International Journal of Industrial Engineering & Production Research

March 2016, Volume 27, Number 1
pp.65 -82

http://IJIEPR.iust.ac.ir/

pISSN: 2008-4889

Two Meta-heuristic Algorithms for Parallel Machines Scheduling Problem with Past-Sequence-Dependent Setup Times and Effects of Deterioration and Learning

***Mir Saber Salehi Mir* & Javad Rezaeian**

Mir Saber Salehi Mir, M.Sc Student, Department of Industrial Engineering, Mazandaran University of Science and Technology, Babol, Iran,
Javad Rezaeian, Department of Industrial Engineering, Mazandaran University of Science and Technology, Babol, Iran

KEYWORDS

Scheduling,
Deteriorating jobs,
Learning effect,
setup times,
Artificial immune system,
Ant colony optimization

ABSTRACT

This paper addresses an identical parallel machines scheduling problem with past-sequence-dependent setup times, deteriorating jobs and learning effects, in which the actual processing time of a job on each machine is given as a function of the processing times of the jobs already processed and its scheduled position on the corresponding machine. In addition, the setup time of a job on each machine is proportional to the actual processing time of the already processed jobs on the corresponding machine, i.e., the setup time of a job is past-sequence-dependent (p-s-d). The objective is to determine jointly the jobs assigned to each machine and the order of jobs such that the total completion time is minimized. Since the problem is strongly NP-hard, exact solution approaches are inefficient for realistic size problems. Thus, two meta-heuristic algorithms including artificial immune system (AIS) and ant colony optimization algorithm (ACO) are proposed to solve the given problem. The performance of the proposed algorithms are evaluated by solving a set of test problems. The computational results demonstrate that the proposed algorithms are effective and appropriate approaches to find solutions as good as exact algorithms, but when the size of the problem is increased, the AIS algorithm obtains better results in comparison with ACO algorithm.

1. Introduction

Scheduling is one of the most important concerns in production planning. Scheduling means to determine which jobs can be processed by which machines in what order within a certain period of time for pre-determined purposes [1]. In deterministic scheduling problem, job processing times are assumed to be static and independent throughout the whole process. However, there are many real situations where the processing times of the jobs might be changed due to the phenomenon of deterioration and/or learning. In scheduling

*Corresponding author: Javad. Rezaeian
Email: j_rezaeian@ustmb.ac.ir
Paper first received Jan. 05, 2015, is accepted in May. 04, 2016

with deteriorating jobs, any delay in starting to process a job may lead to increases the time and effort required to accomplish the job. For instance, Kunnathur and Gupta [2] gave a practical example in the production of steel rolling where the temperature of a molten ingot, while waiting to enter a rolling machine, drops below a certain level, the ingot requires to be reheated before rolling. Comprehensive surveys of different scheduling problems with deteriorating jobs have been conducted by Alidaee and Womer [3], Cheng et al. [4] and Gawiejnowicz [5]. In addition, more recent papers which have considered scheduling problems with deteriorating jobs include Voutsinas and Pappis [6], Cheng et al. [7], Yang and Wang [8], Huang and Wang [9], Zhao and Tang [10] and etc. On the other hand, in scheduling problems with learning effects, the time needed to perform a job may decreases by the repetition of processing operation. For instance, Biskup [11] remarks that the repeated processing the similar tasks improves workers' skills, e.g., workers are able to perform setups, operate hardware and software, and handle raw materials and components at a faster pace. Biskup [11] and Cheng and Wang [12] were the first to present the concept of scheduling problems with learning effects. Moreover, in recent years, extensive surveys of research related to scheduling with learning effects was provided by Biskup [13], Yin et al. [14], Lai and Lee [15], Wang et al. [16], Yeh et al. [17] and etc.

However, scheduling problems with deteriorating jobs and learning effects simultaneously have received considerable attention recently because the phenomena exist in many real-life situations. For instance, Cheng et al. [18] gave a practical example in the manual production of glass crafts in which silicon-based raw material is first heated up in an oven until it becomes a lump of malleable dough from which the craftsman cuts pieces and shapes them according to different designs into different glass craft products. On the other hand, the pieces that are shaped later require shorter shaping times because the craftsman's productivity improves as a result of learning.

In classical scheduling problems, it is reasonable that addition to effects of deterioration and learning, the setup times are also considered. Scheduling problems involving setup times are divided into two classes: sequence-independent and sequence-dependent. In the first case, the setup time is usually added to the job processing time while in the second case, the setup time depends not only on the job currently being scheduled but also on the last scheduled job. Koulamas and Kyparisis [19] first introduced the concept of "past-sequence-dependent" (p-s-d) setup times to scheduling problems whereby the setup times are dependent on all already scheduled jobs. They considered standard single machine scheduling with p-s-d setup times and proved that the problems of minimizing maximum completion time (make span), the total completion time and the total absolute differences in completion times can be solvable in polynomial time, respectively.

In this paper, the problem of scheduling jobs on identical parallel machines with past-sequence-dependent setup times and effects of deterioration and learning is considered to minimize the total completion time of all jobs. For convenience, we denote the p-s-d setup time by S_{psd} and the total completion time by TC. The proposed problem can be represented by $Pm \mid P_{ji[r]}^{A} = P_j\left(1+\frac{\sum_{k=1}^{r-1} P_{i[k]}}{\sum_{k=1}^{n} P_{i[k]}}\right)^{a} r^{b}$, S_{psd} |TC using the three-field notation of Graham et al. [20]. Chen [21] presented scheduling $Pm \mid P_j = b_j s_j \mid$ TC, where $b_j \geq 0$ is a growth rate (deterioration rate) of job j and s_j represents its starting time, as an NP-hard problem in the strong sense even with a fixed number of machines. It is construed that the problem considered in this study is also NP hard in the strong sense. Due to the complexity of this problem, the exact methods may not be able to find optimal solutions for realistic sized problems of this type within reasonable computational times. Here, this paper proposes two meta-heuristic algorithms namely artificial immune system and ant colony optimization along with a

number of innovative features to solve the problem. The first feature is a new stochastic approach for clonal selection about our AIS algorithm and the second is the use of transition SPT rule about our ACO algorithm to solve the problem.

Applicable examples of the aforementioned research in scheduling environments can be seen in some contexts such as high technology industries. As an instance, in the manufacturing of integrated circuit (IC) boards, an IC board consists of a group of electronic components mounted together on it. These electronic components must be processed one-by-one by a machine. The processing each of these components will have an opposite effect on the ‘‘readiness’’ of all the other components which are not yet processed owing to the passing of electric current through the IC board. So, before processing each of electronic component’s, a setup operation must be conducted to return it to ‘‘full- readiness’’ status, so that the time required for this setup operation depends on the degree of ‘‘un-readiness’’ component’s and to the actual processing times of the already processed components is proportionate [19]. On the other hand, the effects of deterioration and learning arise from erosion of machine and equipment, and increase workers proficiency by repeating the tasks, respectively

The remaining of this paper is organized as follows. Section 2 provides a brief review on the related literature and summarizes some proposed papers applied in this case. In Section 3, the problem definition and mathematical model are presented. In sections 4 and 5, the proposed algorithms are described in detail. In Section 6, computational experiments are performed and the results are presented. The last section presents the conclusions.

