# A Single Machine Scheduling with Periodic Maintenance

Francisco Ángel-Bello[1] Ada Álvarez[2] Joaquín Pacheco[3] Iris Martínez[1]

[1] Center for Quality and Manufacturing, Tecnológico de Monterrey,
Eugenio Garza Sada 2501, 64849 Monterrey, NL, Mexico

[2] Graduate Program in Systems Engineering, Universidad Autónoma de Nuevo León
Pedro de Alba s/n, San Nicolás de los Garza, N.L., México

[3] Department of Applied Economics, Universidad de Burgos
Plaza Infanta Doña Elena s/n, Burgos, España

## Abstract

In this paper a problem of sequencing tasks in a machine with programmed preventive maintenance and sequence-dependent set-up times is addressed. The problem has similarities with the Vehicle Routing Problem with distance constraint and asymmetric costs and is an NP-hard problem. A solution approach based on metaheuristic procedures has been designed and implemented and the computational experiments show that it finds good solutions in reasonably short times.

Keywords*: Maintenance, Scheduling, Metaheuristics*.

## 1 Introduction

Planning and scheduling problems have been traditionally modeled and solved on directed graphs, starting for example, with the classical Critical Path Method to obtain the schedule with the minimum makespan of *n* jobs with precedence constraints [13].

Specifically, many scheduling problems can be seen as particular routing problems. A reformulation of the Job Shop Problem as a Vehicle Routing Problem (VRP) can be consulted in the work of Beck, Prosser & Selensky [2]. Kouki et al. [8] present different analogies between the VRP and the Flexible Job Shop Scheduling Problem.

Several similarities and analogies have been investigated between the Travel Salesman Problem (TSP) and the Single Machine Scheduling Problem with sequence-dependent set-up times, and common heuristics have been established for both of them.

A comprehensive survey paper on scheduling problems with setup times or costs was conducted by Allahverdi *et al.*[1].

The work of Tan and Narasimhan [14] is among the firsts that use metaheuristic techniques to solve this problem. They developed a simulated annealing-based algorithm that over performed, in terms of the solution quality, the heuristic algorithms published previously.

Genetic algorithms are the most popular algorithms that have been used to solve this kind of problems; mainly due to the simplicity of implementation and also to the fact that some commercial packages bring them incorporated as toolboxes.

Gagne *et al.* [4] proposed an Ant-colony algorithm that performed very well compared to others, in both quality and computational time. Gupta and Smith [6] developed an approach based on GRASP metaheuristic, yielding better results than the ant-colony algorithm, but consuming more computing time.

Taking into account that machines are an essential part in the production process and that sometimes it is needed to stop their work due to failures or to do adjustments, it is very important to design a program of preventive maintenance that allows reducing the rate of stops.

Recently, researchers have become aware of that issue and have been concerned about including maintenance activities in the production planning in conjunction with the jobs to be processed. We can mention for example the works of Liu and Chen [11]; Liao and Chen [10]; Ji, He and Cheng [7], Low *et*

*al.*[12] among others. Nevertheless, these works do not consider setup times that depend on the order the jobs are processed.

To the best of our knowledge, the problem of sequencing tasks in a machine with programmed preventive maintenance and sequence-dependent set-up times has not treated previously. Nevertheless, there are numerous papers which study each problem separately.

In the present paper we include the preventive maintenance in the planning process. As a result, we obtain a model that combines the problem of sequencing tasks with dependent setup costs and the problem with availability restrictions. It is an NP-hard problem in the strong sense that even for relatively small instances it is not possible to solve by exact algorithms. The main goal of this work is to design and implement solution methodologies based on metaheuristic procedures to find good solutions to this problem in reasonably short times.

## 2 Statement of the problem

As it has been mentioned above, we have a group of jobs to be processed in a single machine which has setup times that depend on the order the jobs are processed. After a fixed amount of time, a maintenance activity should be performed in the machine. Considering that, in general, the available time between two consecutive maintenance activities is not enough to process all the jobs, more than one block of jobs should be programmed. The problem consists on determining the assignment of jobs for each block and the order in which they should be processed in order to minimize the total time required to process all the tasks.

For modelling it, we bear in mind its similarities with the Vehicle Routing Problem with distance constraint and asymmetric costs. For this purpose, each block between two consecutive maintenance activities can be seen as a Travelling Salesman and the objective would be to minimize the number of salesman and to ensure that the distance travelled by the one associated to the last block be as short as possible.

