 | 19 | 0.0 | 0.2 |
| 028 | | | 5 | **220** | 4 | 19 | * | 0.00 | 129.9 | **220** | 4 | 19 | 0.0 | 0.2 |
| 029 | | | 6 | **220** | 4 | 19 | * | 0.00 | 3076.4 | **220** | 4 | 19 | 0.0 | 0.2 |
| 030 | | | 12 | **220** | 4 | 19 | | 0.01 | 1h | **220** | 4 | 19 | 0.0 | 0.2 |
| 031 | | 30 | 3 | **579** | 3 | 27 | * | 0.00 | 6.3 | **579** | 3 | 27 | 0.0 | 1.2 |
| 032 | | | 4 | **220** | 4 | 29 | * | 0.00 | 38.3 | **220** | 4 | 29 | 0.0 | 1.0 |
| 033 | | | 5 | **220** | 4 | 29 | * | 0.00 | 316.8 | **220** | 4 | 29 | 0.0 | 1.0 |
| 034 | | | 6 | **220** | 4 | 29 | | 0.01 | 1h | **220** | 4 | 29 | 0.0 | 1.1 |
| 035 | | | 12 | 223 | 7 | 29 | | 0.99 | 1h | **220** | 4 | 29 | 0.0 | 1.2 |
| 036 | | 40 | 3 | **579** | 3 | 37 | * | 0.00 | 807.3 | 723 | 3 | 36 | 74.4 | 4.5 |
| 037 | | | 4 | **220** | 4 | 39 | * | 0.00 | 238.5 | **220** | 4 | 39 | 0.0 | 4.3 |
| 038 | | | 5 | 221 | 5 | 39 | | 0.01 | 1h | **220** | 4 | 39 | 0.0 | 4.3 |
| 039 | | | 6 | 221 | 5 | 39 | | 0.01 | 1h | **220** | 4 | 39 | 0.0 | 4.3 |
| 040 | | | 12 | 225 | 9 | 39 | | 0.99 | 1h | **220** | 4 | 39 | 0.0 | 4.9 |
| 041 | | 60 | 3 | **1515** | 3 | 51 | | 0.76 | 1h | 1659 | 3 | 50 | 63.0 | 24.6 |
| 042 | | | 4 | **220** | 4 | 59 | * | 0.00 | 1619.3 | **220** | 4 | 59 | 0.0 | 29.8 |
| 043 | | | 5 | 221 | 5 | 59 | | 0.01 | 1h | **220** | 4 | 59 | 0.0 | 29.2 |
| 044 | | | 6 | 222 | 6 | 59 | | 0.99 | 1h | **220** | 4 | 59 | 0.0 | 30.3 |
| 045 | | | 12 | 225 | 9 | 59 | | 1.00 | 1h | **220** | 4 | 59 | 0.0 | 31.3 |
| 046 | | 80 | 3 | 3891 | 3 | 59 | | 0.32 | 1h | **1947** | 3 | 69 | 84.9 | 92.8 |
| 047 | | | 4 | 2524 | 4 | 68 | | 0.13 | 1h | **220** | 4 | 79 | 0.0 | 104.3 |
| 048 | | | 5 | 2525 | 5 | 68 | | 0.26 | 1h | **220** | 4 | 79 | 0.0 | 105.8 |
| 049 | | | 6 | 2886 | 6 | 66 | | 0.42 | 1h | **220** | 4 | 79 | 0.0 | 116.5 |
| 050 | | | 12 | 12895 | 7 | 16 | | 0.90 | 1h | **220** | 4 | 79 | 0.0 | 121.1 |

our MILP formulation found the optimal solution (column Opt with *) for 26 instances (26% of the total). It is also observed that for 92 instances (92% of the total), the VNS algorithm found solutions whose values are equal or better to those obtained by the MILP formulation.

