MILP Models for Scheduling Dynamic Jobs with Sequence Dependent Setup Times and a Variable Maintenance on a Single Machine

Costa Antonio[1,*], Corsini Roberto[2], Cappadonna F. Antonio[1] and Fichera Sergio[1]

[1]*Department of Industrial Engineering, University of Catania, Via S. Sofia, 64, 95125 Catania CT, Italia*

[2]*Department of Physics and Astronomy, University of Catania, Via S. Sofia, 64, 95125 Catania CT, Italia*

Abstract: The single machine scheduling problem with variable maintenance has been widely investigated by both academics and practitioners. Differently from most papers proposed so far, and conforming to a real-world process in the semiconductor industry, in this paper a single variable maintenance task has to be carried out within a specific time interval. The maintenance duration is an increasing function of its starting time. The objective is to minimize the total tardiness considering release times and sequence dependent setup times of jobs as well. Since an earlier maintenance starting time implies a smaller maintenance duration but a higher completion time of the subsequent jobs, the best schedule including maintenance activity and jobs has to be achieved. In order to optimally solve the scheduling problem at hand, two distinct mixed integers linear programming models (MILPs) are proposed and compared under the computational efficiency viewpoint.

Keywords: Mathematical programming, Sequencing, Global optimization, maintenance, Manufacturing.

INTRODUCTION

Investigating the combination of job scheduling and machine maintenance in manufacturing systems is a challenging task. Such research areas aim to propose effective approaches for scheduling jobs when a production stoppage, due to maintenance, occurs. Maintenance operations may involve different kinds of activities such as cleaning, tool replacement, inspection, recharging, and so on. Two different research streams, both of them connecting job scheduling and machine maintenance, have been investigated by literature so far, namely scheduling with fixed maintenance and scheduling with variable maintenance.

In the former case, execution time of maintenance is precedingly known; in other words, both starting time and duration of the maintenance task, are determined in advance. A huge amount of research is ascribable to the scheduling issue with fixed maintenance on different configurations of shop floors. The following two surveys are worthy to be mentioned: Schmidt (2000) and Ma *et al*. (2010).

In the latter case, the decision maker has to place the maintenance operation along the time horizon and, in addition, the maintenance duration is a nondecreasing function of its starting time. Hence, maintenance is a sort of dummy job to be scheduled so to optimize a certain performance indicator. The seminal work of Kubzin and Strusevich (2006) studies the makespan minimization for a basic two-machine manufacturing system with variable maintenance. Notably, they proposed a polynomial time approximation scheme and a fast 3/2-approximation algorithm for the problem at hand. Mosheiov and Sidney (2010) investigated a basic single machine scheduling problem in which the maintenance duration is a nondecreasing function of its starting time and the processing time of the job, immediately after the maintenance activity, is shorter than the processing time before the maintenance starts. They demonstrated a minimization of flowtime and showed that maximum delays, total earliness, tardiness and tardy jobs are polynomials. Contemporarily, Xu *et al*. (2010) proposed two approximation algorithms for the make-span minimization on both, parallel machine and single machine, scheduling problems wherein the maintenance duration is a nondecreasing function of the total processing times of jobs preceding the maintenance operation itself. Luo *et al*. (2010) introduced two approximate approaches for the minimization of the total weighted completion time on a single machine with maintenance activity having a starting time greater than a certain deadline and a duration depending on a nondecreasing function of its starting time. Luo *et al*. (2015) and Xu *et al*. (2015) provided polynomial time algorithms for the basic single machine scheduling problem with a variable maintenance activity and a workload dependent maintenance duration, respectively. A series of

*Address correspondence to this author at the Department of Industrial Engineering, University of Catania, Via S. Sofia, 64, 95125 Catania CT, Italia; Tel. +39(0)957382457; E-mail: antonio.costa@dii.unict.it

E-ISSN: 2409-5761/18

objective function namely total completion times, maximum lateness, makespan and number of tardy jobs have been considered. As a practical example, they mentioned the steel strip manufacturing process in which steel slabs have to pass a re-heat furnace before being rolled into strips. In order to assure a regular work, the furnace has to be cleaned and the fuels must be refilled before a predetermined deadline. Since cleaning and refilling time depends on the total processing time of the steel slabs, it can be considered a valid example of variable maintenance activity.

