Int j simul model 18 (2019) 4, 666-677
ISSN 1726-4529 Original scientific paper

JOINT PROGRAMMING OF PRODUCTION-MAINTENANCE TASKS: A SIMULATED ANNEALING-BASED METHOD

Diaz Cazanas, R.[*]; Delgado Sobrino, D. R.[**]; Caganova, D.[**]; Kostal, P.[**] & Velisek, K.[**]
[*] Central University "Marta Abreu" de Las Villas, Faculty of Mechanical and Industrial Engineering, 5½ km Camajuaní road, 54 830, Santa Clara, Villa Clara, Cuba
[**] Slovak University of Technology, Faculty of Materials Science and Technology, J. Bottu 25, 91724 Trnava, Slovak Republic
E-Mail: daynier_sobrino@stuba.sk

Abstract
Integrated production-maintenance programs are an attractive area given their practical benefits and complexity. This has motivated the development of approximate methods providing good solutions in a reasonable time. This paper introduces a heuristic method for the joint programming of production-preventive maintenance tasks in an environment of identical parallel machines minimizing the makespan. Unlike other proposals, this assumes the time between interventions is unknown, and that the start time of these constitutes a decision variable. It also considers that not all jobs may be available to begin processing at the beginning of periods. The heuristic possesses two phases; the first generates an initial solution based on the LPT dispatch rule with a slight modification allowing it to consider the randomness of equipment failures and their impact on the task's execution time. A dispatch rule called rj-LPT is proposed complying with one of the heuristics' steps. The second phase is based on the Simulated Annealing approach that helps refining the initial solution found.
(Received in September 2019, accepted in October 2019. This paper was with the authors 1 week for 2 revisions.)

Key Words: Production and Maintenance Programming, Preventive Maintenance, Heuristic Method, Longest Processing Time Rule (LPT), Pseudo-Code, Simulated Annealing

1. INTRODUCTION

The problems of joint-programming in operations management involve the need of making decisions related, among others, with the design of joint production and preventive maintenance (PM) programs. Management approaches have traditionally addressed these areas separately, which has been proven to lead to suboptimal solutions [1-10].

On the other hand, the makespan minimization is a key objective for properly balancing the use of resources, which is especially valid in productive configurations with parallel machines, [11]. However, given the complex nature of this type of problem it becomes difficult obtaining a good solution in a reasonable time using exact methods and, thus, the use of heuristics has been a successful and more viable option lately. In its traditional deterministic version, i.e.: without considering the randomness of the jobs´ processing times due to equipment failures, the problem of minimizing the makespan in identical parallel machines has found some solution strategies based on algorithms such as First Fit Decreasing and Best Fit Decreasing referred in [12, 13], also in the well-known LPT dispatch rule [14]. Similarly, years later [15] proposed the MULTIFIT algorithm with a performance ratio even better than same LPT approach.

From the 90s several authors begin to consider the equipment capacity limitations in the creation of production programs due to maintenance. In the case of a parallel machines setting minimizing the makespan, authors like [16, 17] made important contributions. The first considered a relatively long programming horizon that included several preventive interventions, where the time between these was not constant, however deterministic. Lee and Wu [17] considered a single maintenance period within a planning horizon of known starting

and ending instants for each machine, and the processing time of the jobs was considered a function of their own starting time (deteriorating jobs).

Also, over the last years many of the solution strategies for this type of problem have been leaning towards the use of local search algorithms, artificial intelligence and the use of simulation [18]. For example, Mirabedini and Iranmanesh [19] applied a multiobjective dynamic genetic algorithm and an optimization scheme based on a swarm of particles as alternative solution methods for a model that sought to minimize the makespan and other objectives. Wang and Liu [20] investigated the parallel machines scheduling problem with flexible preventive maintenance activities to optimize makespan and unavailability of machine systems. The problem was represented by a non-linear mathematical model and for the solution they designed an elitist multiobjective genetic element (NSGA-II). In this same regard, Da et al. [4] also developed a genetic algorithm of the type NSGA-II as a solution method for an integrated model that includes uniform parallel machines. However, along with the increasing use of local search methods, constructive heuristics have continued to be a key and tempting alternative for implementing simple methods generating good solutions quickly. Besides, these have also proven useful as part of hybrid methods usually made up of simple constructive heuristics whose results are further explored and improved by such search algorithms.

In other to contribute solving the above-mentioned gaps and problematic issues, the paper presents a hybrid method consisting of two phases that aims at generating joint programs of production orders and PM interventions in an environment of identical parallel machines, minimizing the makespan (C_{max}). Unlike others, the proposal considers that different jobs may have different release times, and that the starting time of PM is a decision variable within the algorithm. In the first phase, we propose a new constructive heuristic that generates an initial solution that is to be improved in a second phase through a Simulated Annealing-based approach also developed by the authors. The effectiveness evaluation of the proposals is realized through its comparison with 5 alternative methods that also address this type of problem. This validation strategy was specifically conceived for the purposes of this paper.

2. MATERIALS AND METHODS

2.1 Definition of the problem

The productive system consists of a set of *m* identical machines, arranged in parallel, capable of processing *n* types of products. Each product is processed in one and only one of the machines. The processing time of the jobs is assumed constant and known, and once a job has started it won't be interrupted. We assume that the time between equipment failures follows a Weibull probability distribution. The proposed algorithm assigns jobs to the different machines, besides indicates the most convenient time to carry out the PM interventions, in order to minimize the makespan. It is considered a policy of PM of the type as good as new.