Notice that, for a given $Q$ and $N$, the MILP formulation behaves better for the smaller values of $M$, since the smaller this value the lower the combinatorial complexity of the problem. As seen from Table 1, the MILP formulation obtained better solutions than the VNS algorithm only for the instances with

Table 1 (continuation): Comparative analysis of the MILP formulation and the VNS algorithm.

| Instances | | | | MILP | | | | | | VNS | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ID | Q | N | M | Obj | #T | #P | Opt | Gap | Time(s) | Obj | #T | #P | σ | Time(s) |
| 051 | 4 | 20 | 3 | **723** | 3 | 16 | * | 0.00 | 530.6 | **723** | 3 | 16 | 90.1 | 0.3 |
| 052 | | | 4 | **220** | 4 | 19 | * | 0.00 | 1862.6 | **220** | 4 | 19 | 0.0 | 0.2 |
| 053 | | | 5 | **220** | 4 | 19 | | 0.01 | 1h | **220** | 4 | 19 | 0.0 | 0.2 |
| 054 | | | 6 | **220** | 4 | 19 | | 0.01 | 1h | **220** | 4 | 19 | 0.0 | 0.2 |
| 055 | | | 12 | **220** | 4 | 19 | | 0.01 | 1h | **220** | 4 | 19 | 0.0 | 0.2 |
| 056 | | 30 | 3 | **723** | 3 | 26 | | 0.40 | 1h | **723** | 3 | 26 | 97.2 | 1.2 |
| 057 | | | 4 | **220** | 4 | 29 | | 0.01 | 1h | **220** | 4 | 29 | 0.0 | 1.1 |
| 058 | | | 5 | 221 | 5 | 29 | | 0.02 | 1h | **220** | 4 | 29 | 0.0 | 1.1 |
| 059 | | | 6 | **220** | 4 | 29 | | 0.01 | 1h | **220** | 4 | 29 | 0.0 | 1.2 |
| 060 | | | 12 | 222 | 6 | 29 | | 0.99 | 1h | **220** | 4 | 29 | 0.0 | 1.4 |
| 061 | | 40 | 3 | **867** | 3 | 35 | | 1.00 | 1h | 1011 | 3 | 34 | 37.2 | 4.2 |
| 062 | | | 4 | **220** | 4 | 39 | | 0.01 | 1h | **220** | 4 | 39 | 0.0 | 4.0 |
| 063 | | | 5 | 221 | 5 | 39 | | 0.02 | 1h | **220** | 4 | 39 | 0.0 | 4.0 |
| 064 | | | 6 | 222 | 6 | 39 | | 0.21 | 1h | **220** | 4 | 39 | 0.0 | 3.7 |
| 065 | | | 12 | 223 | 7 | 39 | | 1.00 | 1h | **220** | 4 | 39 | 0.0 | 4.5 |
| 066 | | 60 | 3 | 2163 | 3 | 47 | | 1.00 | 1h | **2091** | 3 | 47 | 88.5 | 23.1 |
| 067 | | | 4 | 652 | 4 | 57 | | 0.73 | 1h | **220** | 4 | 59 | 0.0 | 28.5 |
| 068 | | | 5 | 221 | 5 | 59 | | 0.61 | 1h | **220** | 4 | 59 | 0.0 | 30.0 |
| 069 | | | 6 | 222 | 6 | 59 | | 0.99 | 1h | **220** | 4 | 59 | 0.0 | 29.2 |
| 070 | | | 12 | 1086 | 6 | 55 | | 1.00 | 1h | **220** | 4 | 59 | 0.0 | 35.3 |
| 071 | | 80 | 3 | **3171** | 3 | 62 | | 1.00 | 1h | 3459 | 3 | 60 | 104.3 | 72.6 |
| 072 | | | 4 | **220** | 4 | 79 | | 0.01 | 1h | **220** | 4 | 79 | 0.0 | 95.4 |