The addressed problem has the following characteristics:

- There are $n$ jobs to schedule in a single machine.
- Each job $j$ has associated a processing time $p_j$
- For each job $j$ there is a preparation time $S_{ij}$ that depends on the job $i$ processed just before job $j$.
- After each period $T$ of time a maintenance activity must take place in the machine. This activity will be considered as a job with index 0.
- The jobs can not be interrupted, that is, for scheduling a job it should be enough time (before the following maintenance activity) to prepare the machine and to process the job.
- Each maintenance activity consumes a fixed amount of time $p_0$, and requires a preparation time $S_{i0}$ that depends on the last job $i$ processed.
- Every time a maintenance activity is completed, there is a preparation time $S_{0j}$ for each job $j$.
- All the jobs are available at time zero.
- Objective: To assign jobs to blocks between maintenance activities in such a way that the last job finishes as soon as possible.

To model the problem we have introduced the following notation. Let $G = (V, A)$ an oriented graph, with $V = \{0,1,\cdots,n\}$ being the set of nodes and $A$ the set of arcs. Node 0 is associated to maintenance activities and the subset $\{1,2,\cdots,n\}$ to the jobs. Each arc $(i, j)$ has associated a weight $c_{ij} \geq 0$, defined as the sum of the time required to prepare the machine and to process the job $j$ just after job $i$, that is, $c_{ij} = S_{ij} + p_j$. In general, $c_{ij} \neq c_{ji}$ and do not necessarily satisfy the triangle inequality. Using arc $(i, j)$ in a solution means that job $j$ will be processed right after job $i$.

The decision variables are:

$$x_{ij}^{(1)} = \begin{cases} 1, \text{if arc } (i,j) \text{ is used in the last block} \\ 0, \text{otherwise} \end{cases}$$

$$x_{ij}^{(2)} = \begin{cases} 1, \text{if arc } (i,j) \text{ is used in a block different from the last one} \\ 0, \text{otherwise} \end{cases}$$

$$y_i = \begin{cases} 1, \text{if job } i \text{ is processed in the last block} \\ 0, \text{otherwise} \end{cases}$$

$s$ : Represents the number of blocks different from the last one

The model is:

$$\min z = Ts + \sum_{i=0}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} c_{ij} x_{ij}^{(1)}$$

Subject to:

$$\sum_{j=1}^{n} x_{oj}^{(1)} = 1 \tag{1}$$

$$\sum_{i=1}^{n} x_{i0}^{(1)} = 1 \tag{2}$$

$$\sum_{j=1}^{n} x_{oj}^{(2)} = s \tag{3}$$

$$\sum_{i=1}^{n} x_{i0}^{(2)} = s \tag{4}$$

$$\sum_{\substack{j=0 \\ j \neq i}}^{n} x_{ij}^{(1)} = y_i \ \ (i = 1,2,\cdots n) \tag{5}$$

$$\sum_{\substack{i=0 \\ i \neq j}}^{n} x_{ij}^{(1)} = y_j \ (j = 1,2,\cdots n) \tag{6}$$

$$\sum_{\substack{j=0 \\ j \neq i}}^{n} x_{ij}^{(2)} = 1 - y_i \ \ (i = 1,2,\cdots n) \tag{7}$$

$$\sum_{\substack{i=0 \\ i \neq j}}^{n} x_{ij}^{(2)} = 1 - y_j \ \ (j = 1,2,\cdots n) \tag{8}$$

$$u_i^{(k)} - u_j^{(k)} + (T + c_{ij}) x_{ij}^{(k)} \leq T \tag{9}$$

$$(i = 1,2,\cdots n;\ j = 1,2,\cdots n;\ j \neq i;\ k = 1,2)$$

$$c_{0i} x_{0i}^{(k)} \leq u_i^{(k)} \leq T - c_{i0} x_{i0}^{(k)} \ (i = 1,2,\cdots n; k = 1,2) \tag{10}$$

$$x_{ij}^{(k)} \in \{0,1\} \ (i = 0,1,\cdots n;\ j = 0,1,\cdots n;\ j \neq i; k = 1,2) \tag{11}$$

$$y_i \geq 0, u_i^{(k)} \geq 0 \ \ (i = 1,2,\cdots n; k = 1,2) \tag{12}$$

Note that the first term in the objective function minimizes the number of blocks different from the last one, and the second term guarantees that the last block finishes as soon as possible. Constraints (1)–(8) can be interpreted as the VRP classical in-degree and out-degree constraints for all the nodes, including the central depot. In our context (1) y (3) establish that just one job can be processed right after an maintenance activity , while (2) and (4) establish that just one job can be processed right before. Constraints (5) impose that for each job in the last block there is exactly one arc to other job (including the maintenance activity) in this block, while (6) impose the same for entering arcs. Constraints (7) and (8) are equivalents to (5) and (6) for the nodes belonging to other blocks.