2.2 Definition of the variables and parameters of the heuristic algorithm

n: Number of jobs (products), *m:* Number of machines, *J:* Set of jobs (products) to be programmed, *M:* Set of machines (productive resources)
r_j: Time instant where the job *j* is available to start being processed
p_j: Processing time of the job *j*
t_{pk}: Average duration time of the PM interruption in the machine *k*
t_{rk}: Average duration time of the corrective maintenance interruption in the machine *k*
β_k: Weibull shape parameter for the variable "time between failures" linked to the machine *k*
η_k: Weibull scale parameter for the variable "time between failures" linked to the machine *k*

j_l, j_t: Subscripts used to denote the l^{th} or the t^{th} job inside a set of arrays ordered according to a defined criterium
I_k: Number of jobs assigned to the machine k
Tt_k: End time for the machine k
a_{0k}: Effective age of the machine k at the beginning of the programming period
$a_{k(lk^-)}$: Effective age of the machine k before starting the processing of the job $l_{(k–th)}$
$a_{k(lk)}$: Effective age of the machine k after ending the processing of the job $l_{(k–th)}$
TCE_{lk}: Expected completion time of the job l in the machine k
TE_{lkPM1}: Processing time of the job l in the machine k in the case this is subjected to a PM intervention before processing the job $l_{(k–th)}$
TE_{lkPM0}: Processing time of the job l in the machine k in the case this is not subjected to a PM intervention before processing the job $l_{(k–th)}$ *tce*
PM_{lk}: Binary variable indicating if the PM intervention is to take place on the machine k before initiating the job l
$A(k)$: Array representing the job subsets assigned to the machine k
w_{lt}: Specific weight of the l^{th} job respect to the t^{th} job
ω: Positive constant defined experimentally to decide if it is convenient to first program a job with a lower release date and smaller p than another with higher values of these.
C_{max}: Makespan value associated to the assignment of jobs and maintenance interventions assumed as the solution.

2.3 Description of the algorithm

Our algorithm is composed of 2 phases. The first phase obtains an initial solution through the design of a constructive heuristic (CH) based on the LPT dispatch rule. The second one improves the initial solution through a Simulated Annealing approach. The LPT rule has been slightly modified in order to consider the stochastic character of the problem and the release date of the jobs. These modifications are: (1) The next job to be programmed will be assigned to that machine that achieves for it a shorter expected completion time. (2) If more than one machine achieves the same minimum value of expected completion time for a job, the one with the lowest effective age before starting the job will be selected. (3) It is evaluated the convenience of delaying the processing start time of a job with a shorter processing time than another one of longer processing time that is not available at the current time instance.

Phase 1: Generation of the initial solution through of a constructive heuristic

1. Initialize: $Ttk = \boldsymbol{I_k} = 0\ \forall\ k \in M$; $\boldsymbol{l} = 0$; $A(1) = A(2) = \ldots = A(m) = \emptyset$; $\boldsymbol{r_j}\ \forall\ j = \{1, 2, \ldots, n\}$; $a_{0k} \forall\ k = \{1, 2\ldots, m\}$, ω.
2. Generate the set of jobs J ordered in a non-increasing way according to p_j
3. Make: $t = 1$; $S = \emptyset$; $P = \emptyset$; $l = l + 1$

If $r_{jl} \leq max\left(min(r_{jl}), min(Tt_k)\right) \forall j_l = \{l, l+1, \ldots, n\}$, $\forall\ k \in M$ assign the job j_l to the machine k $|TCE_{lk} = min\ \{TCE_{lk}\}\ \forall\ k \in M$. In case of a match, select the machine $k\ |\ a_{k_{(lk^-)}} = min\ \{a_{k_{(lk^-)}}\}\ \forall\ k\ |TCE_{lk} = min\{TCE_{lk}\}$. If the tie persists, then select the machine k arbitrarily.
$I_k = I_k + 1$

$$TCE_{lk} = \begin{Bmatrix} max\ (Tt_k;\ r_l) + TE_{lkPM1} & \text{if } PM_{lk} = 1 \\ max\ (Tt_k;\ r_l) + TE_{lkPM0} & \text{otherwise} \end{Bmatrix} \quad (1)$$

$$PM_{lk} = \begin{Bmatrix} 1 & \text{if } \left(t_{pk} + t_{rk}\left[\left(\frac{p_l}{\eta_k}\right)^{\beta_k}\right]\right) < \left(t_{rk}\left[\left(\frac{a_{k(lk^-)} + p_l}{\eta_k}\right)^{\beta_k} - \left(\frac{a_{k(lk^-)}}{\eta_k}\right)^{\beta_k}\right]\right) \\ 0 & \text{otherwise} \end{Bmatrix} \quad (2)$$

$$TE_{lkPM1} = p_l + t_{pk} + t_{rk}\left[\left(\frac{p_l}{\eta_k}\right)^{\beta_k}\right] \quad (3)$$

$$TE_{lkPM0} = p_l + t_{rk}\left[\left(\frac{a_{k(lk^-)} + p_l}{\eta_k}\right)^{\beta_k} - \left(\frac{a_{k(lk^-)}}{\eta_k}\right)^{\beta_k}\right] \quad (4)$$

$$a_{k(lk^-)} = \begin{cases} 0 & \text{if } PM_{lk} = 1 \\ a_{k(lk^-1)} + p_{lk^-1} & \text{if } PM_{lk} = 0 \text{ and } I_k > 1 \\ a_{0k} & \text{if } I_k = 1 \end{cases} \quad (5)$$

Make $Tt_k = TCE_{lk}$ and $A(k) = A(k) \cup \{j_l\}$

If $PM_{lk} = 1$ then program a PM intervention in the machine k before starting the job j_l.