2. Literature Review

This section presents a brief review of the relevant literature into two categories: scheduling problems studies and optimization algorithms studies. The study of scheduling research with deteriorating jobs and learning effects has been growing recently. To the best of our knowledge, there are only a few researches of the coexistence of the effects of deterioration and learning. Lee [22] was the first author to discuss both the effects. He showed that the single-machine problems to minimize the makespan and the total completion time are polynomially solvable under simple linear deterioration. Sun [23] introduced a new scheduling model in which the actual processing time of a job is a specific function of the processing time of the jobs already processed and its scheduled position. Also, he showed that the single-machine problems remain polynomially solvable when the performance criterion is make span, total completion time and the sum of the quadratic job completion times. Lee and Lai [24] provided a general scheduling model where the actual job processing time is a general function on the processing times of the jobs already processed and its scheduled position. They proved that the single-machine problems to minimize the make span, the total completion time and the sum of the power of the completion time are polynomially solvable with complexity of o$(n\,logn)$. They also showed that the total weighted completion time, the maximum lateness, the maximum tardiness and the total tardiness problems are polynomially solvable under a certain condition. Huang et al. [25] presented a model with deteriorating jobs and learning effects where the job processing times are defined by functions of their starting times and positions in the sequence. They proved that the parallel identical machines scheduling problems when the objectives are minimizing a cost function containing total completion time and total absolute differences in completion times, and minimizing a cost function containing total waiting time and total absolute differences in waiting times remain polynomially solvable. Wang and Wang [26] investigated unrelated parallel machines scheduling problems in which the actual processing time of a job is a function of joint time-dependent deterioration and position-dependent learning. They showed that the scheduling problems of minimizing a cost

function containing total completion (waiting) time, total absolute differences in completion (waiting) times and total machine load could be solved in polynomial time, if the number of machines is a given constant. Ji et al. [27] considered parallel-machine scheduling problems with deteriorating jobs and DeJong's learning effect to minimize the total completion time and the makespan. Their results showed that the problem is polynomially solvable if the objective is total completion time, while the problem is NP-hard if the objective is makespan. Salehi mir and Rezaeian [28] addressed unrelated parallel machines scheduling problem with past-sequence-dependent setup times, release dates, deteriorating jobs and learning effects. They presented a new scheduling model in which the actual processing time of a job on each machine is given as a function of its starting time, release time and position on the corresponding machine. They also provided an efficient hybrid approach to solve the given problem with the objective of minimizing total machine load.

The above researches on scheduling with deteriorating jobs and learning effects neglected the setup times. Among various types of scheduling problems involving setup times, past-sequence-dependent setup time is considered in this article. As mentioned in the previous section, Koulamas and Kyparisis [19] were the first authors to discuss past-sequence-dependent setup times. Immediately after the researches of Koulamas and Kyparisis [19], Kuo and Yang [29] studied single-machine scheduling problem with past-sequence-dependent setup times and learning effects. They showed that the problems of minimizing makespan, the total completion time minimization, the total absolute differences in completion times and the sum of earliness, tardiness, and common due-date penalty could be solved in polynomial time. Zhao and Tang [30] investigated single machine scheduling problems with past-sequence-dependent setup times and deteriorating jobs. They proposed polynomial time algorithms to solve problems of minimizing makespan, total completion time, and sum of earliness, tardiness, and due-window starting time and size penalties. Hsu et al. [31] provided a polynomial time solution for the unrelated parallel machine scheduling problem with setup time and learning effects. Wang and Wang [32] considered single-machine setup times scheduling problem with general effects of deterioration and learning. They claimed that the problems to minimize the makespan, the sum of the δth power of job completion times, the total lateness are polynomially solvable. They also showed that the total weighted completion time, the discounted total weighted completion time, the maximum lateness (tardiness), and the total tardiness minimization problems could be solved in polynomial time under certain conditions. Liu [33] addressed parallel machine scheduling problems with delivery times and learning effect and developed polynomial algorithms to optimally solve the minimization problems of the total absolute deviation of job completion times, the total load on all machines and the total completion time.

In the last two decades, new nature-inspired meta-heuristic approaches such as genetic algorithm (GA), ant colony optimization algorithm (ACO), simulated annealing algorithm (SA), particle swarm optimization algorithm (PSO), artificial immune system algorithm (AIS), cuckoo search algorithm (CSA) and … have been widely used to further improve the solution of optimization problems with complex nature [28, 34-38].

Among the modern metaheuristic-based algorithms, a significant interest has been arisen in applying ACO and AIS algorithms to solve the complex combinatorial optimization problems. ACO is a promising metaheuristic algorithm that a pioneering work relevant to it was done by Dorigo et al. [39]. In the literature, there are only few reviews dealing with ACO algorithm in the scheduling problems. Gajpal et al. [40] developed a new ant colony algorithm based on local search procedure to solve the flowshop scheduling problem. Rossi and Dini [41] proposed an ant colony optimization to the scheduling of flexible manufacturing systems (FMSs) in a job-shop

environment. The presented algorithm by them consists of two innovative features: the first feature is the disjunctive graph model and a LS algorithm, and the second is the pheromone trail structure. Liao and Juan [42] used an ACO algorithm with two distinctive features, including a new parameter for the initial pheromone trail and adjusting the timing of applying local search for minimizing the weighted tardiness on a single machine. Arnaout et al. [43] addressed the non-preemptive unrelated parallel machine scheduling problem with sequence-dependent setup times. They introduced an ACO algorithm with a local search procedure, in which the innovative feature is the use of two successive pheromone trails to solve the problem. Mirabi [44] designed an ACO algorithm with a local search procedure for the sequence-dependent permutation flowshop scheduling problem. He benefited from a new approach for computing the initial pheromone values in the proposed algorithm. Zhang and Liu [45] hybridized an ACO algorithm with a maximum and minimum ant colony mechanism and elite strategy to solve job shop scheduling problem with unrelated parallel machines. Sun [46] applied an ACO algorithm for the general two machine flow shop problem and showed that the algorithm is capable of providing good quality solutions in comparison with existing GA. Rossi [47] implemented an ant colony system (ASC) based on disjunctive graph where a reinforced relation-learning model of the pheromone is modified in order to solve the flexible job shop scheduling problem. Arnaout et al. [48] in continuation of previous work (Arnaout et al. [43]) used an enhanced two-stage ACO algorithm (ACO II) for minimizing the makespan on the unrelated parallel machine scheduling problem with sequence-dependent setup times. Their results showed the superiority of the enhanced ACO II than other algorithms. On the other hand, the AIS algorithm is another of the newest nature-inspired algorithms which searches the solution space with a population of antibodies for recognizing invader factors (pathogen) and eliminating them. As examples of AIS algorithm applications in different scheduling problems, Naderi et al. [49] proposed a hybrid artificial immune algorithm (AIA) based on the simulated annealing for realistic variant of job shops under the minimization of the total completion time. Al-Anzi and Allahverdi [50] implemented an AIS algorithm for solving two-stage multi-machine assembly scheduling problem with the objective of minimizing total completion time. Lin and Ying [51] applied a revised AIS algorithm for minimizing makespan in a blocking flowshop scheduling problem. Abdollahpour and Rezaian [52] hybridized an AIS algorithm with an iterated greedy algorithm to solve permutation flow shop scheduling problem with the limited buffers between consecutive machines.