Single machine problems can be considered as the building blocks of more complex problems. Formulations of such problems may refer to bottleneck machines or an aggregated machine system (Schmidt 2000).

Recently, Wang *et al*. (2018) presented four improved mathematical programming models for scheduling jobs and a variable maintenance activity on a single machine.

Chen (2006) developed eight mixed integer linear programming models for addressing both, single machine and parallel machine, scheduling issues with variable maintenance. In particular, two distinct mathematical models have been devised to minimize the total tardiness on a single machine environment in case of non-resumable jobs available at time zero and with no sequence dependent setup times. These two optimization approaches have been compared to demonstrate their efficiency in terms of computational times, at varying sizes of complexity.

Besides the variable maintenance issue, all the aforementioned research contributions deal with the regular single machine scheduling issue with no release times and sequence dependent setup times as well. For this reason, optimal and approximate approaches are able to solve the problem that have been devised by literature.

Pang *et al*. (2018) addressed the single machine scheduling problem with job release dates and flexible maintenance to minimize a bi-objective function based on total weighted tardiness and total completion time. The problem was observed in a semiconductor company where the flexible maintenance activity derives from the dirt accumulated during the manufacturing process. In words, cleaning of the so-called wet machines is required whenever a certain level of dirt is achieved on a given machine. The authors proposed a mixed integer linear programming model and, in addition, they introduced two heuristic algorithms and a scatter search simulated annealing (SSA) to cope up with the large-sized instances.

Since the total tardiness minimization for the single machine scheduling problem with job release dates and sequence dependent setup times is demonstrated to be NP-hard (Baker and Triesch 2001), the problem at hand, in which a variable maintenance task has to be scheduled along with the set of jobs, surely deserves the same degree of complexity.

In this paper, two distinct Mixed Integers Linear Programming (MILP) models have been presented with the aim of scheduling a set of non-resumable jobs, with release times and a sequence dependent setup times, along with a variable maintenance activity. The objective is the minimization of the total tardiness. The problem is inspired to a real-world manufacturing issue which was observed in a semiconductor company. The maintenance operation must be executed within a predetermined time window, corresponding to the availability time of the maintenance workforce. Notably, the maintenance consists of a cleaning operation as experienced by Peng *et al*. (2008) and its duration varies with the starting time. It is worthy to point out that, to the best of our knowledge, this is the first approach investigating such a challenging NP-hard scheduling issue.

The paper is organized as follows. Section one deals with the problem statement. Section two presents the MILP models. Sections 3 deals with the way the benchmark of test cases has been generated. In Section 4 the findings from the comparison analysis are commented. In the last section conclusions and future research are reported.

1. PROBLEM STATEMENT

The scheduling problem at hand can be described as follows: A set of *N* independent non-resumable jobs have to be processed on a single machine. Each job 'j' (j=1,…,n) is available at time r_j. Setup times are sequence dependent, *i.e.*, s_{ij} is the setup time of job 'j' which is being processed after job 'i'. Anticipatory setups as well as pre-emption are not allowed. The machine must be subject to a prefixed variable maintenance activity to be executed within a specific time interval. In fact, maintenance duration increases with its starting time, *i.e.*, the later it starts, the longer will be its time duration. Let p_j denote the processing time of job j, 'ZS' denote the maintenance starting time while 'ZC' is the completion time; obviously, it holds $ZS \leq ZC$. Maintenance duration '$R=f(ZS)$' is a positive nondecreasing function of its starting time. The maintenance is a flexible task so that it must be executed within a prefixed time interval [$U_{\min}$, $U_{\max}$]; thus, ZS must be greater than or equal to $U_{\min}$ and ZC must be lower than or equal to $U_{\max}$. For a certain schedule, C_j represents the completion time of job j while T_j=max[0, C_j-d_j] is the tardiness of the same job whether d_j is the corresponding due date. Setup time of the job scheduled after the maintenance operation is negligible. The objective is to place the maintenance operation within the allowed time interval as well as to find a schedule able to minimize the total tardiness. In reality, due to its variable duration, an early maintenance starting time would imply a shorter duration on one hand, but a higher risk of delay for the subsequent jobs on the other hand. Conversely, the later will be the maintenance starting time the longer will be its duration; as a result, the subsequent jobs will be subject to a higher risk of delay as well. Figure **1** depicts a twofold Gantt diagram related to two different schedules, namely $S(1)$ and $S(2)$, for a six-jobs single machine scheduling problem with variable maintenance. The two schedules are identical with exception of what happens after the maintenance starting time. As mentioned above, the maintenance duration R is a combination of a constant (γ) and a variable contribution (δ), whose value is a nondecreasing linear function of the starting time ZS. The positive increment of δ as ZS raises as it is connected with the slope α, pertaining to the linear function $y=f(ZS)$. Whether a linear increment of the maintenance activity is considered or not, the duration of the maintenance activity is $R=\gamma+tg\alpha\left(ZS-U_{\min}\right)$, while the allowed maximum starting time, hereinafter be denoted by ZS_{max}, can be computed as follows:

