An Integrated Artificial Bee Colony Algorithm for Scheduling Jobs and Flexible Maintenance with Learning and Deteriorating Effects

Nesrine Touafek, Fatima Benbouzid-Si Tayeb, Asma Ladj, Alaeddine Dahamni, Riyadh Baghdadi

An Integrated Artificial Bee Colony Algorithm for scheduling jobs and Flexible Maintenance with Learning and Deteriorating Effects

Nesrine Touafek[1], Fatima Benbouzid-Si Tayeb[1], Asma Ladj[2], Alaeddine Dahamni[1], and Riyadh Baghdadi[3]

[1] Laboratoire des Méthodes de Conception de Systèmes (LMCS), Ecole nationale Supérieure d'Informatique (ESI), BP 68M-16270 Oued Smar, Algiers-Algeria
[2] Railenium Research and Technology Institute, 59540 Valenciennes, France
[3] New York University Abu Dhabi

Abstract. In this paper, we address two versions of the permutation flowshop scheduling problem (PFSP) under availability constraints with learning and deteriorating effects. Availability constraints are due to flexible maintenance activities scheduled based on prognostics and health management (PHM) results. This study is motivated by an increasing agreement within academia that human behavior and machine condition effects should no longer be ignored within scheduling models and an explicit modeling is required. Hence, in the first study, human learning effect is considered and position-dependent model is applied to generate variable maintenance processing times. In the second one, besides learning effect, time-dependent machine deteriorating jobs are assumed. The objective of the two studies is the makespan minimization taking into account maintenance operations. Since the PFSP is proven to be NP-complete, improved artificial bees colony algorithms were proposed. Intense computational experiments are carried out on Taillard's well known benchmarks, to which we add both PHM and maintenance data. The results of comparison and experiments show the efficiency of our algorithms.

Keywords: Permutation flowshop scheduling problem · Learning effect · Deteriorating effect · Flexible maintenance · PHM · Artificial bee colony.

1 Introduction

The permutation flowshop scheduling problem (PFSP) is an attractive combinatorial problem in the field of scheduling, encountered in many real world applications given its practical relevance [32]. Classical PFSP problems assumed a stable and deterministic context. In such context, all machines are supposed to be continuously available throughout the scheduling horizon and job processing times are assumed to be constant and independent of human factor impact. However, in real industry settings, these hypotheses fail. Indeed, machines can be unavailable for multiple reasons, such as inspections, maintenance interventions,

sudden breakdowns, etc [22]. Moreover, job processing times may be extended due to machine deterioration effect [14] caused by the increased usage and time, or shortened as a result of the human learning effect [11] caused by frequent repetition of identical tasks. Regarding these features and to deal with a more realistic environment, recent research works are focusing on extending classical models to include practical constraints. Therefore, machines' unavailability constraints, learning as well as deteriorating effects are recently introduced in the study of scheduling problems.

While considering the learning effect, the actual processing time of a job decreases if it is scheduled later in the production sequence. However, with the deteriorating effect, the actual processing time of a job is modeled as an increasing function if it is scheduled later. Scheduling problems with the learning or the deteriorating effect considerations have received a lot of attention in recent years and many models were proposed in the literature. On the other hand, the machine availability constraint is defined as a time interval on which the machine is idle and can't process any job. Since the early 1990s, multiple studies on scheduling problems with availability constraints have been widely applied. In most cases, the availability constraint due to maintenance interventions is the most considered [18].

Given their effectiveness in maintaining machines at a high level of reliability, availability, and security, the maintenance function becomes a key feature in the manufacturing system. Therefore, maintenance strategies have evolved from traditional corrective maintenance carried out after a breakdown and systematic maintenance at fixed intervals to more advanced predictive maintenance (PM) based on prognostic health management (PHM). PHM-based predictive maintenance is a preventive strategy where maintenance decision is taken according to the health status of the machine. Indeed, The PHM module monitors the machine continuously using information collected from sensors to estimate the time left before the occurrence of failure known as the Remaining Useful Life (RUL) [9]. Then, the PHM outputs (i.e, RUL) are exploited in the decision-making that concerns the planning of predictive maintenance. The main advantage of this strategy is to maintain the system only when necessary, which reduces the cost of inopportune maintenance interventions.

In almost studies dealing with the joint optimization of production and preventive maintenance, the maintenance duration is assumed to be constant regardless of the current condition of the machine or operator. However, this assumption fails in real industry settings due to the uncertainty of the machine or the operator's condition. Indeed, the maintenance duration may be affected by the machine's deteriorating effect in that the later maintenance is planned; the worse the machine conditions are, hence, a longer time is needed to perform the maintenance. On the other hand, the learning effect of the operator may shorten the maintenance duration by repeating the same operating processes frequently. Scheduling problems with variable maintenance times are studied in the literature. While deteriorating maintenance activities are most often considered [19,

21, 33]. Variable maintenance durations due to the learning effect are scarcely studied.