Constraints (9) have the same spirit that those obtained from the classical sub-tour elimination constraints for the TSP [15]. Constraints (10) take into account that the costs may not satisfy the triangle inequality. They guarantee that the time $T$ between two maintenance activities is not exceeded and that each block starts and finishes with a maintenance activity.

## 3 Solution Approach

Experimental analyses were performed to validate the model. Cplex 9.1 was used to solve several test instances and we observed that even for the smallest instances, the solver consumes a lot of time, not finding any feasible solution after 12 hours. Therefore, we decided to design a procedure based on the GRASP ideas [3] to solve the problem.

GRASP is a methodology that constructs an initial solution via an adaptive randomized greedy function. Then, a local search is conducted using the constructed solution as an initial starting point.

In the approach designed in this paper we follow these ideas, applying a constructive procedure just for each block between two maintenance activities.

For the block of jobs that is being constructed, the procedure can be sketched as follows:

- Insertion phase: While there is enough time, jobs are assigned to the block
- Sequencing-improvement phase: Jobs are re-accommodated in order to free some time

These two phases are repeated iteratively until, after a sequencing-improvement phase there is no free time to add any un-assigned job. Then, if there are still un-assigned jobs, a new block has to be constructed; in other case a feasible solution to the problem has been obtained. Incorporating randomness in the insertion phase (as it will be explained later) allows for different solutions to be obtained each time the procedure is run. Therefore, repeated applications of the procedure are executed and the best overall solution is kept as the result. We describe this procedure in depth as follows.

*Insertion phase:*

Let *SP* be the partial sequence in the block under construction and *Sa* the set of un-assigned jobs. To select the first job to insert in the block, we consider the following greedy function that measures the time that is left to the next maintenance activity:

$$r_i = T - c(0,i) - c(i,0), \quad \forall i \in Sa$$

To insert the next jobs we proceed in the following way:

For each job *i* and each possible insertion point *l*, the greedy adaptive function measures the time $r_{il}$ that is left to the next maintenance activity if the job *i* is inserted at position *l* in that partial sequence. That is, for each un-assigned job we calculate:

$$r_{il} = r + c(SP_{l-1}, SP_l) - c(SP_{l-1}, i) - c(i, SP_l), \quad \forall i \in Sa, \quad l = 1,2,\cdots,k+1$$

Where the following notation has been used:

- $r$ : time left to the beginning of the next maintenance activity .
- $SP_l$ : job in position $l$ in sequence *SP*.
- $k$: Number of jobs in *SP* and $SP_{k+1} = 0$ .
- For convenience $c(i,j) = c_{ij}$

To select the first job to insert in the block under construction, the $r_i$ values are sorted in a non-increasing order and a restricted candidate list (RCL) is made up of the first $p$ elements, where $p$ is a parameter (cardinality-based mechanism). From RCL one element is randomly chosen as the first job to insert. In order to select the remaining jobs to insert in the block, we proceed in a similar way, considering the values $r_{il}$ .

Note that once a job has been inserted in the partial sequence *SP*, the set *Sa* and the time *r* should be updated.

When it is not possible to add any new job, the next phase is performed.

*Sequencing-improvement phase:*

To decrement the time consumed in the block, two different procedures were tested: 1) tabu search and 2) solving exactly the travelling salesman problem associated to the block.

For the first procedure a local search based on jobs interchange is implemented, guided by a basic tabu search with short term memory [5]. The added and removed arcs are registered in a tabu list, and are not allowed to be interchanged during a pre-established number of iterations. The best solution found after a fixed number of iterations is saved.

## 4 Computational experiments

As in the related literature we did not find test instances for the specific problem addressed here, so we used instances for the Asymmetric VRP, specifically eight instances created by Toth and Vigo [15]. See (http://www.or.deis.unibo.it/research_pages/ORinstances/VRPLIB/VRPLIB.html).