$$ZS_{\max}=\frac{U_{\max}-(1-tg\alpha)\cdot U_{\min}-\gamma}{tg\alpha} \tag{1}$$

Hence, the allowed maintenance time interval can be also formalized as [U_{min}, $\min(ZS_{\max},U_{\max})$].

Using the three-field classification scheme proposed by [Graham *et al*. 1979], the problem under investigation can be coded as 1,VM|r_j,s_{ij}|T, where VM indicates the variable maintenance activity.

2. MILP MODELS

Two distinct Mixed Integer Linear Programming (MILP) models have been devised for the problem under investigation. In the following paragraphs both mathematical models, hereinafter denoted as Model 1 and Model 2, are illustrated. Common notations and objective are omitted in Model 2. Most MILP approaches from the relevant literature on the same scheduling issue (Chen, 2006, Wang *et al*. 2018) consider the maintenance activity as an adding job. Model 1 disregards this conception and makes full use of some more variables. In particular, it employs two

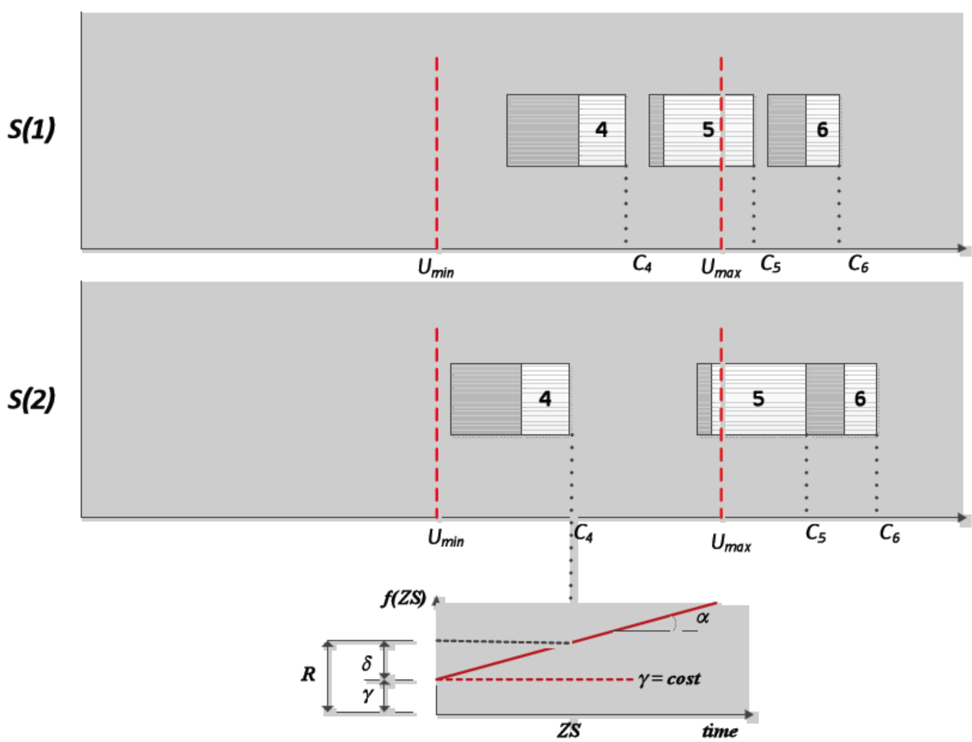

Figure 1: Example of Gantt Diagram and flexible maintenance activity.