To the best of the authors' knowledge, the joint scheduling problem of production and PHM-based predictive maintenance with learning and deteriorating effects was never studied in the literature. In this context, we propose to first study the PFSP with flexible maintenance and variable maintenance duration due to the learning effect, and then with deteriorating jobs and variable maintenance duration due to the learning effect simultaneously, with makespan minimization. When maintenance activities and production jobs are scheduled simultaneously, the problem becomes NP-hard in a strong sense [24]. In this paper, the formulated joint scheduling problem is even more complex. As well as allocating jobs and maintenance activities to machines, the learning and deteriorating effects on processing times are also considered. Therefore, metaheuristics seem to be the most promising resolution approach given their ability to provide near-optimal solutions with reasonable computational times.

In recent decades, the popularity of swarm intelligence algorithms has increased. Complex problems are solved using the behavior of living swarms such as birds, bees, and ants. In this regard, the artificial bee colony (ABC) is a recently introduced swarm knowledge-based algorithm to formulate NP-Hard problems [15]. It is inspired by the intelligent, self-organizing, and aggregated behavior of bee colonies when searching for promising food sources associated to higher nectar amount. Motivated by its high efficiency in solving many variants of the flowshop scheduling problems such as the PFSP [2], the hybrid flowshop [20] and the flexible flowshop [30] it was adapted to solve the studied problems. So improved and adapted ABC algorithms were developed, according to the problem constraints and objectives. The main features of the proposed ABC algorithms are: (1) To get a population with a certain level of quality and diversity, we generated an initial population of integrated candidate solutions that were generated at random and "seeded" with some good solutions. This strategy not only provides a high level of diversity in the population, but it also outperforms the approach where the initial population consisted of only randomly generated integrated solutions; (2) Five integrated local search operators to generate new neighbourhood solutions for employed bees; (3) A local search method with destruction and construction procedures to enhance the best individuals by the onlooker bees. Finally, to prove the effectiveness of our proposed algorithm, it is first compared to a variable neighborhood search (VNS) algorithm proposed by [17]. Then, intensive computational experiments are carried out on well-known Taillard's benchmarks enhanced with both PHM and maintenance data.

The rest content of the paper is structured as follows. A literature review is presented in section 2. Section 3 describes the tackled scheduling problems. Section 4 is devoted to presenting the proposed ABC algorithm. Finally, section 5 compares and analyses the performances of the newly designed algorithm. A general conclusion and perspectives of the work are drawn in section 6.

2 A brief review of scheduling problems with learning/deteriorating effects

Scheduling problems with the learning or the deteriorating effect considerations have received a lot of attention in recent years and many models were proposed in the literature which can be either time-dependent or position dependent [6]. In the former models, variable processing times of jobs depend on their processing times, while in the later, variable processing times of jobs depend on their scheduled position in the sequence. For details on this research topic, time-dependent scheduling problems are discussed in comprehensive reviews of [12] and [7], while position-dependent scheduling problems are studied in surveys of [4, 6]. More recently, [31] addressed the flowshop scheduling problem under time-dependent learning effect to minimize the makespan. To solve this problem, a Branch and Bound (B&B) algorithm and heuristics were proposed. [10] studied the flowshop scheduling problem under position-dependent learning effect. They proposed a B&B algorithm and four metaheuristics to minimize the total completion time. [3] investigated the PFSP under position-dependent learning effect and time-dependent deteriorating effet. A Population-based Tabu search algorithm is proposed for its resolution.

The integrated production and maintenance scheduling problems with learning and/or deteriorating effects have been widely studied in the literature. Recently, [1] addressed the integrated scheduling problem in the configuration of a parallel machine under position-dependent deteriorating effect to minimize makespan. To solve large-size problems, six heuristic solution methods are developed. [29] investigated the integrated single-machine scheduling problem under a time-dependent deteriorating effect. To minimize the makespan and the total completion time, polynomial algorithms were proposed. [13] considered the makespan single machine scheduling problem with a time-dependent deteriorating effect. Two batch-based heuristics and an iterated greedy algorithm were developed to solve the problem.