For determining the distance constraint, the procedure proposed by Li, Simchi-Levy and Desrocher [9] was applied. We generated four possible distance constraints as a function of the most distant point (*dm*), in such a way that the distance constraint λ takes values: 2.25*dm*, 2.5*dm*, 3*dm* ó 4*dm*. In this context, the central depot corresponds to the maintenance activity; each client corresponds to a job to be processed; the distance between clients *i* and *j* is $c_{ij}$ (the total time required to process job *j* right after job *i*); and finally λ is equivalent to *T*, the time between two consecutive maintenance activities.

All experiments were conducted on a Pentium 4 PC with a 3.00 GHz and 1 GB RAM processor, under Windows 2000. The solution procedure was coded in C++ 6.1 and the exact solutions for the TSP in each block were obtained with CPLEX 9.1.

Different sizes for the restricted candidate list were tested. Table 1 shows the best results obtained for each instance with both procedures designed for the Sequencing-improvement phase. The column "Completion Time" shows the time the last job finishes; the column "Blocks" indicates how many blocks were needed and column "Time" shows the computing time used for each procedure.

As it can be seen, there were better results obtained for the completion times with the tabu search in the Sequencing-improvement phase in more than the 80 percent of the test instances used. Nevertheless, it should be remarked that the number of iterations allowed in the procedure with the exact solution was very small. In particular, for instances A065-4f, A071-225f, A071-25f, A071-3f and A071-4f, only five or less iterations were allowed. This is due to the excessive computing time it takes.

Note that if we applied both improvement procedures to a block, the exact TSP must release more time compared to the tabu search. Therefore, when new jobs are inserted, the blocks obtained can contain different jobs in general.

Table 1: Computational results

| Dataset | Improving with Tabu Search | | | Improving with CPLEX | | |
|---|---|---|---|---|---|---|
| | Completion Time | Blocks | Time (sec) | Completion Time | Blocks | Time (sec) |
| *A034-2.25dm* | 1377 | 2 | 0.219 | 1587 | 3 | 12.125 |
| *A034-2.5dm* | 1393 | 2 | 0.203 | 1393 | 2 | 24.390 |
| *A034-3dm* | 1374 | 2 | 0.015 | 1344 | 2 | 48.625 |
| *A034-4dm* | 1453 | 2 | 0.031 | 1259 | 1 | 281.171 |
| *A036-2.25dm* | 1635 | 3 | 0.219 | 1720 | 3 | 5.468 |
| *A036-2.5dm* | 1542 | 2 | 0.500 | 1580 | 2 | 8.172 |
| *A036-3dm* | 1559 | 2 | 0.015 | 1559 | 2 | 13.281 |
| *A036-4dm* | 1493 | 2 | 0.078 | 1493 | 2 | 84.969 |
| *A039-2.25dm* | 1741 | 3 | 0.250 | 1830 | 3 | 16.828 |
| *A039-2.5dm* | 1592 | 2 | 0.062 | 1592 | 2 | 30.516 |
| *A039-3dm* | 1583 | 2 | 0.625 | 1630 | 2 | 41.844 |
| *A039-4dm* | 1596 | 2 | 0.422 | 1612 | 2 | 190.328 |
| *A045-2.25dm* | 1842 | 3 | 0.656 | 1938 | 3 | 19.578 |
| *A045-2.5dm* | 1825 | 3 | 0.375 | 1847 | 3 | 27.797 |
| *A045-3dm* | 1658 | 2 | 0.406 | 1656 | 2 | 58.937 |
| *A045-4dm* | 1676 | 2 | 0.531 | 1676 | 2 | 145.078 |
| *A048-2.25dm* | 2127 | 3 | 0.343 | 2127 | 3 | 35.250 |
| *A048-2.5dm* | 2057 | 3 | 0.765 | 2122 | 3 | 99.937 |
| *A048-3dm* | 1877 | 2 | 0.922 | 1851 | 2 | 238.75 |
| *A048-4dm* | 1968 | 2 | 0.593 | 1955 | 2 | 936.313 |
| *A056-2.25dm* | 1864 | 3 | 1.000 | 1878 | 3 | 78.688 |
| *A056-2.5dm* | 1835 | 3 | 0.563 | 1919 | 3 | 121.141 |
| *A056-3dm* | 1737 | 2 | 0.625 | 1758 | 2 | 344.969 |
| *A056-4dm* | 1749 | 2 | 1.641 | 1677 | 2 | 2457.719 |
| *A065-2.25dm* | 2255 | 3 | 0.625 | 2490 | 4 | 151.078 |
| *A065-2.5dm* | 2250 | 3 | 1.375 | 2299 | 3 | 347.156 |
| *A065-3dm* | 2251 | 3 | 1.640 | 2253 | 3 | 1157.218 |
| *A065-4dm* | 2144 | 2 | 1.109 | 2210 | 2 | 63.343 |
| *A071-2.25dm* | 2474 | 4 | 0.016 | 2474 | 4 | 106.047 |
| *A071-2.5dm* | 2321 | 3 | 1.609 | 2735 | 4 | 276.953 |
| *A071-3dm* | 2341 | 3 | 1.875 | 2524 | 3 | 18.485 |
| *A071-4dm* | 2267 | 2 | 2.641 | 2389 | 2 | 70.860 |