3. Mathematical Model

3-1. Problem description

There are n jobs ($J = \{J_1, J_2, \dots J_n\}$) that are immediately available for processing on m identical parallel machines $\{M_1, M_2, \dots, M_m\}$. At any time, each machine can process at most one job. All jobs are available at the beginning of the scheduling and the preemption is not allowed. Let $n_i$denotes the number of jobs assigned to M_i and $\sum_{i=1}^{m} n_i = n$. Let P_j be the normal (basic) processing time of job J_j in a sequence. In addition, let $P_{i[k]}$ and $P^A_{i[k]}$ be the normal processing time and actual processing time of a job if it is scheduled in the *k*th position on machine M_i in a sequence, respectively. Due to the learning and the deterioration effects, the actual processing time of job $J_j$which is scheduled on machine M_i in position *r* is given by:

$$P^A_{ji[r]} = P_j \left(1 + \frac{\sum_{k=1}^{r-1} P_{i[k]}}{\sum_{k=1}^{n} P_{i[k]}}\right)^a r^b \quad i = 1,2,\dots,m\,, j = 1,2,\dots,n\,, r = 1,\dots,n_i \qquad (1)$$

where $a \geq 1$ is the common deterioration rate for all jobs, and $b < 0$ is the learning index. In this model, the actual processing time of job $J_j$on machine M_i might be prolonged due to the deterioration effect which depends on the processing times of the jobs already processedon machine M_i. On the other hand, the actual processing time of job J_j on machine M_i might be shortened due to the learning effect which depends on its scheduled positionon machine M_i.

We also take setup times into consideration in the scheduling model by adopting the notion of Koulamas and Kyparisis [19] and Kuo and Yang [29], in which setup times are past-sequence-dependent (p-s-d), i.e., the p-s-d setup time of job $J_{[r]}$ if it is scheduled on machine M_i in position *r* is given by:

$$S_{i[1]} = 0 \quad and \quad S_{i[r]} = \gamma \sum_{k=1}^{r-1} {P_{i[k]}}^A \; i = 1, 2, \ldots, m \;, r \; = 2, \ldots, n_i \tag{2}$$

where $\gamma \geq 0$ is a normalizing constant. It is clear from the above formulation that the value of normalizing constant γ determines the actual lengths of the required setups and p-s-d setup time of the scheduled job in the first position of each machine is equal to zero.

3-2. Notations

To develop the mathematical model of the parallel machines problem with past-sequence-dependent setup times and effects of deterioration and learning, the notation adopted in this paper includes as follows:

- *Indices and parameters*

i	index of machine, $i = 1, 2, \ldots, m$
j	index of job, $j = 1,2, \ldots, n$
r,k	index of position, $r, k = 1, \ldots, n_i$
$J = \{ J_1, J_2, \ldots J_n \}$	set of all jobs; where n is the total number of jobs
$M = \{M_1, M_2, \ldots, M_m\}$	set of all machines; where m is the total number of machines
$P_j (= P_{ji})$	normal processing time for job J_j
$P_{i[k]}$	normal processing time of the job which is scheduled in the kth position on machine i
n_i	number of jobs assigned to machine i
$UB = n - m + 1$	Maximum number of positions on each machine that jobs are placed on them.
a	common deteriorating rate
b	learning index
γ	a normalizing constant

- *Decision variables*

X_{jir}	1 if job $j$is assigned to machine $i$in position $r$and 0 otherwise

3-3. Model formulation

The objective function and constraints for assigning jobs to each machine and sequencing jobs on the corresponding machine to minimize the total completion time will be formulated as follows:

$$\text{Minimize} \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{r=1}^{UB} (n_i - r + 1)(1 + \gamma\,(n_i - r)/2) P_j \left(1 + \frac{\sum_{k=1}^{r-1} P_{i[k]}}{\sum_{k=1}^{n} P_{i[k]}}\right)^a r^b \, X_{jir} \tag{3}$$

Subject to :

$$\sum_{i=1}^{m}\sum_{r=1}^{UB} X_{jir} = 1 \qquad j = 1,2,\dots,n \tag{4}$$

$$\sum_{j=1}^{n} X_{jir} \le 1 \qquad i = 1,2,\dots,m \ , r = 1,\dots,UB \tag{5}$$

$$\sum_{j=1}^{n} X_{jir} - \sum_{l=1}^{n} X_{l,i,r-1} \le 0 \qquad i = 1,2,\dots,m \ , r = 2,\dots,UB \tag{6}$$

$$\sum_{j=1}^{n}\sum_{r=1}^{UB} X_{jir} = n_i \qquad i = 1,2,\dots,m \tag{7}$$

$$\sum_{j=1}^{n} P_j X_{jik} = P_{i[k]} \qquad i = 1,2,\dots,m \ , k = 1,\dots,UB \tag{8}$$

$$X_{jir} = 0 \ or \ 1 \qquad i = 1,2,\dots,m \ , j = 1,2,\dots,n \ , r = 1,\dots,UB \tag{9}$$

Eq. (3) minimizes the total completion time. Constraint (4) ensures that each job is assigned to one of the existing positions on the machines. Constraint (5) ensures that in each existing positions on each machine, at most one job can be assigned. Constraint (6) ensures that until one position on a machine is empty, jobs are not assigned to subsequent positions and jobs assigned to empty positions on each machines, respectively. Constraint (7) ensures that the number of jobs assigned to each machine is equal to the number of jobs that be assigned to the existing positions on corresponding machine.Constraint (8) measures processing time of a job for each position on each machine. Constraint (9) defines the range of variable.

4. Artificial Immune System

The artificial immune system (AIS) was introduced by Farmer et al. [53] as a new computational intelligence technique. The artificial Immune System (AIS) is a stochastic search algorithm based on the principles of biological immune system. AIS is inspired on biological immune system (BIS), so that performs an evolutionary search by imitating the biological immune system. A BIS has a special mechanism to defend the body against exogenous infectious microorganisms such as viruses, bacteria and other parasites (pathogens). Pathogens are associated with specific proteins that are called antigens. The immune system is composed of variety of molecules called antibodies, cells and organs spread in body that are capable to recognize antigens and killing pathogens.

The AIS, similar to genetic algorithm (GA), generally starts with an initial set of antibodies population that each of them shows a point of solution space. Many of the principles applied in immune algorithms are inspired by clonal selection theory and affinity maturation. An affinity is assigned to each antibody with regard to its solution quality, such that the solution of an antibody with the better objective value has the higher affinity value. Clonal selection is the process in order to describe the capabilities of recognizing and eliminating of an immune system in the face of antigenic stimulus, in which some of the particular antibodies with higher affinity values are selected and proliferated. During the process of clonal proliferation, the affinity between antibodies and antigens is improved by the mutation of antibodies called as hyper-mutation mechanism. This mechanism plays a critical role in creating diverse antibodies, increasing affinity and enhancing specificity of the antibodies (Shahrizal Muhamad and Deris, [54]).

The pseudo-code of the AIS algorithm is illustrated in Figure 1, and how to implement and details the proposed method that has been utilized to solve the parallel machines scheduling problem are described as follows.

4-1. The Representation of Antibody and Initial Solutions

The candidate solution (i.e. antibody) is encoded as a string of integers representing of a permutation list of n jobs. In this study, we divide the antibody into two segments. The first segment of the antibody is made up of n job parts, denoted by integers, numbered from 1 to n. The second segment is made up of $m-1$ partitioning parts, denoted by "*", with distinct subscripts from 1 to $m-1$. Then generally, for a problem with n jobs to be assigned to m machines, a antibody consists of $n+m-1$ distinct parts. An example with 8 jobs and 3 machines is demonstrated in Figure 2, where the antibody would assign jobs 2, 5, 1 to machine 1, jobs 6, 3, 8 to machine 2, and jobs 4, 7 to machine 3. Also in initialization phase, the population includes *pop-size* antibodies and each antibody is generated in a random procedure from the solution space.