3 Problem statement

The PFSP is a well-known scheduling problem that can be described as follows. A set J of n independent jobs $J = \{J_1, J_2, ..., J_n\}$ has to be processed on a set M of m independent machines $M = \{M_1, M_2, ..., M_m\}$ in the same order on the m machines. We assume that all jobs are available at time zero and no preemption is allowed. In order to ensure the system reliability and to prevent the occurrence of fatal breakdowns during the processing horizon, machines are monitored continuously by a PHM module. This module is able to predict, for each machine M_i, the relative time before failure after processing a job J_j, noted RUL_{ij}. This value is used to estimate the corresponding degradation of the machine M_i caused by processing the job J_j. Let σ_{ij} be this degradation, and σ_{ij}=f(RUL$_{ij}$). We fix $\sigma_{ij} = p_{ij}/RUL_{ij}$, where $0 < \sigma_{ij} < 1$. Let Δ be the maximal authorized degradation of a machine. We fix $\Delta = 1$ for all machines.

When the accumulated degradation ($\sum_{j=1}^{l} \sigma_{ij}$) of a machine M_i after processing l jobs sequentially reaches this threshold, a predictive maintenance intervention (PM) should be planned. During the maintenance, the machine is not available for processing jobs and it will be restored to *"As good as new"* state after the end of the maintenance intervention. We assume also, that at least one predictive maintenance intervention is performed on each machine and no predictive maintenance operation is performed after the processing of the last job. In order to make the model more realistic, we further assume, in a first time, variable maintenance duration due to the learning effect, then variable job processing times due to the deterioration effect, besides.

The objective of this study is to propose an integrated solution that combines production schedule and flexible maintenance planning so that the total completion time of the schedule after maintenance operations insertion is minimized. Let S_i denotes a schedule of n jobs and k_i (≥ 1) predictive maintenance on machine M_i. Then S_i can be seen as a succession of $k+1$ blocks of jobs (B_{il}) separated by predictive maintenance operations (PM_{ix}):
S_i= {B_{i1}, PM_{i1}, B_{i2}, PM_{i2}, ..., B_{ik}, PM_{ik}, $B_{i(k+1)}$}, where $\cup_{l=1}^{k+1} B_{il} = J$.

We use the following notations to formulate the production and the maintenance data, as well as, the problem assumptions:

p_{ij}: the normal processing time of job j on machine i.
p_{ij}^{A}: the actual processing time of job j on machine i.
t_{ij}: the processing time of job j on machine i.
PM_{il}: the basic processing time of the l^{th} maintenance on machine i.
PM_{il}^{A}: the actual processing time of the l^{th} maintenance on machine i.
α: the learning index.
β: the deteriorating index.

For the first scheduling problem with variable maintenance duration due to the learning effect, the actual maintenance duration are defined by the following position-dependent learning model:

$$PM_{il}^{A} = PM_{ij}.l^{\alpha}, -1 < \alpha < 0 \tag{1}$$

For the second scheduling problem with variable maintenance duration due to the learning effect and variable job processing times due to the deterioration effect, the actual maintenance duration are defined by equation (1), while the actual job processing times are defined by the following time-dependent deteriorating model:

$$P_{ij}^{A} = P_{ij} + \beta.t_{ij}, \beta > 0 \tag{2}$$

The makespan criteria, noted C_{max}, corresponds to the completion time of the last processed job on the last machine after predictive maintenance operations insertion. Under the learning and the deteriorating effects, the calculation of C_{max} depends mainly on the sum of actual processing times of maintenance operations and those of production jobs. The C_{max} of the first and the second problem respectively are calculated using formula 3 and 4 respectively, where IT

refers to the total idle time of the last machine m, waiting jobs arrival and k is the number of inserted maintenance activities on the last machine m.

$$C_{max} = IT + \Sigma_{j=1}^{j=n} P_{mj}^{A} + \Sigma_{l=1}^{l=k} PM_{ml}^{A} = IT + \Sigma_{j=1}^{j=n} P_{mj} + \Sigma_{l=1}^{l=k} (PM_{mj}.l^{\alpha}) \quad (3)$$

$$C_{max} = IT + \Sigma_{j=1}^{j=n} P_{mj}^{A} + \Sigma_{l=1}^{l=k} PM_{ml}^{A} = IT + \Sigma_{j=1}^{j=n} (P_{mj} + \beta.t_{ij}) + \Sigma_{l=1}^{l=k} PM_{mj}.l^{\alpha} \quad (4)$$

4 Proposed solving approach

An improved and adapted artificial bee colony (ABC) algorithms are proposed to solve the two problems. In the ABC algorithm [15], candidate solutions to the problem are represented as food sources and their quality or fitness corresponds to the nectar amount. The ABC algorithm starts from previous generated solutions, and applies an iterated search process to approach the optimal solution. Each iteration of the search process consists of three main phases: employed and onlooker bees local search phases, for exploitation and scout bees global search phase, for exploration. In the employed bees phase, each food source is associated to an artificial employed bee, which generates new solutions by a neighborhood search approach. Then, the solution with the higher fitness value replaces the current one. In order to enhance the local search of optimal solution, onlooker bees apply a probability selection of the best candidate solutions and then generate new neighboring solutions using similar method as employed bees. In the scout bees phase, if a solution cannot be further improved through a limited number of iterations *limit*, then the solution is assumed to be abandoned and a new solution is generated randomly. The search process is iterated till the termination condition is met. Main steps of the proposed ABC algorithms, starting by solution representation and initial population generation, then the iterated process of employed, onlooker and scout bees phases, and finally the termination condition, are detailed in the following sub-sections.