Program the start time (t_{ilk}) and the expected completion time (TCE_{lk}) for the job j_l in the machine k using the Eqs. (6) and (1) respectively.

$$t_{ilk} = \begin{cases} max\,(Tt_k;\, r_l) & \text{if } PM_{lk} = 0 \\ max\,(Tt_k + t_{pk};\, r_l) & \text{otherwise} \end{cases} \quad (6)$$

If $\sum_{k=1}^{m} I_k < n$ repeat step 3 otherwise obtain $C_{max} = max\,(Tt_k)\ \forall\ k \in M$

If $r_{jl} > max\left(min(r_j), min(Tt_k)\right) \forall\ j = \{l, l+1, \ldots, n\}, \forall\ k \in M$ make $Tt_k' = Tt_k$, $I_k' = I_k$, $a_{k(lk^-)}' = a_{k(lk^-)}$, $A_k' = A_k \forall\ k \in M$ and move to step 4.

4. Generate the subset of jobs P ordered in a non-decreasing way according to r_j; $P \subset J \setminus \{j_l;\ A(k)\ \forall\ k \in M\} \mid r_l > r_t \forall j_t \in P$.

4.1. Calculate the relative importance of the job j_l with respect to the job j_t (w_{lt}) using Eq. (7).

$$w_{lt} = \frac{\frac{p_l}{\bar{p}}}{r_{jl} - max\left(min(r_{jt}), min(Tt_k)\right)} \quad \forall\ k \in M, j_t \in P \quad (7)$$

$$\bar{p} = \frac{\sum_{\forall j \in P} p_j}{\|P\|} \quad (8)$$

If $w_{lt} < \omega$ move to step 4.2 otherwise to step 4.3.

4.2. Calculate the expected completion time of the job j_t for each machine k using the Eq. (1). Identify the machine $k \mid TCE_{tk} = min\,\{TCE_{tk}\}$. In case of a match, select arbitrarily.

If $w_{lt} < \omega$ *or* $TCE_{tk} \le r_{jl}$

Set $Tt_k = TCE_{tk}$; $I_k = I_k + 1$; $S = S \cup \{j_t\}$ else move to step 4.3

4.3. Make $t = t + 1$ *If* $t \le \|P\|$ and $r_{jl} > min(Tt_k)\ \forall\ k \in M$ go to step 4.1, otherwise to step 5.

5. If $S \neq \emptyset$ make $Tt_k = Tt_k'$, $I_k = I_k'$, $a_{k(lk^-)} = a_{k(lk^-)}'$, $A_k = A_k'$ and assign the jobs from the subset S using the combined rule r_j– *LPT*. If $S = \emptyset$ move to step 6

6. Make $Tt_k = Tt_k'$, $I_k = I_k'$, $a_{k(lk^-)} = a_{k(lk^-)}'$, $A_k = A_k'$ and assign the job j_l to the machine $k \mid TCE_{lk} = min\ \{TCE_{lk}\}\ \forall\ k = \{1, 2, \ldots, m\}$. In case of a match select the machine $k \mid a_{k(lk^-)} = min\{a_{k(lk^-)}\}\ \forall$ k $\mid TCE_{lk} = min\{TCE_{lk}\}$. If the tie persists, then select the machine k arbitrarily.

Set $A(k) = A(k) \cup \{j_l\}$ *and* $Tt_k = TCE_{lk}$.

If $PM_{lk} = 1$ then program a PM intervention in the machine k before starting the job j_l

Program the start time (t_{ilk}) and the expected completion time (TCE_{lk}) for the job j_l in the machine k using the Eqs. (6) and (1) respectively.

If $\sum_{k=1}^{m} I_k < n$ move to step 3, otherwise obtain $C_{max} = max\,\{Tt_k\}\ \forall\ k \in M$ and stop.

Definition of the pseudo-code for the proposed combined rule rj- LPT

1. Set $t_o = max\left(min(r_j), min(Tt_k)\right) \forall\ j \in S, k \in M$, $JA = \emptyset$, $MA = \emptyset$
2. Set $JA \subset S \mid r_j \le t_o \forall\ j \in S$
3. Set $MA \subset M \mid Tt_k \le t_o \forall\ k \in M$
4. If $\|JA\| \le \|MA\|$ assign each job of JA to some machine k of MA on the time instant t_o, otherwise assign these jobs according to the LPT rule.

Update the arrays $A(k)$ and $I(k)\ \forall\ k \in MA \mid \exists\ j \in A_k$. Set $Tt_k = TCE_{jk} \mid j \in A(k)\ \forall\ k \in MA$.

Evaluate the variable PM_{lk} in each machine k before assigning the job j. If $PM_{lk} = 1$, then program PM intervention in the machine k before starting the job j_l.

Program the start time (t_{ilk}) and the expected completion time (TCE_{lk}) for the job j_l in the machine *k* using the Eqs. (6) and (1) respectively.

5. Set $S = S \setminus j \in JA \mid j \in A(k) \; \forall \; k \in MA$. If $S \neq \emptyset$ go to step 1 otherwise stop.