| 073 | | | 5 | 221 | 5 | 79 | | 0.99 | 1h | **220** | 4 | 79 | 0.0 | 103.7 |
| 074 | | | 6 | 222 | 6 | 79 | | 0.99 | 1h | **220** | 4 | 79 | 0.0 | 117.9 |
| 075 | | | 12 | 8215 | 7 | 42 | | 1.00 | 1h | **220** | 4 | 79 | 0.0 | 102.0 |
| 076 | 5 | 20 | 4 | **724** | 4 | 16 | * | 0.00 | 62.9 | **724** | 4 | 16 | 68.3 | 0.3 |
| 077 | | | 5 | **221** | 5 | 19 | * | 0.00 | 277.8 | **221** | 5 | 19 | 0.0 | 0.2 |
| 078 | | | 6 | **221** | 5 | 19 | | 0.01 | 1h | **221** | 5 | 19 | 0.0 | 0.2 |
| 079 | | | 7 | **221** | 5 | 19 | | 0.01 | 1h | **221** | 5 | 19 | 0.0 | 0.2 |
| 080 | | | 14 | **221** | 5 | 19 | | 0.01 | 1h | **221** | 5 | 19 | 0.0 | 0.3 |
| 081 | | 30 | 4 | **724** | 4 | 26 | | 0.20 | 1h | **724** | 4 | 26 | 70.0 | 1.1 |
| 082 | | | 5 | **221** | 5 | 29 | | 0.01 | 1h | **221** | 5 | 29 | 0.0 | 1.1 |
| 083 | | | 6 | **221** | 5 | 29 | | 0.01 | 1h | **221** | 5 | 29 | 0.0 | 1.1 |
| 084 | | | 7 | **221** | 5 | 29 | | 0.01 | 1h | **221** | 5 | 29 | 0.0 | 1.1 |
| 085 | | | 14 | 225 | 9 | 29 | | 0.98 | 1h | **221** | 5 | 29 | 0.0 | 1.2 |
| 086 | | 40 | 4 | **868** | 4 | 35 | | 1.00 | 1h | **868** | 4 | 35 | 107.6 | 3.5 |
| 087 | | | 5 | **221** | 5 | 39 | | 0.01 | 1h | **221** | 5 | 39 | 0.0 | 3.4 |
| 088 | | | 6 | 222 | 6 | 39 | | 0.02 | 1h | **221** | 5 | 39 | 0.0 | 3.7 |
| 089 | | | 7 | 222 | 6 | 39 | | 0.02 | 1h | **221** | 5 | 39 | 0.0 | 3.5 |
| 090 | | | 14 | 223 | 7 | 39 | | 1.00 | 1h | **221** | 5 | 39 | 0.0 | 3.9 |
| 091 | | 60 | 4 | **1588** | 4 | 50 | | 1.00 | 1h | 1732 | 4 | 49 | 86.2 | 21.5 |
| 092 | | | 5 | **221** | 5 | 59 | | 0.02 | 1h | **221** | 5 | 59 | 0.0 | 23.0 |
| 093 | | | 6 | 222 | 6 | 59 | | 0.02 | 1h | **221** | 5 | 59 | 0.0 | 24.9 |
| 094 | | | 7 | 223 | 7 | 59 | | 0.99 | 1h | **221** | 5 | 59 | 0.0 | 27.4 |
| 095 | | | 14 | 225 | 9 | 59 | | 1.00 | 1h | **221** | 5 | 59 | 0.0 | 28.9 |
| 096 | | 80 | 4 | 3388 | 4 | 61 | | 0.92 | 1h | **3316** | 4 | 61 | 108.7 | 67.9 |
| 097 | | | 5 | 509 | 5 | 78 | | 0.43 | 1h | **221** | 5 | 79 | 0.0 | 82.7 |
| 098 | | | 6 | 510 | 6 | 78 | | 0.43 | 1h | **221** | 5 | 79 | 0.0 | 86.2 |
| 099 | | | 7 | 511 | 7 | 78 | | 0.43 | 1h | **221** | 5 | 79 | 0.0 | 98.6 |
| 100 | | | 14 | 3102 | 6 | 67 | | 0.91 | 1h | **221** | 5 | 79 | 0.0 | 101.2 |

the smallest number of workers, considering the instances with the same number preventive maintenance orders and the same number of industrial machines.