4.1 Encoding scheme and solution representation

In the proposed ABC algorithms, a food source (solution) is represented by a two-field structure corresponding each to production and predictive maintenance. For production part, a sequence S represents the execution order of production jobs by machines. For maintenance part, a $m * (n-1)$ binary matrix PM that represents the positions of predictive maintenance on the m machines is used [5]. If an element $M[i,j]$ is set to 1 value, then a predictive maintenance is scheduled on the i^{th} machine after the j^{th} job. Otherwise $M[i,j] = 0$. For instance, let S=$\begin{pmatrix}1&9&3&8&5&6&7&4&2&0\end{pmatrix}$, and PM=$\begin{pmatrix}0&1&0&0&1&0&0&0&1\\0&0&1&0&0&1&0&0&0\\0&1&0&0&0&1&0&0&0\end{pmatrix}$, represent a solution of an

integrated flowshop scheduling problem with $m = 3$ machines and $n = 10$ jobs. Then, the execution order of the ten jobs and predictive maintenance operations on the three machines is the following:

Machine 1: j_1, j_9, PM_{11}, j_3, j_8, j_5, PM_{12}, j_6, j_7, j_4, j_2, PM_{13}, j_0
Machine 2: j_1, j_9, j_3, PM_{21}, j_8, j_5, j_6, PM_{22} j_7, j_4, j_2, j_0
Machine 3: j_1, j_9, PM_{31}, j_3, j_8, j_5, j_6, PM_{32}, j_7, j_4, j_2, j_0

4.2 Initial population generation

In order to generate an initial population of $PopSize$ complete integrated solutions (S, PM), we propose a two-step initialization procedure as follows:

- **Step 1.** Firstly, we generated a mixed initial production population which comprises: (1) one production sequence generated using the well-known NEH heuristic [23] ; (2) α%PopSize production sequences generated using the modified NEH [25]; (3) the remaining part, and the largest one, consists of $(100 - \alpha)$%PopSize randomly generated production sequences. Thus, we ensure quality and diversity of the research space.
- **Step 2.** After generating production sequences in Step 1, predictive maintenance activities are scheduled using the PHM-based greedy heuristic of [16]. In this heuristic, maintenance operations are inserted according to the current accumulate degradation of machine estimated by the PHM module, starting from the first machine M_1 to the last one M_m.

4.3 Employed bees phase

Each artificial employed bee is placed on a complete integrated solution (S, M) and performs a local search, based either on the production sequence S nor on the maintenance matrix M, for an improved one through the local neighborhood. If the fitness of the generated solution is higher, then the current one is replaced and the *nb_trails* counter is reset. Otherwise, the *nb_trails* counter of the current solution is incremented. From prior literature, we learn that two common operators, insert and swap, are commonly used to generate neighboring solutions [26]. Therefore, these neighborhoods are applied in this paper. These neighborhood structures allow to create new solutions by changing the execution order of production jobs. Besides, a third type of neighborhood structure by shifting predictive maintenance tasks, is used. In each iteration of the employed bees phase, one of these operators is chosen randomly with a uniform distribution. Based on these neighborhoods, five local search operators are described as follows:

1. **Swap move on production jobs.** it consists in swapping the positions of two production jobs selected randomly from the production sequence.
2. **Double swap move on production jobs**. It consists of making two consecutive swap moves on the production sequence.

3. **Insert move on production.** it consists in selecting a production job randomly and then inserting it into another selected position in the production sequence.
4. **Double insert move on production jobs.** It consists of making two consecutive insert moves.
5. **Right/Left shift move on maintenance activities.** it consists in choosing a maintenance intervention in a selected machine and randomly shifting it to the left, i.e. before the previous job in the sequence, or to the right, i.e. after the next one.

After generating a new solution by the employed bee, the positions of PM operations may be perturbed. Therefore, the maintenance insertion heuristic should be applied again to re-adjust the PM positions.

4.4 Onlooker bees phase

In order to enhance the intensification of promising food sources, the onlooker bees use the roulette wheel selection where solutions with higher fitness value have higher probability to be selected. Probability value for each solution is given by equation 5. Then a local search method borrowed from the iterated local search (ILS) algorithm [28] with destruction and reconstruction procedures is applied on selected solutions to explore possible promising neighbors.

$$p(sol_i) = \frac{fitness(sol_i)}{\sum_{k=1}^{n} fitness(sol_k)} \tag{5}$$

4.5 Scout bees phase

In order to escape from local optima and to allow diversification, the food sources whose *nb_trials* is greater than the limited number of iterations *limit* are abandoned and their associated employee bees become scout bees. Subsequently, they randomly generate new food sources.