## 5 Concluding remarks

In this paper we present the results obtained when addressing a scheduling problem that combines periodic maintenance and sequence-dependent setup times. The complexity of the problem forces to design heuristic algorithms to solve it in a reasonable time. The proposed algorithm is inspired in the GRASP metaheuristic. Two versions were tested to improve the solutions: one based in tabu search and the other one based on an exact method. The algorithm which included the tabu search in its Sequencing-improvement phase performed very well in acceptable times.


### *Acknowledgment*

This work was partially supported by CONACYT (México) grant 61903 and by Spanish Ministry of Science and Education SEJ2005-08923/ECON, ECO2008-06159/ECON. These supports are gratefully acknowledged.


## References


[1] A. Allahverdi, C. T. Ng, T.C.E. Cheng, M.Y. Kovalyov, *A survey of scheduling problems with setup times or costs,* European J. of Operational Research 187 (2008) pp. 985-1032.

[2] J.C. Beck, P. Prosser and E. Selensky, *Vehicle routing and job shop scheduling: What's the difference?* Proceedings of the 13th International Conference on Artificial Intelligence Planning and Scheduling, 2003.

[3] T.A. Feo, M. Resende, *Greedy Randomized Adaptive Search Procedures*, Journal of Global Optimization 6 (1995) pp. 109-133.

[4] C. Gagne, W.L. Price, M. Gravel, *Comparing an ACO algorithm with other heuristics for the single machine scheduling problem with sequence-dependent setup times*. Journal of the Operational Research Society 53 (2002) pp. 895-906.

[5] F. Glover and M. Laguna, *Tabu Search*, Kluwer Academic Publisher, 1997.

[6] S. Gupta, J. Smith, *Algorithms for single machine total tardiness scheduling with sequence dependent setups*. European J. of Operational Research 175 (2006) pp. 722-739.

[7] M. Ji, Y. He, T.C.E. Cheng, *Single-machine scheduling with periodic maintenance to minimize makespan*, Computers & Operations Research 34 (2007) pp. 1764-1770.

[8] Z. Kouki, B. Fayech Chaar, S. Hammadi, M. Ksouri, *Analogies between Flexible Job Shop Scheduling and Vehicle Routing problems,* IEEE International Conference on Industrial Engineering and Engineering Management (2007) pp. 880-884.

[9] Ch. Li, D. Simchi-Levi, M. Desrochers, *On the Distance Constrained Vehicle Routing Problem*. Operations Research 40 (1992) pp. 790-799.

[10] C. J. Liao, W. J. Chen, *Single-machine scheduling with periodic maintenance and non resumable jobs,* Computers & Operations Research 30 (2003) pp. 1335-1347.

[11] Z. Liu, T.C. Cheng, *Scheduling with job release dates, delivery times and preemption penalties*. Information Processing Letters 82 (2002) pp. 107-111.

[12] Ch. Low, Ch-J. Hsu, Ch-T Su, *Minimizing the makespan with an availability constraint on a single machine under simple linear deterioration*, Computers and Mathematics with Applications 56 (2008) pp. 257-265.

[13] M. Pinedo, Planning *and Scheduling in Manufacturing and Services*, Springer Series in Operations Research*s*, New York, 2005.

[14] K.C.Tan, R. Narasimhan, *Minimizing tardiness on a single processor with sequence-dependent setup times: A simulated annealing approach*, Omega 25 (1997) pp. 619-634.

[15] P. Toth, D. Vigo, *The vehicle routing problem*. Siam Monographs on Discrete Mathematics and Applications, 2002.