Algorithm. Artificial immune system meta-heuristic

1. Initialization:
   1.1. Initialize the parameters
      1.1.1. Population size $(\mathbf{N_{pop}})$
      1.1.2. Maximum number of iteration ($\mathbf{MaxIt}$)
      1.1.3. Number of selection $(\mathbf{N_c})$
   1.2. Randomly generate an initial population of antibodies
   1.3. Set I = 1.
2. Evaluate each antibody of initial population by calculating its corresponding affinity function value (Eq. 10)
3. While termination condition is not met do the following:
   3.1. Select the best $\mathbf{N_c}$ antibodies with the highest affinity function values
   3.2. Clone antibodies selected in step 3.1 using the roulette wheel and tournament procedures
   3.3. Mutate the cloning antibodies by destruction and construction phases of iterated greedy
   3.4. Evaluate each mutant antibody by calculating and sorting its corresponding affinity function value
   3.5. Update the antibody population in step 3.4 by selecting the best $\mathbf{N_c}$ antibodies
   3.6. Perform the Receptor editing
4. Set I=I+1. If I<$\mathbf{I_{max}}$(maximum number of iterations) got to step 2, otherwise stop.

Fig. 1. The AIS pseudo-code

4-2. Affinity Calculation

Each schedule (antibody) has an affinity value which is evaluated based on an affinity function. The aim of affinity evaluation is to calculate the competence of each antibody in the population based on criterion that defines the objective function. Since the objective function of the formulated model is minimization, we define the following function to calculate each antibody's affinity:

$$Affinity\,(k) = exp\left(-\beta\left(\sum_{j=1}^{n} c_j(k) \Big/ worst \sum_{j=1}^{n} c_j\right)\right) \quad (10)$$

where β denotes the selection pressure; and $\sum_{j=1}^{n} c_j(k)$ denotes the total completion time of antibody k. From this equation, it can be observed that the lower objective value of a solution equals to higher affinity value of its corresponding antibody. It means that an antibody with higher affinity function value has a higher probability of the cloning to transfer into the mutating pool.

2	5	1	$*_1$	6	3	8	$*_3$	4	7

Fig. 2. A typical solution representation

4-3. Selection and Clone

During the search procedures of the proposed AIS, each antibody in the population will be cloned proportionally to its affinity to compose a new set of fittest antibodies. According to the cloning process, the high-affinity antibodies will produce more clones compared to weaker ones. In this study, a new stochastic strategy is presented for generating the clone population via two types of selection procedure. The purpose of using this strategy is to add diversity to the algorithm and to raise the average of affinity value in the next antibody population. The first procedure is the roulette wheel selection, in which the antibodies in each generation are sorted based on their corresponding affinity values. The second one is tournament selection procedure, in which a number of antibodies (tour-size) are selected from the population randomly and antibody with more favorable affinity value will receive a higher probability to participate in next generation. According to the standard cloning and experimental results, 10 individuals of a population proliferates 55 numbers of clones.

4-4. Mutation Mechanism

Each clone in clone population will be mutated to generate the matured population by exploring the neighborhood solutions. In this study, an approach with destruction and construction phases of iterated greedy is applied as hyper-mutation mechanism to improve the quality of the solutions. In the destruction phase, h unrepeatable components are randomly selected and removed from a complete sequence of antibody in the same order that they are chosen. During the implementation process, a partial solution with $(n+m-1-h)$ elements will be obtained. Meanwhile in the construction phase, h unequal positions are randomly selected in the obtained partial sequence and then the removed components are re-inserted into the selected positions.

Due to the affinity value, each clone receive different probability of change. The probability of applying the mutation operator is calculated according to Eq. (11).

$$\vartheta_k = exp\left(-Affinity\ (k)\right) \qquad (11)$$

where ϑ_k represents the probability of applying a mutation mechanism on antibody k. It is evident from this equation that the antibody with smaller affinity value is mutated at a higher probability. An illustration is provided in Figure 3 to help achieve a better understanding of the proposed mutation method.

4-5. Receptor Editing

Receptor editing is the process which generates a new population by combining the best antibodies of current population and the best antibodies of matured population. In this mechanism, after cloning and mutation processes, the algorithm eliminates a number of worst antibodies from antibody population and then adds the same number of the best mutated antibodies into the population.

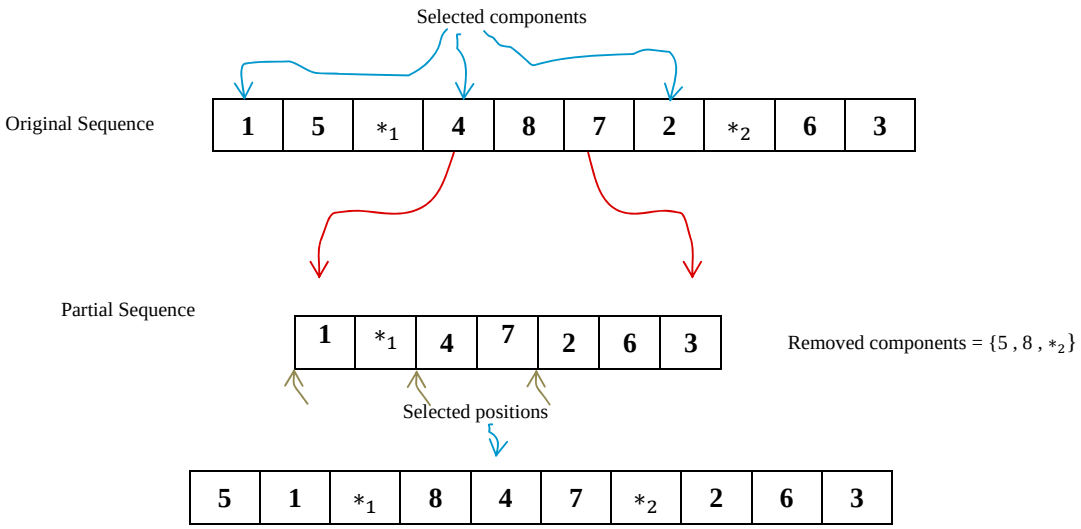


Fig. 3. An illustration of the proposed mutation method

5. Ant Colony Optimization (ACO)

Ant colony optimization algorithms have been proposed by Dorigo et al. [39]. The ACO algorithm is a metaheuristic that inspired by the real ants' behavior to find the optimal path between a food source and their nest. When ants move randomly in different directions to find a food source, they deposit a chemical substance called pheromone. Other ants, which leave the nest later, are able to smell this pheromones, i.e., they instinctively follow each other by sensing that pheromone. Hence, after a certain period of time, they converge to the optimal route which has the greatest concentration of the pheromone when compared to other routes.

Once the ants have been constructed their solutions, they evaluate the solutions obtained by the update pheromones procedure to decide how much pheromone to deposit. The pheromone update includes pheromone deposit and pheromone trail evaporation. Pheromone deposit increases the level of the pheromone of solution components that are associated with a chosen set of fine solutions. Pheromone trail evaporation decreases over time the pheromone deposited by previous ants. In other words, pheromone deposit causes convergence of the algorithm toward global optimal solution and pheromone evaporation causes avoid immature convergence of the algorithm toward a local optimal solution (Dorigo and Stützle, [55]).

The pseudo-code of the ACO algorithm is presented in Figure 4, and then the detailed description is as follows.