variables, namely *W* and *V* which refer to the first and the last job to be scheduled, respectively. In addition, maintenance starting and completion times, respectively denoted as *ZS* and *ZC*, are two further variables able to strengthen this approach. On the other hand, similarly being done from the literature, Model 2 handles the variable maintenance as a dummy job to be scheduled; thus, *N*+1 jobs have to be sequenced along the time horizon. Of course, motivated by the sequence dependent setup times, Model 2 significantly differs from the mathematical models proposed by literature so far.

2.1. Model 1

Notations and Parameters

N number of jobs

$i, j = 1,2,\ldots,N$ job indexes

a_j $j = 1,2,...,N$ setup time of job *j* in case it is processed as first;

s_{ij} $j = 1,2,...,N$ setup time of job *i* processed just before job *j*;

p_j $j = 1,2,...,N$ processing time of job *j*;

r_j $j = 1,2,...,N$ release time of job *j*;

d_j $j = 1,2,...,N$ due date of job *j*;

$U_{\min}$ earliest starting time of the maintenance activity;

$U_{\max}$ dead line to accomplish the maintenance activity;

γ maintenance base time;

α slope parameter of the flexible maintenance function;

M a big number

Optimization variables

$Y_{ij} \in \{0,1\}$ $i = 1,2,...,N$ $j = 1,2,...,N$ 1 if job *j* is processed immediately after job *i*, 0 otherwise;

$X_j \in \{0,1\}$ $j = 1,2,...,N$ if job *j* is processed before the maintenance interval, 0 otherwise;

$W_j \in \{0,1\}$ $j = 1,2,...,N$ 1 if job *j* is processed as first, 0 otherwise;

$V_i \in \{0,1\}$ $j = 1,2,...,N$ 1 if job *i* is processed as last, 0 otherwise;

C_j $j = 1,2,...,N$ completion time of job *j*;

T_j $j = 1,2,...,N$ tardiness of job *j*;

ZS maintenance starting time;

ZC maintenance completion time;

Objective

Minimize $\sum_{j=1}^{N} T_j$

Subject to:

$$\sum_{j=1}^{N} Y_{ij} = 1 - Vi \quad i = 1,2,...,N \tag{1}$$

$$\sum_{i=1}^{N} Y_{ij} = 1 - W_j \quad j = 1,2,...,N \tag{2}$$

$$\sum_{j=1}^{N} X_j \geq 1 \tag{3}$$

$$C_j \geq C_i + s_{ij} + p_j - M \cdot (1 - Y_{ij}) \quad i = 1,2,...,N \quad j = 1,2,...,N \tag{4}$$

$$C_j \geq (r_j + a_j + p_j) \cdot W_{ij} \quad j = 1,2,...,N \tag{5}$$

$$C_j \geq (r_j + s_{ij} + p_j) \cdot Y_{ij} \quad i = 1,2,...,N \quad j = 1,2,...,N \tag{6}$$

$$C_j \leq ZS + M \cdot (1 - X_j) \quad j = 1,2,...,N \tag{7}$$

$$C_j \geq ZC + (s_{ij} + p_j) \cdot Y_{ij} - M \cdot X_j \quad j = 1,2,...,N \tag{8}$$