The first 2 steps of CH follow the general approach of the LPT rule, seeking to assign those jobs with the longest duration first. In steps 3 and 4, the release dates of the jobs are considered. Eq. (2) summarizes a greedy heuristic deciding if carrying out a PM intervention or not before starting the processing of the job being considered. If the job with longest processing time were not have the lowest release date, it is evaluated, through step 4 and Eq. (7) , if it is convenient to assign other jobs that have a lower release date first. Those jobs that fulfil the condition set by Eq. (7) will be assigned in step 5 using the combined r – LPT rule proposed within this paper. If all jobs that were convenient to be programmed before l^{th} longest job have been already assigned or if it was decided that it would be convenient to delay the processing start time of these, then it proceeds to program the l^{th} longest job as indicated in step 6.

Phase 2: Improvement of the initial solution using a Simulated Annealing approach

Our Simulated Annealing-based algorithm is composed of 3 functions: a function that defines the strategy to generate a neighbour solution from the current one (neighbord function), an annealing function (schedules generation), and the objective function (fitness function).The initial solution to start the search process is obtained from the CH defined in section 2.3.

Function for the generation of neighbouring solutions (neighbord function)

Consider the *Sch_neighb* matrix where each row represents the vector defining the initial solution obtained for the machine corresponding to the row number. The strategy adopted to generate a neighbouring solution is of the interchange type where a pair of machines m_1 and m_2 are selected randomly. The machine *k* with the longest expected end time (Tt_k) in the current solution will be more likely to be selected as m_1; while the machine *i* that has the shortest Tt_i will be more likely to be selected as m_2. Two cells, one in the row of m_1 and one in the row of m_2, are also selected randomly. If the element (cell) selected in the row of m_1 is a PM activity, this is transferred, with a probability of 0.5, to the position (cell) of the row of m_2 previously selected, as long as the element of the cell of m_2 is also job. If on the contrary, the selected item in the row of m_1 is a job, this is transferred to the first empty position available in the row of m_2, and a PM activity is inserted in the previous cell with a probability of 0.5. The pseudocode that defines the neighbord function has been implemented in MATLAB as it appears bellow:

1. Set *Sch_neighb*: matrix storing the initial solution obtained from the previous algorithm.
   Tt: vector containing the end time for the machine set
   t_1 = *randi* (*n*)
   Set *x* = *round* (*rand*)
2. if *x* == 1
   $m_1 = m_k \mid Tt_k = max(Tt_k) \; \forall \; k \; \in \; M$
   else
   m_1 = *randi* (*m*);
   end
3. Set *x* = *round* (*rand*);
   if x == 1
   $m_2 = m_i \mid Tt_i = min(Tt_i) \; \forall \; i \; \in \; M$
   else
   m_2 = *randi*(*m*);
   end
   if $m_2 = m_1$ repeat step 3 else move to step 4.
4. Set *j* = *randi* (*n*)

5. if *sch_neighb* (m_1, t_1) == *PM*
   if *sch_neighb* (m_2, *j*) ~= *PM* and *j* == 1
   Set *d* = *round* (*rand*)
   if *d* == 1 insert the PM activity in the cell (m_2, *j*) and update the positions in this row.
   end
   end
   if *sch_neighb* (m_2, *j*) ~= *PM* and *j* > 1 and *sch_neighb* (m_2, *j*-1) ~= *PM*
   Set *d* = *round* (*rand*)
   if *d* ==1 insert the PM activity in the cell (m_2, *j*) and update the positions in this row.
. end
   end
   if *sch_neighb* (m_2, *j*) ~= *PM* and *j* > 1 and *sch_neighb* (m_2, *j*+1) ~= *PM* and *j* < *n*
   Set *d* = *round* (*rand*)
   if *d* == 1 insert the PM activity in the cell (m_2, *j*) of *sch_neighb* and update positions
   end
   end
   5.1 Remove the PM activity from the cell (m_1, t_1) and update the positions in this row.
   else
   set *d* = *round* (*rand*).
   if *d* == 1 insert the PM activity after the last job in row m_2 and, in the next position, the job located in the cell (m_1, t_1), else insert the job located in (m_1, t_1) as the last job in m_2
   end
   5.2 Update the positions of row m_1 making sure that there are no PM activities remaining in adjacent positions. If this happens, eliminate one of these.
   end

Annealing function (schedules generation)

This function uses the solution obtained by the neighbord function, expressed in the matrix Sch_neighb, and makes successive changes to this solution in an amount determined by the temperature parameter. The solution obtained also constitutes a neighbour solution, but with a different structure. The pseudocode that defines this function is shown below:

Set *sch_neighb = optimValues.x*;
for *i* = 1*: floor*(*optimValues.temperature*)+1
[*machorders*] = *size*(*sch_neigbh*);
sch_neighb = neighbor (*sch_neighb, mach, orders*);
end

Objective function (fitness function)

This function uses some variables and parameters defined in the CH; this is its pseudocode:

Set *Tm = I = TElk = zeros* (1, *m*), $a klk-= a klk = TCElk$ = *zeros* (*m*, *n*)
for *k* = 1: *m*
for l = 1: *n*
if l == 1 and *sch_neighb* (*k*, *l*) ~= *PM*
set $a_{k_{(lk^-)}}$ = initial age of machine *k* before processing the job *j*
$I_k = I_k + 1$, $a_{k_{(lk)}} = a_{k_{(ij^-)}} + p_l$, $TE_{lk} = TE_{lkPM0}$
end
if l > 1 and sch_neighb (*k*, l) ~= *PM* and *sch_neighb* (m_2, *j*-1) == *PM*
$I_k = I_k + 1$, $a_{k_{(lk^-)}} = 0$, $a_{k_{(lk)}} = a_{k_{(lk^-)}} + p_l$, $TE_{lk} = TE_{lkPM1}$
end
if l > 1 and *sch_neighb* (*k*, l) ~= *PM* and *sch_neighb* (m_2, *j*-1) ~= *PM*
$I_k = I_k + 1$, $a_{k_{(lk^-)}} = a_{k_{(lk-1)}}$, $a_{k_{(lk)}} = a_{k_{(lk^-)}} + p_l$, $TE_{lk} = TE_{lkPM0}$

end
$TCE_{lk} = max\ (Tt_k;\ r_l) +\ TE_{lk}$, $Tt_k = TCE_{lk}$
end
end
Set $C_{max} = max\ (Tt_k)$