4.6 Termination condition

The termination condition of the proposed ABC is a maximum number of iterations of the algorithm or when the limit of the number of iterations without improving the best solution is reached. Moreover, at the end of the ABC algorithm and in the aim to readjust possible perturbed position of PM operations, we use the maintenance insertion heuristic described in section 3.2 to regenerate the PM matrix PM of the best individual returned by the ABC algorithm.

5 Experimental testing and analyses

In this section, we describe our test plan to evaluate performances of the proposed ABC algorithm. Firstly, details on test environment and data generation

are given. Secondly, we present the ABC parameters calibration step. Thirdly, a performance comparison of the proposed ABC algorithms against the VNS algorithm of [17] is conducted, where joint scheduling problem of production and PHM-based predictive maintenance without effects of learning and deteriorating, is considered. Finally, computational results testing the performances of our algorithms to resolve the two studied problems are analysed and CPU times are discussed.

5.1 Test environment and data generation

Our computational study is performed on Microsoft Azure NC6 Promo virtual machine with six cores and 6 GB RAM. The test instances are generated using the well-known Taillard's benchmark instances [27] comprising 120 instances with different size of jobs and machines (n×m) where n $\in \{20, 50, 100, 200\}$ and m $\in \{5, 10, 20\}$, to which we add PHM and maintenance data

For PHM data, we consider three kinds of jobs according to their processing times intervals as follow:

1. Jobs with processing times < 20 inducing small machine degradation σ_{ij} generated from a uniform distribution $\mathcal{U}[0.02; 0.03]$.
2. Jobs with processing times between 20 and 50 inducing medium machine degradation σ_{ij} generated from a uniform distribution $\mathcal{U}[0.03; 0.06]$.
3. Jobs with processing times > 50 inducing big machine degradation σ_{ij} generated from a uniform distribution $\mathcal{U}[0.06; 0.1]$.

For maintenance data, two levels of maintenance durations are separately considered as follow :

1. First level with medium maintenance durations generated from a uniform distribution $\mathcal{U}[50, 100]$.
2. Second level with long maintenance durations generated from a uniform distribution $\mathcal{U}[100, 150]$.

We have then 2 possible configurations which will be run on the 120 instances. There is therefore a suite of 240 different instances to perform the tests. Each instance is tested five times, resulting in 1200 executions for the proposed ABC, per problem. We average the results for all the given instances.

The performance of the proposed ABC algorithm to solve the different problems is evaluated based on the following metrics:

1. P_RPD : the relative percentage deviation of the returned C_{max} over the best known solution C_{max}^{best} (upper bound) for each instance as described in equation 6.
$$P\_RPD = \frac{C_{max} - C_{max}^{best}}{C_{max}^{best}} \times 100. \tag{6}$$
2. CPU: computational time of the proposed algorithm.

5.2 ABC features analysis

We undertaken a sensitive analysis of performance for the proposed ABC by varying different parameters. We have chosen a full factorial design in which all possible combinations of the following factors are tested: the population size ($PopSize$), the maximum iterations number ($Max\_iterations$), the number of consecutive iterations in which an employed bee become a scout bee ($limit$) and the percentage of onlooker bees over the population size ($onlooker\_bees\%$). We set $PopSize$ to three levels: 50, 70 and 100; $Max\_iterations$ to three levels: 100, 150 and 200; $limit$ to three levels: 5, 10 and 50; finally, $onlooker\_bees\%$ to three levels: 20%, 30% and 40%. Resulting in a total of 3x3x3x3=81 possible configurations for the proposed ABC algorithm. We run the 81 possible combinations on a test-bed of problem instances generated using procedure proposed by [27] where $n \in \{20; 70; 120; 170; 220\}$ and $m \in \{5; 10; 15; 20\}$. The job processing times of the instances are generated by a discrete uniform distribution in the interval of $[1, 99]$, and maintenance durations are generated according to the first level described in previous section. Each instance is executed five times which means a total of 16 200 executions. The P_RPD is calculated as a response variable. The experimental results are analysed by means of a multi factor analysis of variance (ANOVA) technique [8]. Each parameter is set to its most adequate level as follow: $PopSize = 70$, $Max\_iterations = 200$, $limit = 5$, $onlooker\_bees\% = 40\%$.