$$ZS \geq U_{\min} \tag{9}$$

$$ZC \leq U_{\max} \tag{10}$$

$$ZC \geq ZS + \gamma + tg(\alpha) \cdot (ZS - U_{\min}) \tag{11}$$

$$\sum_{i=1}^{N} V_i = 1 \tag{12}$$

$$\sum_{j=1}^{N} W_j = 1 \tag{13}$$

$$T_j \geq C_j - d_j \quad j = 1,2,...,N \tag{14}$$

$$T_j \geq 0 \quad j = 1,2,...,N \tag{15}$$

Constraint (1) states that each job must precede another job, unless it is processed as last. Constraint (2) states that each job must be preceded by another job, unless it is processed as first. Constraint (3) sets at least one job to be processed before the maintenance interval. Constraint (4) imposes each job to start after the preceding job is completed. Constraints (5) and (6) state that each job can be processed after it is released to the system. Specifically, constraint (5) is referred to the case the job processed as first, while constraint (6) works otherwise. In case a job is processed before the maintenance interval, constraint (7) states that its completion time must precede the maintenance starting time. In case a job is processed after the maintenance interval, constraint (8) states that its starting time must follow the maintenance completion time. Constraints (9) and (10) fix bounds for maintenance starting and completion time, respectively. Constraint (11) calculates maintenance duration. Constraint (12) and (13) impose one only job to be processed as first and as last, respectively. Constraint (14) calculates tardiness of jobs, which has to be non-negative according to constraint (15).

2.2. Model 2

Notations and Parameters

N number of jobs (job N + 1 being the maintenance interval);

i = 0,1,…,N + 1 preceding job index (includes dummy job 0);

j = 1,2,…,N + 1 job index;

Optimization Variables

$Y_{ij} \in \{0,1\}$ $i = 0,1,...,N+1$, $j = 1,2,...,N+1$; 1 if job j is processed immediately after job i, 0 otherwise;

I_j $j = 1,2,...,N+1$ starting time of job j;

C_j $j = 1,2,...,N+1$ completion time of job j;

Subject to:

$$\sum_{i=0}^{N+1} Y_{ij} = 1 \quad j = 1,2,...,N+1 \tag{16}$$

$$\sum_{j=1}^{N+1} Y_{ij} \leq 1 \quad i = 0,1,...,N+1 \tag{17}$$

$$\sum_{j=1}^{N} Y_{0j} = 1 \tag{18}$$

$$I_j \geq r_j + \sum_{i=0}^{N} s_{ij} \cdot Y_{ij} \quad j = 1,2,...,N \tag{19}$$

$$I_j \geq r_j + \sum_{i=0}^{N} s_{ij} \cdot Y_{i(N+1)} - M \cdot (1 - Y_{(N+1)j}) \qquad j = 1,2,...,N \tag{20}$$

$$I_j \geq C_i + s_{ij} - M \cdot (1 - Y_{ij}) \qquad i = 1,2,...,N \;\; j = 1,2,...,N \tag{21}$$

$$I_j \geq C_{(N+1)} + \sum_{i=0}^{N} s_{ij} \cdot Y_{i(N+1)} - M \cdot (1 - Y_{(N+1)j}) \quad j = 1,2,...,N \tag{22}$$

$$C_j \geq I_j + p_j \quad j = 1,2,...,N \tag{23}$$

$$C_{(N+1)} \geq I_{(N+1)} + \gamma + tg(\alpha) \cdot (I_{(N+1)} - U_{\min}) \tag{24}$$

$$I_{(N+1)} \geq C_i - M \cdot (1 - Y_{i(N+1)}) \quad i = 1,2,...,N \tag{25}$$

$$I_{(N+1)} \geq U_{\min} \tag{26}$$

$$C_{(N+1)} \leq U_{\max} \tag{27}$$

$$T_j \geq C_j - d_j \quad j = 1,2,...,N \tag{28}$$

$$Y_{(N+1)(N+1)} = 0 \tag{29}$$

$$T_j \geq 0 \quad j = 1,2,...,N \tag{30}$$

Constraint (16) states that each job must have a predecessor. According to constraint (17), each job can precede one other job at most. Constraint (18) sets at least one job to be processed before the maintenance interval. Constraints (19) and (20) state that each job can be processed after it is released to the system. Specifically, constraint (19) is referred to the case the job is preceded by a real job, while constraint (20) works in case the job comes immediately after the maintenance interval. Similarly, constraints (21) and (22) impose each job to start after the preceding job is completed, in case the preceding job is real or it coincides with the maintenance interval, respectively. Constraint (23) links starting and completion time of each job. Constraint (24) calculates maintenance completion time. Constraint (25) calculates maintenance starting time. Constraints (26) and (27) fix bounds for maintenance starting and completion time, respectively. Constraint (28) calculates tardiness of jobs. Constraints (29) imposes the maintenance interval not to precede itself. Finally, constraint (30) assigns only non-negative tardiness values to tardy jobs.