3. APPLICATION OF THE ALGORITHM TO A CASE STUDY

3.1 Main characteristics of the production system

It´s a workshop of the plastic industry with 7 identical injection machines arranged in parallel. The programming period was of 3 months and included 29 products. There are 3 shifts of 8 hours each daily. Table I shows the processing time and release date of the products.

Table I: Processing time and release date of the jobs to be programmed.

Products	**Processing time** (h/u)	**Lot size** (u)	**Production time** (h)	**Release date** (h)
1. Cabinet handle	0.0054	61222	360.60	0
2. Decoration of the handle FOK	0.0071	16198	145.01	0
3. Decoration of the handle WYA 40	0.0071	13463	125.59	0
4. Decoration of the handle YBD40A	0.0071	6337	74.99	72
5. Decoration YBXB40-80C	0.0071	2988	51.21	0
6. Support board	0.0069	56301	418.48	96
7. Support board cover	0.0017	122000	237.4	0
8. Power outlet cover	0.0016	217567	378.11	120
9. Measuring vessel	0.0153	26745	439.20	0
10. Grounded power outlet base	0.0014	184570	288.40	0
11. Propeller 12' (Spare part)	0.0195	42022	849.43	48
12. Time adjustment button	0.0058	3976	53.06	0
13. Locator YBXB40-80C	0.0190	25736	518.98	168
14. Box 2×4 house	0.0072	94068	707.29	264
15. Top lug of the rice cooker	0.0064	31293	230.27	210
16. Box 4×4 house	0.0072	43857	345.77	0
17. Lateral lid	0.0055	33084	211.96	480
18. Tank lid CK	0.0058	16804	127.46	210
19. Base Column VE 79	0.0207	30694	665.37	240
20. Protective nut Vietnam	0.0043	19167	112.42	0
21. Base VE 79	0.0156	6002	123.63	288
22. Base lid VE 78	0.0173	8310	173.76	120
23. Propeller VE 79	0.0229	19056	466.38	360
24. Propeller 16' (Spare part)	0.0229	792	48.14	0
25. Drawer for vegetables	0.0229	2740	92.75	0
26. Front panel	0.0172	2289	69.37	0
27. Freezer frame	0.0229	1704	69.02	0
28. Upper panel	0.0533	1740	122.74	240
29. Protective ring VE 79	0.0110	14412	188.53	72

Data of the time between failures (*TBF*) and time to repair (*TTR*) were collected for each machine (more than 30 per machine). The goodness-of-fit tests performed (Kolmogorov-Smirnov, Anderson-Darling and Chi square) reaffirmed the hypothesis that the data fit the Weibull distribution with 5 % significance. Table II contains information referring to the shape and scale parameters of the Weibull distribution for *TBF* and *TTR*, the average time dedicated to PM ($\bar{t_p}$), the average time of repair ($\bar{t_r}$), the time between preventive interventions ($\bar{t_i}$) according to the policy of preventive maintenance adopted in the company, the optimal time between preventive interventions (t_i^*), and the effective age of each machine

at the beginning of the period (E_0). The average repair time for each machine was obtained from Eq. (9).

$$\overline{t_r} = \eta * \Gamma(1 + \frac{1}{\beta}) \quad (9)$$

where:

$\overline{t_r}$: Average time of repair after the occurrence of the failure (Corrective Maintenance).
η: Scale parameter of the Weibull distribution corresponding to variable *TTR*.
β: Shape parameter of the Weibull distribution corresponding to variable *TTR*.
Γ: Gamma distribution function.

The optimal time between preventive interventions (t_i^*) was calculated with the aim of maximizing equipment availability. The Eq. (10) used in [21] was used for these purposes.

$$t_i^* = \eta * (\frac{\overline{t_p}}{\overline{t_r} * (\beta - 1)})^{\frac{1}{\beta}} \quad (10)$$

Table II: Maintenance parameters of the production equipment.

No.	Machines	β_{TBF}	η_{TBF} (h)	$\overline{t_p}$ (h)	β_{TTR}	η_{TTR} (h)	$\overline{t_r}$(h)	$\overline{t_i}$ (h)	t_i^*(h)	E_0 (h)
1	TTI-130-1	2.06	585.97	33.45	1.42	127.62	116.06	400	311.41	88
2	TTI-130-2	2.20	911.78	33.45	1.41	135.72	123.56	400	463.43	96
3	TTI-300	1.87	397.64	33.45	1.29	61.43	56.82	400	322.69	105
4	IJ-300	2.03	1318.53	35.54	1.42	73.99	67.29	550	948.86	368
5	EM-300	5.67	2002.27	27.29	1.53	22.42	20.19	385	1608.94	339
6	EM-480	1.63	952.41	27.29	1.29	36.64	33.89	385	1107.18	365
7	JM-650	2.59	1454.74	27.29	1.36	97.57	89.35	425	515.45	407

3.2 Solution obtained through the constructive heuristic of phase 1

The solution provided by the CH, implemented in MATLAB R2015a, is shown in the matrices Schedule and *TTj*.