Algorithm. Ant colony optimization meta-heuristic

1. Initialization:
    - 1.1. Initialize the parameters
        - 1.1.1. Number of ants ($\boldsymbol{N_{ant}}$)
        - 1.1.2. Maximum number of iteration ($\boldsymbol{MaxIt}$)
        - 1.1.3. Initial pheromone values($\boldsymbol{\tau_0}$)
        - 1.1.4. Evaporation rate ($\boldsymbol{\rho}$)
        - 1.1.5. Possible selection rate ($\boldsymbol{q_0}$)
        - 1.1.6. Number of neighborhoods ($\boldsymbol{nn}$)
    - 1.2. Randomly generate an initial population of ants
    - 1.3. Set $I = 1$.
2. Evaluate each ant of initial population
3. While termination condition is not met do the following:
    - 3.1. Construct solution:
        - 3.1.1. Create a feasible permutation of jobs by using the SPT rule
        - 3.1.2. Assign the jobs into machines using the specified of possible rules (Eqs. 12,13)
    - 3.2. Improve the solution by local search (optional)
    - 3.3. Update the pheromone trail or trail intensities
    - 3.4. Evaluate and compare the fitness constructed solutions with together
4. Return the best solution found
5. Set $I=I+1$. If $I<\boldsymbol{I_{max}}$(maximum number of iterations) got to step 2, otherwise stop.

Fig. 4. The ACO pseudo-code

5-1. Construction graph

To solve the optimization problems using ACO algorithm, a graph can be used to indicate the desired problem.The graph that is employed in ACO is based on the graph used by Keskinturk et al. [56]. In this graph, jobs are considered as supernodes. Each supernode consists of '' *m* '' nodes, which represents themachines on which each job can be processed. In the proposed approach, at each iteration of the algorithm, first a

population of ants is generated and a certain amount of pheromone (τ_0) on each node $(\text{i.e., existing nodes on supernodes})$ to be deposit. In order to assign jobs to machines and construct a solution (tour), initially a feasible order of jobs is generated by using of SPT rule (shortest processing time) and then, each ant chooses the next node, which has not been visited yet, according to the following mechanism:

1. Let q is the random number generated from the uniform distribution $U\sim[0,1]$ and let q_0 is the possible selection rate, that $0 \le q_0 \le 1$.
2. If $\le q_0$, an ant selects the node that has the maximum pheromone amount, i.e.

$$S = \arg\max\{\tau_{(j_i, j'_{i'})}(t)\} \quad , j'_{i'} \in N^k \qquad if q \le q_0 \qquad (12)$$

3. If $> q_0$, an ant selects the node with respect to the following discrete distribution probability function:

$$P^k_{(j_i, j'_{i'})}(t) = \begin{cases} \dfrac{\tau_{(j_i, j'_{i'})}(t)}{\sum_{j'_l \in N^k} \tau_{(j_i, j'_l)}(t)} & j'_{i'} \in N^k \\ 0 & otherwise \end{cases} \qquad (13)$$

Where
j_i denotes node i of supernode j, and $P^k_{(j_i, j'_{i'})}(t)$ is the transition probability from node j_i to $j'_{i'}$ for ant k at iteration t, and $\tau_{(j_i, j'_{i'})}(t)$ is the pheromone intensity between nodes j_i and $j'_{i'}$ at iteration t, and S is the arc that connects two nodes j_i and $j'_{i'}$ to each other, N^k is the list of nodes not yet visited by the kth ant. This procedure repeats until all the jobs are scheduled. For example, for a problem with 7 jobs and 3 machines; the route that an ant builds in order to assignment the jobs to the machines, can be shown in Figure 5. The route of ant 1,
is $[0, 2_1, 4_1, 5_2, 7_3, 6_2, 3_3, 1_2, 0]$,where (j_i) is $(\text{job}_{\text{machine}})$. This route indicates that the first machine will process jobs 2 and 4; the second machine will process jobs 5, 6 and 1; and the third machine will process jobs 7 and 3; respectively.

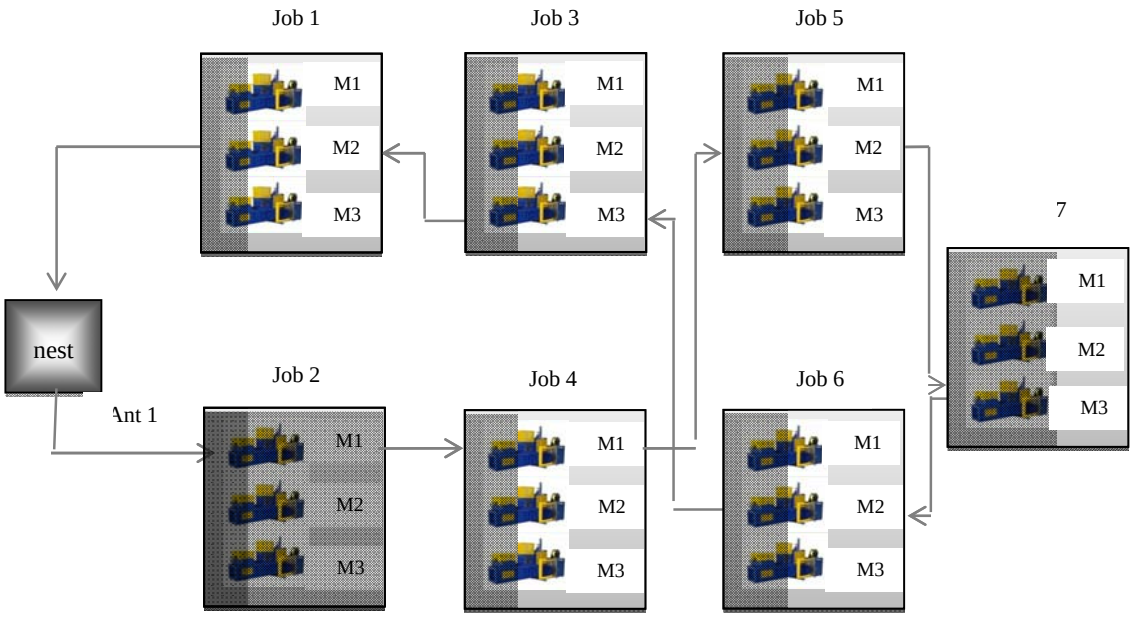
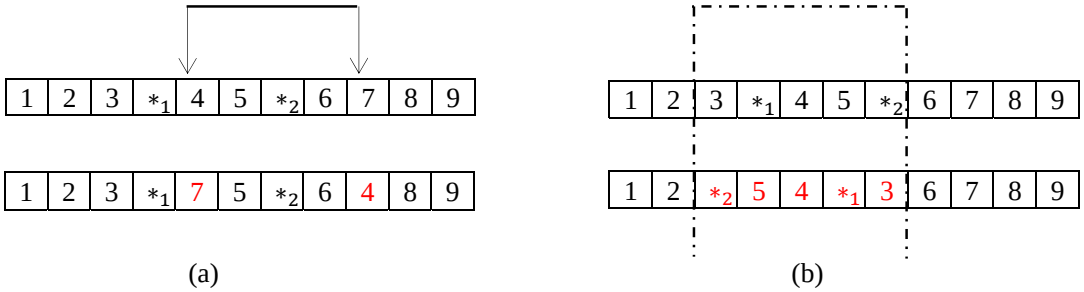


Fig. 5. Tour for Ant 1 in ACO

5-2. Local search

In each iteration, in order to improve the generated solutions, a local search procedure can be utilized by exploring the neighborhood of a given initial solution. One of the most important components when designing a local search is choice of the neighborhood structure. In this study, we use a local search with two operators: swap and inversion, and because the local search is a time-consuming procedure, we will apply local search to the best ant found in this iteration. Figure 6 shows the schemes of these two operators. The detail description of the local search algorithm applied is as follows:

1. Set $t = 1$.
2. *Swap*: randomly choose two elements from ith ant, ant i is a sequence of elements containing jobs and machines , and their positions are exchanged. If a better solution is found, it replaces the current solution; otherwise, go to step 3.
3. *Inversion*: randomly choose two cut points from ith ant and the positions of the elements placed between these points will be reversed. If a better solution is found, it replaces the current solution; otherwise, go to step 4.
4. Set $t = t + 1$. If $t < MaxIt$ (maximum number of iterations), go to step 2; otherwise stop.