3. GENERATION OF TEST CASES

An extended benchmark of test cases has been generated to investigate how a set of parameters influences the total tardiness objective function. Similarly, as done by the other researchers [Allahverdi *et al*. 2018, Cui and Li, 2018], job processing times p_j have been extracted from a uniform distribution U[1,100], setup times from U[1,25], release dates r_j from $0.5 \times U[1, LB]$ and due dates from $U[0.25, 0.75] \times LB$, where $LB = 1.15 \times \sum p_j$. The earliest maintenance starting time is $UB_{\min} = \lfloor \beta \times LB \rfloor$, *i.e.*, the smallest integer value of the product in brackets, where β is a specific coefficient in the range (0,1). The allowed maintenance time interval [$U_{\max}$-$U_{\min}$] is fixed at 200.

A full factorial experimental plan involving four factors at different levels was arranged as in Table **1**. The number of jobs was varied at three levels while parameters α, β and γ were varied at two levels. As a result, 24 different scenario problems have been arranged. Finally, ten instances have been randomly generated for each scenario problem, according to the aforementioned criteria. Therefore, a total amount of 240 runs has been employed for analyzing the computational efficiency of the mathematical models.

4. COMPARISON ANALYSIS

The computational efficiency consists of the time each MILP model takes to converge to global optimum. Both mathematical models have been solved by IBM® Cplex Optimization Studio release 12.8.0, installed on a 24Gb Ram DELL® Workstation powered by two 2.40GHz quad-core Intel Xeon® processors. Table **2** resumes the output from both models at varying the number of jobs. The average (*ave*), maximum (*max*) and minimum (*min*) computational times (in seconds) required by each model have been reported in Table **2**. In addition, the same table depicts the standard deviation (*stdev*) and the number of times a specific model assures the minimum computational time (*Tmin*) over the provided 80 instances. Findings from Table **2** are emphasized by Figure **2**-**4**, as the number of jobs changes, respectively. Instances involving six jobs are not able to emphasize the difference between the model in terms of computational performance. Model 1 appears to be more efficient than Model 2 with exception of the *Tmin* performance indicator. It is worth pointing out as the two models needed the same time to converge for six instances out of 80. As for the 8 jobs related instances, a slight outperformance of Model 2 emerges by observing the average time and the standard deviation as well. A significant difference appears when the number of jobs to be scheduled with a variable maintenance is equal to ten. The computational efficiency of Model 2 is about two times better than that required by Model 1. The variability as the α, β and γ change is smaller for Model 2 which, according to the *Tmin* indicator, assure the minimum time to converge 73 times out of 80.

Findings from Table **2** can be graphically analyzed by the following figures. Figure **1** refers to the 6 jobs scheduling problems. Median on the computational times from Model 2 seems to be lower than Model 1 but the higher outlier related to Model 2 confirm their equivalence under the computational time viewpoint. As for the 8 jobs related problems, despite the single outlier, Figure **3** shows as Model 2 requires a smaller time to convergence. However, the computational out performance of Model 2 over Model 1 is definitely proved by Figure **4**. Figure **5** depict the interval plot at 95% confidence interval for the two models, at varying

Table 1: Experimental Plan

Factor	Level	Low	Medium	High
N	3	6	8	10
β	2	0.25	-	0.75
γ	2	0.25	-	0.75
α	2	0.25	-	0.75

Table 2: Numerical Results

N		6		8		10
Model	1	2	1	2	1	2
ave	0.24	0.22	3.7	2.8	445.5	222.2
max	1.29	1.84	6.1	6.2	972.9	446.7
min	0.03	0.05	0.9	1.0	96.5	36.1
stdev	0.19	0.22	1.3	1.1	200.9	88.8
Tmin	36	50	17	63	7	73

the number of jobs. As confirmed by the previous investigation, the difference of efficiency between Model 1 (left side) and Model 2 (right side) rises up as the number of jobs increases too. In addition, Model 2 assures a smaller dispersion around the average computational time, thus confirming a greater consistency and robust performance.