Schedule = Columns 1 through 8

```
   2    25   100    13   100    22   100    24
   7    26    12   100     6   100    17     4
   3    20    27   100     8   100    15     5
  10   100    11     0     0     0     0     0
   9    23    29    18     0     0     0     0
  16    14    28     0     0     0     0     0
   1   100    19    21     0     0     0     0
```

TTj = 1000 *

```
Columns 1 through 10
 0.0090   0.0010   0.0160   0.0100   0.0070   0.0020   0.0030   0.0110   0.0200   0.0250
      0        0        0        0        0        0        0   0.3352   0.1414   0.1600
 0.4393   0.3744   0.3597   0.2997   0.2500   0.1600   0.1414   1.2122   0.2764   0.2701

Columns 11 through 20
 0.0260   0.0140   0.0190   0.0130   0.0270   0.0120   0.0230   0.0060   0.0080   0.0150
 0.2500   0.3597   0.4017   0.3035   0.2764   0.3264   0.4393   0.4193   0.3965   0.8598
 0.3264   1.1108   1.0788   0.9129   0.3630   0.3858   0.9069   0.8600   0.8263   1.1105

Columns 21 through 29
 0.0170   0.0290   0.0220   0.0180   0.0210   0.0280   0.0040   0.0050   0.0240
 0.8935   0.9069   0.9463   1.0971   1.0788   1.1108   1.1104   1.1105   1.1630
 1.1104   1.0971   1.1296   1.2265   1.2230   1.2429   1.2142   1.1934   1.2118
```

The value of the objective function (C_{max}) for this solution is C_{max} = 1242.94 h.

The schedule matrix shows the assignment of each job to each machine. Rows correspond to each machine, in the order they appear in Table II. The values correspond to the id of each job, in the order in which they appear in Table I. The value 100 indicates a PM activity. According to the solution from above, machine 1 will first process the job 2, then 25, a PM intervention and so on till ending with the job 24. In this case the value of the constant ω was adjusted to 0.5. The *TTj* matrix shows the id of each job in row 1, in the order in which they were assigned. Rows 2 and 3 show the expected start and end times for each job respectively.

3.3 A synthetized take on the results obtained by applying the hybrid method (CHSA)

The solution provided by CH was taken as the starting solution for the proposed hybrid algorithm (CHSA). The parametrization of the Simulated Annealing algorithm was defined as follows: initial temperature equal to 1000 (a relatively high value for expanding the search space); definition of the search area using neighbord function and schedules generation; equilibrium condition defined by adjusting to 100 the reannealing interval, exponential cooling scheme, with an $\alpha = 0.95$; stopping criterion corresponding to 5000 iterations. The results of C_{max}, corresponding to 50 runs implemented in MATLAB, is shown in Table III.

4. VALIDATION STRATEGY FOR THE PROPOSED CHSA

Since the CH is based on a modified version of the LPT dispatch rule, an appropriate way to demonstrate the effectiveness of our proposal is: 1. to analyse the effectiveness of such proposed modifications in the CH design, and 2. to analyse the convenience of generating integrated production and preventive maintenance programs instead of generating these independently. In this sense we propose and use 5 alternative methods to address the problem of integrated programming that lead us to key comparisons and conclusions with respect to our proposal. These methods were also encoded in MATLAB and applied to the case study; their essence and results obtained are shown next:

1. Production Programming model without preventive maintenance. The jobs are programmed following the FIFO rule, i.e.: the one with the lowest release date is programmed first, and in case of a tie, the LPT rule is applied. Here PM interventions are not considered but a corrective maintenance strategy is adopted instead. The resulting value is $C_{max} = 1291.4$ h.

2. Interrelated model of production – preventive maintenance programming. This alternative consists in the design of two independent optimization models, one for production and the other for maintenance. In this interrelated model, the start instants of PMs obtained for each machine according to an optimal maintenance program (see Table II) are inserted as constraints to the production programming model previously obtained. The structure of the solution included 13 PM interventions and yielded a $C_{max} = 1272.4$ h.

3. Integrated model of production and preventive maintenance programming based on the combined r-LPT rule. In this model, the next job to assign will be the one with the highest release date, and if there is more than one, the one with the longest processing time will be selected. As with the LPT rule, the job to be started will do so in the first available machine and not in the one that achieves a lower TCE for it. By means of this method it is possible to analyse the feasibility of the rule that is summarized in expression 7 and step 4.1 of the CH, which constitutes one of the differences of this our proposal with respect to the LPT dispatch rule. The structure of the solution included 9 PM interventions and yielded a $C_{max} = 1257.4$ h.

4. Integrated model of production and preventive maintenance programming based on the LPT rule. This method is similar to the proposed CH heuristic, with the difference that the considered job will be programmed in the machine that is first available. Unlike the alternative method 3, this model will evaluate the convenience of delaying the start of a job as

indicated in step 4.1 of the CH. On the other hand, this alternative allows analysing the effect of starting the next job on the machine that provides the lowest TCE for such job, and not the first available. The structure of the solution included 7 PM interventions and yielded a C_{max} = 1247.3 h.