5.3 Comparative analysis

The first set of tests is conducted to evaluate the performances of the ABC algorithm when comparing to a VNS algorithm proposed in [17]. ABC parameters are set as follow: $PopSize = 70$, $Max\_iterations = 200$, $limit = 5$, $onlooker\_bees\% = 40\%$. For a fair comparison, the VNS algorithm was re-executed on the same machine and instances as the ABC algorithm. The results of comparison based on the P_RPD and CPU values are listed in table 1, for the two levels of maintenance durations.

Although the VNS algorithm [17] has proved its efficiency against the NEH heuristic which is considered as the most powerful construction heuristic for C_{max} minimization in literature, best P_RPD values are provided by the ABC algorithm for almost the instances except instances (50×5, 50×10,50×20). The mean difference between these two algorithms is 02.29%, which can go up to 10.19% for the 20×20 benchmark for the first level of maintenance and it is of 0.93% for the second level of maintenance. This enhance the efficiency of our ABC algorithm with its embedded local search methods for solution intensification and global search approach for diversification throughout an iterated process. On the other hand, comparing the computational times (CPU) of the two algorithms, we notice that, for small instances (20×5, 20×10, 20×20, 50×5) VNS is faster than ABC. For the other instances, ABC is faster than VNS with a mean difference of 2915.51 seconds which can go up to 05.07 hours for the 200×20 benchmark. We note that we did not could run VNS tests for instances of size 500×20 because of too long execution times.

Table 1. P_RPD and CPU results for ABC and VNS.

instance	P_RPD for level 1		P_RPD for level 2		CPU time (s)	
	VNS	ABC	VNS	ABC	VNS	ABC
20x5	11.52	4.62	10.617	3.68	0.85	15.94
20x10	23.01	12.82	24.199	14.05	1.97	22.47
20x20	28.19	25.03	32.982	27.52	5.54	34.38
50x5	1.72	1.94	1.933	3.03	19.32	29.56
50x10	8.24	9.8	9.955	12.39	55.93	50.32
50x20	18.50	19.69	19.276	24.9	198.85	72.73
100x5	1.35	1.17	1.463	1.47	178.05	53.2
100x10	5.09	4.37	5.483	6.23	569.9	103.36
100x20	15.00	13.47	15.614	16.43	2375.4	166.37
200x10	3.13	1.75	3.429	2.52	10116	234.4
200x20	11.21	7.25	12.549	9.64	20808	262.61
500x20		3		3.04		1761.34
Average	11.55	9.26	12.50	11.57	3149.40	233.89

5.4 Performance analysis of the ABC with learning effects

In this section, we report experiment results evaluating the performance of the ABC algorithm to solve the integrated scheduling problem with variable maintenance durations due to the learning effect. For each instance described in section 5.1, we generate The learning indexes of the maintenance processing times randomly from a uniform distribution within three different modes, as follow:

1. Mode 1: small common learning index in [0, 0.20] reflecting low learning rate among maintenance operators.
2. Mode 2: large common learning index in [0.80, 1] reflecting high learning rate among maintenance operators.
3. Mode 3: different learning indexes per machine in [0, 1] reflecting dependency of the learning rate on machine characteristics.

Results of the execution of the ABC algorithm on the different instances with the two levels of maintenance durations and the three modes of learning indexes (LE_M1, LE_M2, LE_M3) are given in table 2.

For small size instances ($n = 20$), we notice equivalent C_{max} deviation values for the ABC algorithm in the two cases where the learning effect is considered (see table 2) or no (see table 1). While a significant decrease of P_RPD can be noticed when the learning effect is considered in case of large size instances and high learning indexes. Consequently, we can deduce that the learning effect has no significant impact on the P_RPD when few maintenance interventions are inserted (the case of small size instances) as well as in case of low learning rate among maintenance operators. On the other hand, the learning effect manifests clearly when an important number of maintenance interventions are inserted (the case of large size instances) as well as in case of high learning rate.

5.5 Performance analysis of the ABC with learning and deteriorating effects

In this section, experiment results evaluating the performance of the ABC algorithm to solve the integrated scheduling problem with both deteriorating jobs and variable maintenance durations due to the learning effect, are listed in table 2 for the two levels of maintenance (LDE columns). We used the instances described in section 5.1, to which we added learning and deteriorating indexes generated for each machine, randomly from a uniform distribution in [0, 1].