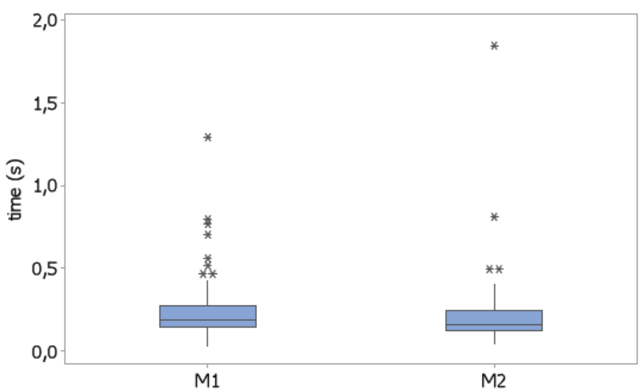

Figure 2: Computational time box plots when N=6.

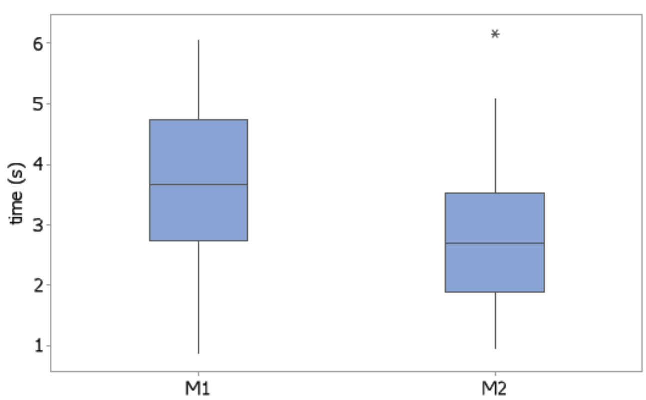

Figure 3: Computational time box plots when N=8.

CONCLUSIONS

In this paper, a single machine scheduling problem with variable maintenance activity is studied. Differently from most contribution proposed by literature so far, release times and sequence dependent setup times are included in this study. In addition, the maintenance activity must be completed within a certain due date. Two distinct Mixed Integer Linear Programming (MILP) models have been devised and tested under the computational efficiency viewpoint. The former handles the maintenance activity as an additional job with variable length to be sequenced, while the latter just manages the starting and completion time of the maintenance operation, regardless of its scheduling issue. A series of numerical analysis and graphs confirmed as Model 2 assures a better computational efficiency as the size of the problem increases. Indeed, when the number of jobs is equal to 6 the two models achieve the same performance. On the other hand, when the number of jobs increases to 10, the computational time required by Model 2 is approximately half of than that needed by Model 1 to converge to the global optimum. For future research, due to its better time efficiency, Model 2 can be used for a further analysis involving heuristic or meta-heuristic algorithms. In addition, Model 2 can be subject to further refining actions with the aim of further improving the computational efficiency towards global optima.

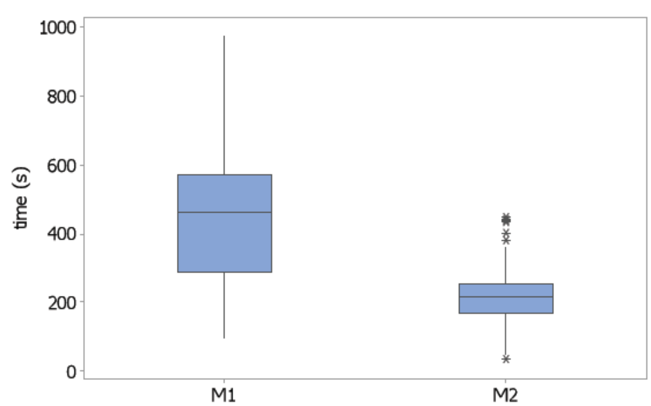

Figure 4: Computational time box plots when N=10.