(a) (b)

Fig. 6. Example of Local search procedures: (a) swap; (b) inversion

5-3. Updating the pheromone trails

Updating pheromones is the process through which the pheromone trails are modified at the end of each iteration. After an iteration is completed, i.e., after all ants have constructed their tour, the amount of pheromone trail on each edge is updated according to the quality of the constructed tour solutions (objective function value). The update rule consists of two segments: the first, is the evaporation of the existing pheromone and the second, is the quantity of added pheromone on each edge. Hence, the total amount of pheromone trails on each edge is calculated as follow:

$$\tau_{ij}(t+1) = \tau_{ij}(t) - \rho.\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \qquad (14)$$

where $\tau_{ij}(t+1)$ is the pheromone intensity between nodes i and j at iteration $t+1$, ρ is the evaporation rate, m is the number of ants, and $\Delta\tau_{ij}^{k}(t)$ is the additive pheromone amount on edge(i,j) by the kth ant:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L_K} & \text{if arc } (i,j) \text{ is visited by ant } k \\ 0 & otherwise \end{cases} \qquad (15)$$

where Q is a constant and L_K is the length of the tour done by ant k.

6. Computational Experiments

In this section computational experiments are conducted to evaluate the performance of the algorithms that were applied to solve the presented model. The proposed algorithms,

namely artificial immune system and ant colony optimization, were coded in MATLAB R2009b and executed on a personal computer with 2 GHz CPU and 768 MB RAM on Windows XP. Also, as a comparison, optimal solution approach (i.e., B&B) under the LINGO 9.0 software is employed to solve the presented model with randomly generated instances which run on the same computer. Below the details of the doing experiments and also their results are described.

6-1. Data generation

The proposed model is applied for 15 test problems, comprising 10 small- and 5 large-size problems. The data required to solve the problem include the number of machines, number of jobs, normal processing times, normalizing constant, common deterioration rate and learning index. The number of machines ranges from 2 to 14, and the number of jobs ranges from 6 to 40.The normal processing times are generated from a discrete uniform distribution between (1,15). Three parameters normalizing constant, deterioration rate and learning rate are fixed values that can be changed due to the solution space of problem, but the values of these three parameters which are considered to solve problems are 0.5, 1.05 and -0.3, respectively.

6-2. Experimental results

In this section, an experimental design to compare the results of meta-heuristic algorithms is provided using the set of 15 different instances. All instances are solved by the AIS and ACO algorithms. For each configuration, two meta-heuristic algorithms are run in 10 replications and the best, the average solutions acquired and CPU time related to best solution are recorded. In small-scale instances, in order to evaluate the efficiency of proposed algorithms in achieving optimal solutions, the problems are solved by three approaches: the exact optimal solution with branch-and-bound approach using LINGO 9.0 software and the proposed meta-heuristic algorithms. Due to the complexity of problem studied, LINGO software could not find an optimal solution for some of the small size problems in a reasonable CPU time. Thus, the best solution obtained after 3 h is reported for the small size problems that cannot be optimally solved by LINGO software. Computational results obtained from algorithms in small size problems are listed in Table 1. As can be seen from Table 1, the proposed AIS and ACO algorithms are able to achieve optimal/near optimal solutions within a considerably shorter CPU time than LINGO software.

Tab. 1. The computational results of the small-scale experiments

No.	n	m	LINGO (B&B)		AIS			ACO			RPD (%)	
			O.V.	Time(s)	Avg	Best	Time(s)	Avg	Best	Time(s)		
			AIS	ACO								
1	6	3	76.109	5	76.109	76.156	8.31	76.109	76.109	5.92	0	0
2	6	2	92.822	3	97.217	97.326	5.75	97.217	98.016	4.48	4.734	4.734
3	7	3	94.073	460	94.073	94.073	11.38	94.073	94.305	9.48	0	0
4	8	4	110.631	4500	110.631	110.631	33.42	110.631	110.631	32.96	0	0
5	9	3	134.972	877	143.867	144.891	44.89	143.867	143.968	37.22	6.590	6.590
6	10	5	116.742	8100	116.742	116.766	73.29	116.742	116.796	68.52	0	0
7	10	4	113.066	6515	115.558	117.045	78.55	115.558	116.329	62.41	2.204	2.204
8	11	6	99.663	8532	99.663	99.718	85.90	99.663	99.982	77.69	0	0
9	13	6	194.343	10800	196.254	196.867	104.37	196.254	197.059	92.01	0.983	0.983

10	15	7	166.581	10800	172.481	173.148	133.35	172.481	174.334	117.11	3.541	3.541
Average											1.8052	1.8052

n : number of jobs; m : number of machines; O.V. : objective value; Avg : average of objective values in 10 replications..

In large-scale instances, the performance of the proposed algorithms in solving large problems are evaluated by comparison with each other in terms of solution quality and CPU time. Table 2 and Figure 7 show the results obtained from algorithms in large size problems. As is clear, the AIS algorithm acquires better results than ACO algorithm in terms of solution quality and on the other hand, the ACO spends less CPU times than AIS in achieving a near-optimal or optimal solution.

Tab. 2. The computational results of the large-scale experiments

No.	n	m	AIS			ACO			RPD (%)
			Best	Avg	Time(s)	Best	Avg	Time(s)	
1	17	8	171.220	177.582	116.27	174.949	181.262	94.39	2.177
2	20	10	193.328	201.957	138.56	199.702	206.715	124.39	3.296
3	25	10	231.631	267.298	156.33	243.118	261.523	137.23	4.959
4	30	13	367.109	380.833	190.08	381.261	401.396	155.49	3.854
5	40	14	455.537	484.135	236.41	477.367	513.367	171.24	4.792
Average									3.8156

n : number of jobs; m : number of machines; Avg : average of objective values in 10 replications.

Furthermore, for each problem set, the average relative percentage deviation (RPD) in terms of the TC are determined by averaging solution RPDs measured. RPD is computed for each algorithm in each instance by the following formula:

$$RPD = \frac{Alg_{sol} - Best_{sol}}{Best_{sol}} \times 100 \qquad (16)$$

where Alg_{sol} is the objective value obtained for a given algorithm and $Best_{sol}$ is the best solution obtained for each instance. As revealed in Tables 1, the average of the RPDs between the B&B method and AIS and ACO in terms of the TC are obtained 1.8052 and 1.8052, respectively. In addition, the average of RPDs obtained from AIS and ACO algorithms in large-size problems is equal to 3.8156.