Table 2. P_RPD results for first and second problems

instance	P_RPD for level 1				P_RPD for level 2			
	LE_M1	LE_M2	LE_M3	LDE	LE_M1	LE_M2	LE_M3	LDE
20x5	4.47	4.73	4.4	13.04	3.35	3.29	3.38	12.45
20x10	12.81	13.34	13.69	23.66	14.44	14.22	14.2	23.02
20x20	24.76	25.1	25.11	37.3	27.2	26.95	27.28	36.17
50x5	1.87	0.21	0.76	11.3	2.24	-0.43	1.24	12.09
50x10	8.98	6.85	7.63	18.49	12.26	8.15	9.86	19.32
50x20	19.43	16	17.66	28.54	23.37	18.69	20.96	30.06
100x5	0.39	-2.82	-1.47	10.84	0.38	-4.3	-1.55	11.45
100x10	3.7	-0.4	1.84	13.29	4.93	-1.62	1.52	13.32
100x20	12.39	7.71	9.9	21.41	14.46	6.52	9.3	20.32
200x10	0.43	-3.84	-1	12.86	0.74	-6.36	-2.1	8.91
200x20	6.31	0.7	3.45	17.69	7.16	-1.36	3.19	15.01
500x20	1.26	-4.39	-0.52	13.04	0.23	-8.17	-2.49	12.45
Average	8.69	6.14	7.45	23.66	10.05	5.80	7.93	23.02

We notice a significant increase of C_{max} deviations values in this case where the deteriorating effect on job processing times is considered comparing to those of the first problem where no deteriorating effect is considered. Therefore, we can conclude that the deteriorating effect has a negative impact on the makespan because of the insertion of additional delays.

An important note to highlight is that, high C_{max} deviations do not reflect bad solution quality since they are calculated relatively to Taillard best known solutions with no consideration of maintenance operations and learning/deteriorating effects. Therefore, the presented P_RPD deviations (for all the experiments) are much higher than it should be if they were calculated with respect to a more precise lower bound which takes into account maintenance tasks and learning/deteriorating effects, as reported in [17].

6 Conclusion

In this paper, two makespan minimization flowshop scheduling problems under flexible maintenance activities with learning and deteriorating effects, are inves-

tigated. The maintenance activities are scheduled based on PHM post decision approach. Motivated by the complexity of the presented problems, an improved and adapted ABC algorithm is developed for their resolution. To prove the high performances of the ABC algorithm, a total of 3600 executions of the ABC algorithms is conducted on different combinations of problems data and constraints. The ABC algorithm is also compared to a VNS algorithm developed for the same problem in literature. All the experiment results show the efficiency of our proposed algorithm with its embedded local search methods for solution intensification and global search approach for diversification throughout an iterated process. As perspective of this work and given the lack of suitable test benchmarks, we propose to extend the taillard well-known benchmark with maintenance, learning and deteriorating effects data for a standard comparison study. Furthermore, it will be interesting to use the Machine learning algorithms to calibrate the ABC algorithm parameters and to propose ideal learning/deteriorating effects' indexes.

References

1. Alfares, H., Mohammed, A., Ghaleb, M.: Two-machine scheduling with aging effects and variable maintenance activities. Computers & Industrial Engineering **160**, 107586 (2021)
2. Arık, O.A.: Artificial bee colony algorithm including some components of iterated greedy algorithm for permutation flow shop scheduling problems. Neural Computing and Applications **33**(8), 3469–3486 (2021)
3. Arık, O.A.: Population-based tabu search with evolutionary strategies for permutation flow shop scheduling problems under effects of position-dependent learning and linear deterioration. Soft computing **25**(2), 1501–1518 (2021)
4. Bachman, A., Janiak, A.: Scheduling jobs with position-dependent processing times. Journal of the Operational Research Society **55**(3), 257–264 (2004)
5. Benbouzid-Sitayeb, F., Guebli, S.A., Bessadi, Y., Varnier, C., Zerhouni, N.: Joint scheduling of jobs and preventive maintenance operations in the flowshop sequencing problem: a resolution with sequential and integrated strategies. International Journal of Manufacturing Research **6**(1), 30–48 (2011)
6. Biskup, D.: A state-of-the-art review on scheduling with learning effects. European Journal of Operational Research **188**(2), 315–329 (2008)
7. Blazewicz, J., Ecker, K.H., Pesch, E., Schmidt, G., Sterna, M., Weglarz, J.: Time-dependent scheduling. In: Handbook on Scheduling, pp. 431–474. Springer (2019)
8. Borror, C., Montgomery, D.: Mixed resolution designs as alternatives to taguchi inner/outer array designs for robust design problems. Quality and reliability engineering international **16**(2), 117–127 (2000)
9. Bougacha, O., Varnier, C., Zerhouni, N.: A review of post-prognostics decision-making in prognostics and health management. International Journal of Prognostics and Health Management **11**(15), 31 (2020)
10. C, W., D, B., A, A., I, C., S, C., D, J., W, L., L, S., Ben: A branch-and-bound algorithm and four metaheuristics for minimizing total completion time for a two-stage assembly flow-shop scheduling problem with learning consideration. engineering optimization p. 52 (2020). https://doi.org/10.1080/0305215X.2019.1632303