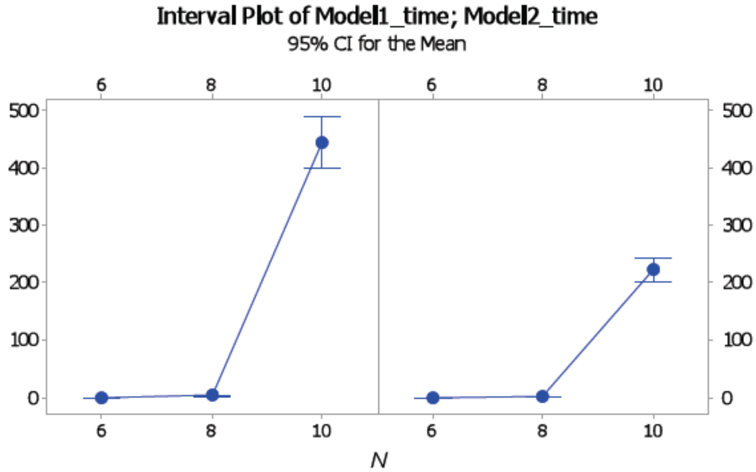

Figure 5: Computational time interval plots at varying number of jobs.

REFERENCES

[1] Allahverdi A, Aydilek H and Aydilek A. No-wait flowshop scheduling problem with two criteria; total tardiness and makespan. European Journal of Operational Research 2018; 269(2): 590-601. https://doi.org/10.1016/j.ejor.2017.11.070

[2] Baker K. R and Trietsch D. Principles of sequencing and scheduling. John Wiley & Sons 2013.

[3] Chen JS. Optimization models for the machine scheduling problem with a single flexible maintenance activity. Engineering Optimization 2006; 38(1): 53-71. https://doi.org/10.1080/03052150500270594

[4] Cui WW, and Lu Z. Minimizing the makespan on a single machine with flexible maintenances and jobs' release dates. Computers & Operations Research 2017; 80: 11-22. https://doi.org/10.1016/j.cor.2016.11.008

[5] Graham RL, Lawler EL, Lenstra JK, and Kan AR. Optimization and approximation in deterministic sequencing and scheduling: a survey. In Annals of discrete mathematics 1979; 5: pp. 287-326 Elsevier. https://doi.org/10.1016/S0167-5060(08)70356-X

[6] Kubzin MA & Strusevich VA. Planning machine maintenance in two-machine shop scheduling. Operations Research 2006; 54(4): 789-800. https://doi.org/10.1287/opre.1060.0301

[7] Luo W, Chen L & Zhang G. Approximation algorithms for scheduling with a variable machine maintenance. In International Conference on Algorithmic Applications in Management 2010; (pp. 209-219). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-14355-7_22

[8] Luo W, Cheng TE & Ji, M. Single-machine scheduling with a variable maintenance activity. Computers & Industrial Engineering 2015; 79: 168-174. https://doi.org/10.1016/j.cie.2014.11.002

[9] Ma Y, Chu C, & Zuo C. A survey of scheduling with deterministic machine availability constraints. Computers & Industrial Engineering, 2010; 58(2): 199-211. https://doi.org/10.1016/j.cie.2009.04.014

[10] Mosheiov G, & Sidney JB. Scheduling a deteriorating maintenance activity on a single machine. Journal of the Operational Research Society 2010; 61(5): 882-887. https://doi.org/10.1057/jors.2009.5

[11] Pang J, Zhou H, Tsai YC & Chou FD. A scatter simulated annealing algorithm for the bi-objective scheduling problem for the wet station of semiconductor manufacturing. Computers & Industrial Engineering 2018; 123: 54-66. https://doi.org/10.1016/j.cie.2018.06.017

[12] Schmidt G. Scheduling with limited machine availability. European Journal of Operational Research 2000; 121(1): 1-15. https://doi.org/10.1016/S0377-2217(98)00367-1

[13] Wang Q, Liu A & Xiao J. On exact algorithms for single-machine scheduling problems with a variable maintenance. Computers & Industrial Engineering 2017; 107: 276-279. https://doi.org/10.1016/j.cie.2017.03.024

[14] Xu D, Yin, Y & Li H. Scheduling jobs under increasing linear machine maintenance time. Journal of scheduling 2010; 13(4): 443-449. https://doi.org/10.1007/s10951-010-0182-0

[15] Xu D, Wan L, Liu A, & Yang DL. Single machine total completion time scheduling problem with workload-dependent maintenance duration. Omega 2015; 52: 101-106. https://doi.org/10.1016/j.omega.2014.11.002

Received on 27-11-2018 Accepted on 21-12-2018 Published on 31-12-2018

DOI: http://dx.doi.org/10.15377/2409-5761.2018.05.5