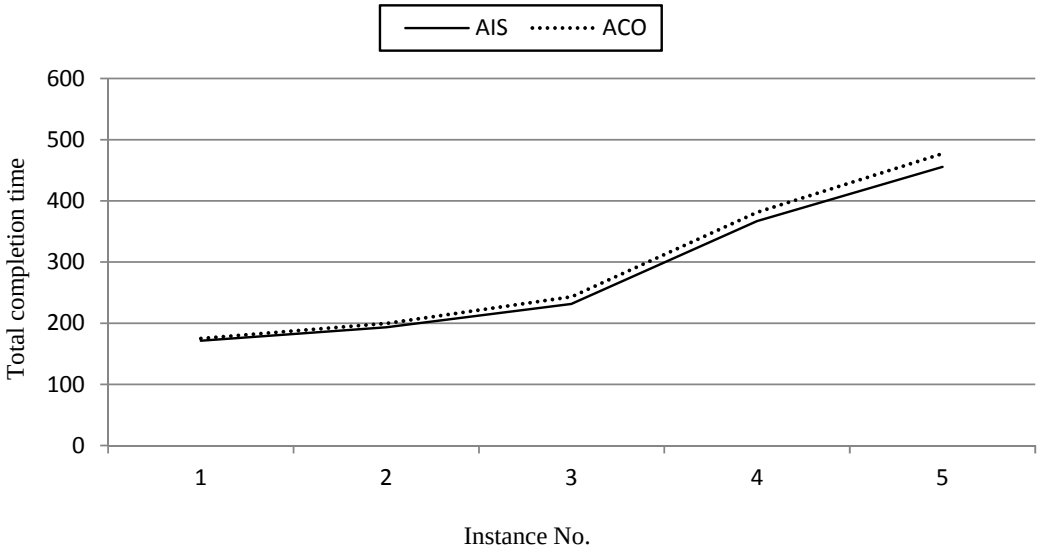


Fig. 7. Comparison of The Algorithms' Performance Based on Best of Objective Value

7. Conclusion and Future Research Directions

In this paper, we considered an identical parallel machines scheduling problem in which past-sequence-dependent setup times, job deterioration, and learning effect exist simultaneously. Since the problem under study is NP-hard, we have proposed two meta-heuristic algorithms namely: artificial immune system and ant colony optimization to minimize the total completion time. To evaluate the effectiveness and efficiency of the proposed algorithms, some random test problems in two scales were generated and results obtained by the algorithms were compared with each other. The results show that the proposed meta-heuristics were effective and efficient in solving small-size problems. In addition, we can see that the performance of AIS is consistently better than GA in terms of solution quality in large-size problems. Future research may focus on unrelated parallel machines scheduling problems with unequal release dates and effects of learning and deterioration. Besides, other meta-heuristic algorithms can be used to obtain better solutions.

References

[1] M. Pinedo, "Scheduling: theory, algorithms, and systems", *New York, Science Business Media, Springer,* (2012).

[2] A.S.Kunnathur, S.K.Gupta,"Minimizing the makespan with late start penalties added to processing times in a single facility scheduling problem*", European Journal of Operational Research*, Vol. 47, (1990), pp. 56-64,

[3] B.Alidaee, N.K.Womer, "Scheduling with time dependent processing times: Review and extensions", *Journal of the Operational Research Society*, Vol. 50, (1999). pp. 711-720,

[4] T.C.E.Cheng, Q.Ding, B.M.T.Lin, *"*A concise survey of scheduling with time-dependent processing times*"*, *European Journal of Operational Research*, Vol. 152, No. (1), (2004), pp. 1-13,

[5] S.Gawiejnowicz, "Time-Dependent Scheduling. Monographs in Theoretical Computer Science", Heidelberg, Berlin, Springer, (2008).

[6] T. G.Voutsinas, C.P.Pappis, "A branch and bound algorithm for single-machine scheduling with deteriorating value of jobs", *Mathematical and Computer Modelling*, Vol. 52, No. (2), (2010), pp. 55-61,

[7] T.C.E.Cheng, W.C.Lee, C.C.Wu, "Single-machine scheduling with deteriorating functions for job processing times", *Applied Mathematical Modelling*, Vol. 34, No. (12), (2010), pp. 4171-4178

[8] S.-H.Yang, J.B.Wang, "Minimizing total weighted completion time in a two-machine flow shop scheduling under simple linear deterioration", *Applied Mathematics and Computation*, Vol. 217, No. (9), (2011). pp. 4819-4826,

[9] Huang, X., & Wang, M.-Z., "*Parallel identical machines scheduling with deteriorating jobs and total absolute differences penalties*", Applied Mathematical Modelling, Vol. 35, No. (3), (2011), pp. 1349-1353.

[10] Zhao, C., & Tang, H., "*Parallel machines scheduling with deteriorating jobs and availability constraints*", Japan Journal of Industrial and Applied Mathematics, Vol. 31, No. (3), (2014), pp. 501-512.

[11] Biskup, D., "*Single-machine scheduling with learning considerations*", European Journal of Operational Research, Vol. 115, No. (1), (1999), pp. 173-178.

[12] Cheng, T. C. E., & Wang, G., "*Single machine scheduling with learning effect considerations*", Annals of Operations Research, Vol. 98, (2000), pp. 273-290.

[13] Biskup, D., "*A state-of-the-art review on scheduling with learning effect*", European Journal of Operational Research, Vol. 118, No. (2), (2008), pp. 315-329.

[14] Yin, Y., Xu, D., Sun, K., & Li, H., "*Some scheduling problems with general position-dependent and time-dependent learning effects*", Information Sciences, Vol. 179, No. (14), (2009), pp. 2416-2425.

[15] Lai, P.-J., & Lee, W.-C., "*Single-machine scheduling with general sum-of-processing-time-based and position-based learning effects*", Omega, Vol. 39, No. (5), (2011), pp. 467-471.

[16] Wang, X.-Y., Zhou, Z., Zhang, X., Ji, P., & Wang, J.-B., "*Several flow shop scheduling problems with truncated position-based learning effect* ",Computers and Operations Research, Vol. 40, No. (12), (2013), pp. 2906-2929.

[17] Yeh, W.-C., Lai, P.-J., Lee, W.-C., & Chuang M.-C., " *Parallel-machine scheduling to minimize makespan with fuzzy processing times and learning effects*", Information Sciences, Vol. 269, (2014), pp. 142-158.

[18] Cheng, T.C.E., Wu, C.-C., & Lee, W.-C., "*Some scheduling problems with deteriorating jobs and learning effects* ", Computers and Industrial Engineering, Vol. 54, No. (4), (2008), pp. 972-982.

[19] Koulamas, C., & Kyparisis, G. J., "*Single-machine scheduling with past-sequence-dependent setup times* ", European Journal of Operational Research, Vol. 187, No. (3), (2008), pp. 1045-1049.

[20] Graham, R.L., Lawler, E.L., Lenstra, J.K., & Rinnooy Kan, A.H.G., "*Optimization and approximation in deterministic sequencing and scheduling: A survey"*, Annals of Discrete Mathematics, Vol. 5, (1979), pp. 287-326.

[21] Chen, Z.-L.," *Parallel machine scheduling with time dependent processing times*", Discrete Applied Mathematics, Vol. 70, (1996), pp. 81-93.

[22] Lee, W.-C., "*A note on deteriorating jobs and learning in single-machine scheduling problems*", International Journal of Business and Economics, Vol. 3, (2004), pp. 83-89.

[23] Sun, L., "*Single-machine scheduling problems with deteriorating jobs and learning effects*", Computers and Industrial Engineering, Vol. 57, No. (3), (2009), pp. 843-846.