11. D, B.: Single-machine scheduling with learning considerations. European Journal of Operational Research **115** (1999)
12. Gawiejnowicz, S.: A review of four decades of time-dependent scheduling: main results, new topics, and open problems. Journal of Scheduling **23**(1), 3–47 (2020)
13. H, X., X, L., R, R., H, Z.: Group scheduling with nonperiodical maintenance and deteriorating effects. IEEE Transactions on Systems, Man, and Cybernetics: Systems pp. 1–13 (2019). https://doi.org/10.1109/tsmc.2019.2917446
14. J, G., S, G.: Single facility scheduling with nonlinear processing times. Computers and Industrial Engineering (1988). https://doi.org/10.1016/0360-8352(88)90041-1
15. Karaboga, D., et al.: An idea based on honey bee swarm for numerical optimization. Tech. rep., Technical report-tr06, Erciyes university, engineering faculty, computer ... (2005)
16. Ladj, A., Varnier, C., Tayeb, F.S.: Ipro-ga: an integrated prognostic based ga for scheduling jobs and predictive maintenance in a single multifunctional machine. IFAC-PapersOnLine **49**(12), 1821–1826 (2016)
17. Ladj, A., Tayeb, F.B.S., Varnier, C., Dridi, A.A., Selmane, N.: A hybrid of variable neighbor search and fuzzy logic for the permutation flowshop scheduling problem with predictive maintenance. Procedia Computer Science **112**, 663–672 (2017)
18. Lee, H.T., Yang, D.L., Yang, S.J.: Multi-machine scheduling with deterioration effects and maintenance activities for minimizing the total earliness and tardiness costs. The International Journal of Advanced Manufacturing Technology **66**(1-4), 547–554 (2013)
19. Li, X.J., Wang, J.J.: Parallel machines scheduling based on the impact of deteriorating maintenance. Journal of Interdisciplinary Mathematics **21**(3), 729–741 (2018)
20. Li, Y., Li, X., Gao, L., Zhang, B., Pan, Q.K., Tasgetiren, M.F., Meng, L.: A discrete artificial bee colony algorithm for distributed hybrid flowshop scheduling problem with sequence-dependent setup times. International Journal of Production Research **59**(13), 3880–3899 (2021)
21. Lu, S., Liu, X., Pei, J., Thai, M.T., Pardalos, P.M.: A hybrid abc-ts algorithm for the unrelated parallel-batching machines scheduling problem with deteriorating jobs and maintenance activity. Applied Soft Computing **66**, 168–182 (2018)
22. Ma, Y., Chu, C., Zuo, C.: A survey of scheduling with deterministic machine availability constraints. Computers & Industrial Engineering **58**(2), 199–211 (2010)
23. Nawaz, M., Enscore Jr, E.E., Ham, I.: A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega **11**(1), 91–95 (1983)
24. Qi, X., Chen, T., Tu, F.: Scheduling the maintenance on a single machine. Journal of the operational Research Society **50**(10), 1071–1078 (1999)
25. Ruiz, R., Maroto, C., Alcaraz, J.: Two new robust genetic algorithms for the flowshop scheduling problem. Omega **34**(5), 461–476 (2006)
26. Ruiz, R., Stützle, T.: A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European journal of operational research **177**(3), 2033–2049 (2007)
27. Taillard, E.: Benchmarks for basic scheduling problems. european journal of operational research **64**(2), 278–285 (1993)
28. Tasgetiren, M.F., Pan, Q.K., Suganthan, P., Oner, A.: A discrete artificial bee colony algorithm for the no-idle permutation flowshop scheduling problem with the total tardiness criterion. Applied Mathematical Modelling **37**(10-11), 6758–6779 (2013)

29. X, S., X, G.: Single-machine scheduling with deteriorating effects and machine maintenance. International Journal of Production Research **57**, 3186–3199 (2019). https://doi.org/10.1080/00207543.2019.1566675
30. Xuan, H., Zhang, H., Li, B.: An improved discrete artificial bee colony algorithm for flexible flowshop scheduling with step deteriorating jobs and sequence-dependent setup times. Mathematical Problems in Engineering **2019** (2019)
31. Y, Z., D, W., W, L., J, C., P, Y., W, W., Y, C., C, W.: Twostage three-machine assembly scheduling problem with sum-of-processing-times-based learning effect. Soft Computing p. 24 (2020). https://doi.org/10.1007/s00500-019-04301-y
32. Yenisey, M.M., Yagmahan, B.: Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. Omega **45**, 119–135 (2014)
33. Zhang, X., Wu, W.H., Lin, W.C., Wu, C.C.: Machine scheduling problems under deteriorating effects and deteriorating rate-modifying activities. Journal of the operational research society pp. 1–10 (2017)