5. Standard Simulated Annealing algorithm as a joint programming solution method. To demonstrate the effectiveness of including our CH heuristic as the initial phase of CHSA, this problem was addressed by using a standard Simulated Annealing algorithm (SA), where the initial solution was to be generated randomly. Due to such random nature, as in CHSA, 50 runs of 5000 iterations were made. The parameters adjusted values were the same as in CHSA. Table III shows the result obtained for the makespan (C_{max}) in each run for both methods.

Table III: C_{max} values obtained in each of the 50 runs made with the SA and CHSA algorithms.

Run	C_{max}**_SA** (h)	C_{max}**_CH SA** (h)	**Run**	C_{max}**_SA** (h)	C_{max}**_CH SA** (h)	**Run**	C_{max}**_SA** (h)	C_{max}**_CH SA** (h)
1	1398.39	1242.90	18	1376.72	1242.35	35	1384.81	1242.90
2	1400.73	1242.90	19	1397.22	1242.90	36	1413.61	1242.90
3	1345.05	1238.48	20	1424.38	1238.57	37	1447.89	1238.48
4	1364.14	1242.35	21	1379.41	1242.90	38	1458.08	1242.90
5	1414.81	1242.90	22	1336.85	1241.82	39	1364.34	1242.90
6	1364.84	1242.90	23	1380.68	1242.90	40	1427.11	1242.90
7	1396.69	1238.48	24	1422.10	1242.90	41	1464.72	1242.35
8	1390.61	1242.90	25	1434.41	1242.90	42	1443.34	1242.90
9	1344.43	1242.90	26	1407.68	1238.48	43	1364.47	1238.48
10	1394.52	1241.82	27	1361.77	1241.82	44	1370.00	1242.90
11	1349.75	1238.48	28	1444.11	1242.90	45	1368.44	1242.90
12	1417.38	1238.48	29	1368.29	1242.90	46	1351.66	1238.48
13	1434.41	1242.35	30	1375.39	1238.57	47	1389.92	1242.90
14	1434.41	1242.90	31	1375.34	1242.90	48	1432.24	1238.57
15	1442.60	1242.90	32	1338.37	1242.90	49	1413.92	1242.90
16	1434.41	1242.90	33	1404.36	1238.57	50	1363.21	1242.90
17	1442.60	1242.90	34	1379.30	1242.90			

5. ANALYSIS OF THE RESULTS

Table IV shows the values of the makespan resulting from the CH of phase 1, also those of the CHSA hybrid algorithm, and the values of the 5 alternative methods from the previous section.

Table IV: Values of C_{max} corresponding to each of the methods analysed in this research.

Method	$\boldsymbol{C_{max}}$ (h)	$\boldsymbol{C_{max}^{-}}$ (h)	$\boldsymbol{\bar{C}_{max}}$ (h)	$\boldsymbol{C_{max}^{+}}$ (h)
HC	1242.90			
CHSA		1238.48	1241.74	1242.9
Alternative method 1	1291.40			
Alternative method 2	1272.40			
Alternative method 3	1257.40			
Alternative method 4	1247.30			
SA		1336.85	1396.68	1464.72

Given the solution randomness obtained with the SA and CHSA methods, the minimum, average and maximum values of C_{max} are presented for both cases. The standard Simulated Annealing algorithm yields the worst result, then alternative method 1 that models a scenario corresponding to a corrective maintenance policy. As expected, alternative methods 3 and 4, corresponding both to the generation of integrated programs of production and preventive

maintenance, surpass alternative method 2 that models a scenario in which optimum production and preventive maintenance programs are obtained separately. Finally, the result obtained by our proposal demonstrates the effectiveness of the integration versus the alternative of achieving local optimum separately for each, also the validity of our modifications on the LPT dispatch rule, formulated in a deterministic context, to face the stochastic nature of our problem.

6. CONCLUSIONS

This research addressed a significant problem related to the need of establishing joint production and PM programs that help reducing the inconveniences caused by the lack of integration traditionally characterizing the management of both areas. In order to contribute solving this problem in the specific context of workshops of identical parallel machines, a new hybrid method (CHSA) was developed, integrated in a first phase by a CH based on the LPT dispatch rule, but introducing key modifications that, unlike other methods published before, at the moment of making the decision about the next job to assign, allow to consider the impact of interruptions due to failure and the effective age of the equipment on the elaboration time of the product, as well as the possibility of the jobs having different release times. In the second phase a Simulated Annealing-based algorithm was used to improve the initial solution obtained in phase 1, this algorithm was also proposed in this work. In order to prove the effectiveness of our proposal, a case study was conducted in a plant dedicated to the production of plastic products. The solution obtained was compared with those from 5 alternative methods that address this type of problem. The CHSA hybrid algorithm provided a makespan reduction of 49.66 h, 30.66 h and 154.94 h in relation to alternative methods 1, 2 and 5 respectively. Finally, the superiority of our CH with respect to alternative methods 3 and 4 showed the relevance of the modifications made to the LPT dispatch rule to obtain a method capable of adapting to the characteristics of the problem addressed. The logics behind this validation strategy can be certainly used for similar problems, e.g.: flexible job shop one, and in general, the results of the paper certainly could work as an inspiration for many production companies facing the same challenges addressed here. Future research will focus on the possibility of adopting a multicriteria approach for our proposal where, in addition to minimizing the makespan, energy saving, environmental risks reduction, etc. would be also considered. Also, the presented approach will be extended to other productive configurations such as the flexible job shop one.

ACKNOWLEDGEMENT

The work was supported by the MTF STU project: KEGA 021STU-4/2018 and the EU project H2020: 873134 – CALIPER.