According to the column σ, we can note that the VNS algorithm presented good convergence, as the standard deviation of the 10 solutions obtained is equal to zero or close to zero for most of the instances.

Regarding the computational time and the solution's quality, we can see that the VNS algorithm is more efficient while compared to the MILP formulation. For all instances the VNS algorithm was faster than the MILP formulation, and analyzing the 8 instances (instances with ID 011, 016, 021, 036, 041, 061, 071 and 091) in which the VNS algorithm obtained worse solutions than the MILP formulation, we can see that the difference between the solutions is only an unperformed preventive maintenance, except for the instance with ID 071, which has that difference of 2 preventive maintenance orders.

Note that for none of the 100 instances, reported in Table 1, the MILP formulation and the VNS algorithm were able to allocate all preventive maintenance orders for the set of available work teams (see the columns *N* and *#P*). This occurs even when the number of available work teams is greater than the number of used work teams (see the columns *M* and *#T*), and even when the optima solution was found (see e.g. the instances with ID 003, 004, 005, 008, 028, 029). This fact arises because the time windows defined for maintenance orders prevent them from being performed due to the constraint that only one maintenance can be performed at the same time on a specific machine (see constraints (10) and (11) in the MILP formulation described in Section 2).

For the complete instance it was not possible to make a comparison with the MILP formulation because the CPLEX optimizer could not handle the instance. To get a result with the VNS approach within a reasonable time, we limited the application of the algorithm to one hour. Even with this time limit, the VNS could assign 30,975 maintenance orders from a total of 33,484 (92,5% of the orders).

## 5 CONCLUSIONS

In this paper, we examined the optimization approach proposed in the literature and built our own MILP model based on the real objective function and restrictions of a preventive maintenance of an existing iron ore processing plant. This MILP model is very straightforward and helps us understand the problem mathematically. Although its easiness to understand, it can not solve large instances. In order to get a reasonable solution, we proposed a VNS approach that was able to handle the real instance with good results.

Although the proposed algorithm gave good results, it was not able to compare the results of the real instance with a baseline. In the future, we would like to implement more meta-heuristic approaches and evaluate the results using the VNS as the baseline. Other enhancement is a improvement on the local search and shaking algorithm to consider only exploring promising solutions.

## ACKNOWLEDGEMENTS

The authors thank Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG), Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and Universidade Federal de Ouro Preto (UFOP) for supporting this research.

## REFERENCES

Adhikary, D. D., Bose, G. K., Jana, D. K., Bose, D., and Mitra, S. (2016). Availability and cost-centered preventive maintenance scheduling of continuous operating series systems using multi-objective genetic algorithm: A case study. *Quality Engineering*, 28(3):352–357.

Brucker, P. (2007). *Scheduling algorithms*, volume 3. Springer.

Hansen, P. and Mladenović, N. (2014). Variable neighborhood search. In *Search methodologies*, pages 313–337. Springer.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100.

Sharma, A., Yadava, G., and Deshmukh, S. (2011). A literature review and future perspectives on maintenance optimization. *Journal of Quality in Maintenance Engineering*, 17(1):5–25.

Simões, J. M., Gomes, C. F., and Yasin, M. M. (2011). A literature review of maintenance performance measurement: A conceptual framework and directions for future research. *Journal of Quality in Maintenance Engineering*, 17(2):116–137.

Yamayee, Z., Sidenblad, K., and Yoshimura, M. (1983). A computationally efficient optimal maintenance scheduling method. *IEEE Transactions on Power Apparatus and Systems*, 102(2):330–338.

Yao, X., Fernández-Gaucherand, E., Fu, M. C., and Marcus, S. I. (2004). Optimal preventive maintenance scheduling in semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 17(3):345–356.