REFERENCES

[1] Ruiz, R.; García-Díaz, J. C.; Maroto, C. (2007). Considering scheduling and preventive maintenance in the flowshop sequencing problem, *Computers & Operations Research*, Vol. 34, No. 11, 3314-3330, doi:10.1016/j.cor.2005.12.007

[2] Hadidi, L. A.; Al-Turki, U. M., Rahim, A. (2012). Integrated models in production planning and scheduling, maintenance and quality: a review, *International Journal Industrial and Systems Engineering*, Vol. 10, No. 1, 21-50, doi:10.1504/IJISE.2012.044042

[3] Aguado, S.; Velazquez, J.; Samper, D.; Santolaria, J. (2016). Modelling of computer-assisted machine tool volumetric verification process, *International Journal of Simulation Modelling*, Vol. 15, No. 3, 497-510, doi:10.2507/IJSIMM15(3)9.353

[4] Da, W.; Feng, H.; Pan, E. (2016). Integrated preventive maintenance and production scheduling optimization on uniform parallel machines with deterioration effect, *2016 IEEE International Conference on Industrial Engineering and Engineering Management*, 951-955, doi:10.1109/IEEM.2016.7798018
[5] Straka, M.; Lenort, R.; Khouri, S.; Feliks, J. (2018). Design of large-scale logistics systems using computer simulation hierarchic structure, *International Journal of Simulation Modelling*, Vol. 17, No. 1, 105-118, doi:10.2507/IJSIMM17(1)422
[6] Kumar, S.; Kumar Lad, B. (2017). Integrated production and maintenance planning for parallel machine system considering cost of rejection, *Journal of the Operational Research Society*, Vol. 68, No. 7, 834-846, doi:10.1057/jors.2016.46
[7] Zahedi, Z.; Salim, A. (2017). Integrating preventive maintenance scheduling as probability machine failure and batch production scheduling, *ComTech: Computer, Mathematics and Engineering Applications*, Vol. 7, No. 2, 105-112, doi:10.21512/comtech.v7i2.2247
[8] Jing, Z.; Hua, J.; Yi, Z. (2017). Multi-objective integrated optimization problem of preventive maintenance planning and flexible job-shop scheduling, *Proceedings of the 23rd International Conference on Industrial Engineering and Engineering Management*, 137-141
[9] Boudjelida, A. (2019). On the robustness of joint production and maintenance scheduling in presence of uncertainties, *Journal of Intelligent Manufacturing*, Vol. 30, No. 4, 1515-1530, doi:10.1007/s10845-017-1303-9
[10] Chansombat, S.; Pongcharoen, P.; Hicks, C. (2019). A mixed-integer linear programming model for integrated production and preventive maintenance scheduling in the capital goods industry, *International Journal of Production Research*, Vol. 57, No. 1, 61-82, doi:10.1080/00207543.2018.1459923
[11] Pinedo, M. L. (2012) *Scheduling: Theory, Algorithms, and Systems*, 4th edition, Springer, New York
[12] Anily, S.; Bramel, J.; Simchi-Levi, D. (1994). Worst-case analysis of heuristics for the bin packing problem with general cost structures, *Operations Research*, Vol. 42, No. 2, 287-298, doi:10.1287/opre.42.2.287
[13] Johnson, D. S.; Demers, A.; Ullman, J. D.; Garey, M. R.; Graham, R. L. (1974). Worst-case performance bounds for simple one-dimensional packing algorithms, *SIAM Journal on Computing*, Vol. 3, No. 4, 299-325, doi:10.1137/0203025
[14] Baker, K. R. (1974). *Introduction to Sequencing and Scheduling*, John Wiley & Sons, New York
[15] Coffman, Jr., E. G.; Garey, M. R.; Johnson, D. S. (1978). An application of bin-packing to multiprocessor scheduling, *SIAM Journal on Computing*, Vol. 7, No. 1, 1-17, doi:10.1137/0207001
[16] Xu, D.; Sun, K.; Li, H. (2008). Parallel machine scheduling with almost periodic maintenance and non-preemptive jobs to minimize makespan, *Computers & Operations Research*, Vol. 35, No. 4, 1344-1349, doi:10.1016/j.cor.2006.08.015
[17] Lee, W.-C.; Wu, C.-C. (2008). Multi-machine scheduling with deteriorating jobs and scheduled maintenance, *Applied Mathematical Modelling*, Vol. 32, No. 3, 362-373, doi:10.1016/j.apm.2006.12.008
[18] Straka, M.; Rosová, A.; Lenort, R.; Besta, P.; Šaderová, J. (2018). Principles of computer simulation design for the needs of improvement of the raw materials combined transport system, *Acta Montanistica Slovaca*, Vol. 23, No. 2, 163-174
[19] Mirabedini, S. N.; Iranmanesh, H. (2014). A scheduling model for serial jobs on parallel machines with different preventive maintenance (PM), *The International Journal of Advanced Manufacturing Technology*, Vol. 70, No. 9-12, 1579-1589, doi:10.1007/s00170-013-5348-4
[20] Wang, S.; Liu, M. (2015). Multi-objective optimization of parallel machine scheduling integrated with multi-resources preventive maintenance planning, *Journal of Manufacturing Systems*, Vol. 37, Part 1, 182-192, doi:10.1016/j.jmsy.2015.07.002
[21] Pan, E.; Liao, W.; Xi, L. (2010). Single-machine-based production scheduling model integrated preventive maintenance planning, *The International Journal of Advanced Manufacturing Technology*, Vol. 50, No. 1-4, 365-375, doi:10.1007/s00170-009-2514-9