[24] Lee, W.-C., & Lai, P.-J., "*Scheduling problems with general effects of deterioration and learning* ", Information Sciences, Vol. 181, No. (6), (2011), pp. 1164-1170.

[25] Huang, X., Wang, M.-Z., & Ji, P., "*Parallel machines scheduling with deteriorating and learning effects* ", Optimization Letters, Vol. 8, (2014), pp. 493-500.

[26] Wang, X.-Y., & Wang, J.-J., "*Scheduling deteriorating jobs with a learning effect on unrelated parallel machines* ", Applied Mathematical Modelling, Vol. 38, (2014), pp. 5231-5238.

[27] Ji, M., Tang, X., Zhang, X., & Cheng T.C.E., *"Machine scheduling with deteriorating jobs and DeJong's learning effect",* Computers and Industrial Engineering, Vol. 91, (2016), pp. 42-47.

[28] Salehi Mir M.S., & Rezaeian, J., "A robust hybrid approach based on particle swarm optimization and genetic algorithm to minimize the total machine load on unrelated parallel machines " , Applied Soft Computing, Vol. 41, (2016), pp. 488-504.

[29] Kuo, W.-H., & Yang, D.-L., "*Single-machine scheduling with past-sequence-dependent setup and learning effects* ", Information Processing Letters, Vol. 102, No. (1), (2007), pp. 22-26.

[30] Zhao, C. & Tang, H., "*Single machine scheduling with past-sequence-dependent setup times and deteriorating jobs* ", Computers and Industrial Engineering, Vol. 59, No. (4), (2010), pp. 663-666.

[31] Hsu, C.-J., Kuo, W.-H., &Yang, D.-L., "*Unrelated parallel machine scheduling with past-sequence-dependent setup time and learning effects* ", Applied Mathematical Modelling, Vol. 35, No. (3), (2011), pp. 1492-1496.

[32] Wang, X.-Y., & Wang, J.-J., "*Scheduling problems with past-sequence-dependent setup times and general effects of deterioration and learning* ", Applied Mathematical Modelling, Vol. 37, No. (7), (2013), pp. 4905-4914.

[33] Liu, M., "*Parallel-machine scheduling with past-sequence-dependent delivery times and learning effect* ", Applied Mathematical Modelling, Vol. 37, No. (23), (2013), pp. 9630-9633.

[34] Kayvanfar, V., Komaki, G.M., Aalaei, A., & Zandieh, M., "*Minimizing total tardiness and earliness on unrelated parallel machines with controllable processing times* ", Computers and Operations Research, Vol. 41, No. (1), (2014), pp. 31-43.

[35] Rambod, M., & Rezaeian, J., "*Robust meta-heuristics implementation for unrelated parallel machines scheduling problem with rework processes and machine eligibility restrictions* ", Computers and Industrial Engineering, Vol. 77, (2014), pp. 15-28.

[36] Arnaout, J.-P., Musa, R., & Rabadi, G., "*A two-stage Ant Colony optimization algorithm to minimize the makespan on unrelated parallel machines—part II: enhancements and experimentations* ", Journal of Intelligent Manufacturing, Vol. 25, No. (1), (2014), pp. 43-53.

[37] Guo, P., Cheng, W., & Wang, Y., "*Parallel machine scheduling with step-deteriorating jobs and setup times by a hybrid discrete cuckoo search algorithm*", Engineering Optimization, Vol. 47, No. (11), (2015), pp. 1564-1585.

[38] Caniyilmaz, E., Benli, B., & Ilkay, M.S., "*An artificial bee colony algorithm approach for unrelated parallel machine scheduling with processing set restrictions, job sequence-dependent setup times, and due date*", International Journal of Advanced Manufacturing Technology, Vol. 77, Nos. (9-12), (2015), pp. 2105-2115.

[39] Dorigo, M., Maniezzo, V., & Colorni, A., "*Ant system: optimization by a colony of cooperating agents*", IEEE Transactions on Systems, Man and Cybernetics, Vol. 26, No. (1), (1996), pp. 29-41.

[40] Gajpal, Y., Rajendran, C., & Ziegler, H., *"An ant colony algorithm for scheduling in flowshops with sequence-dependent setup times of jobs ",* International Journal of Advanced Manufacturing Technology, Vol. 30, Nos. (5-6), (2006), pp. 416-424.

[41] Rossi, A., & Dini, G., *"Flexible job-shop scheduling with routing flexibility and separable setup times using ant colony*

optimisation method ", Robotics and Computer-Integrated Manufacturing, Vol. 23, No. (5), (2007), pp. 503-516.

[42] Liao, C.-J., & Juan, H.-C., *"An ant colony optimization for single-machine tardiness scheduling with sequence-dependent setups ",* Computers & Operations Research, Vol. 34, No. (7), (2007), pp. 1899-1909.

[43] Arnaout, J.-P., Rabadi, G., & Musa, R., "*A two-stage Ant Colony Optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times* ", Journal of Intelligent Manufacturing, Vol. 21, (2010), pp. 693-701.

[44] Mirabi, M., *"Ant colony optimization technique for the sequence-dependent flowshop scheduling problem",* International Journal of Advanced Manufacturing Technology, Vol. 55, Nos. (1-4), (2011), pp. 317-326.

[45] Zhang, J., & Liu, G., *"Hybrid ant colony algorithm for job shop schedule with unrelated parallel machines ",* Advanced Materials Research, Vols. 430-432, (2012), pp. 905-908.

[46] Sun, J.U., *"An Ant Colony Optimization Algorithm for the Press Shop Scheduling Problem ",* Mechatronics and Industrial Informatics, Book Series: Applied Mechanics and Materials, Vols. 321-324, (2013), pp. 2116-2121.

[47] Rossi, A., *"Flexible job shop scheduling with sequence-dependent setup and transportation times by ant colony with reinforced pheromone relationships ",* International Journal of Production Economics, Vol. 153, (2014), pp. 253-267.

[48] Arnaout, J.-P., Musa, R., & Rabadi, G., "*A two-stage Ant Colony optimization algorithm to minimize the makespan on unrelated parallel machines—part II: enhancements and experimentations* ", Journal of Intelligent Manufacturing, Vol. 25, No. (1), (2014), pp. 43-53.

[49] Naderi, B., Khalili, M., & Tavakkoli-Moghaddam, R., "*A hybrid artificial immune algorithm for a realistic variant of job shops to minimize the total completion time*", Computers and Industrial Engineering, Vol. 56, No. (4), (2009), pp. 1494-1501.

[50] Al-Anzi, F.S., & Allahverdi, A., "*An artificial immune system heuristic for two-stage multi-machine assembly scheduling problem to minimize total completion time* ", Journal of Manufacturing Systems, Vol. 32, No. (4), (2013), pp. 825-830.

[51] Lin S.-W., & Ying, K.-C., "*Minimizing makespan in a blocking flowshop using a revised artificial immune system algorithm* ", Omega, Vol. 41, No. (2), (2013), pp. 383-389.

[52] Abdollahpour, S., & Rezaian, J., "*Minimizing makespan for flow shop scheduling problem with intermediate buffers by using hybrid approach of artificial immune system* " , Applied Soft Computing, Vol. 28, (2015), pp. 44-56.

[53] Farmer, J.D., Packard, N.H., & Perelson, A.S., *" The immune system, adaption, and machine learning"*, Physica D: Nonlinear Phenomena, Vol. 22, (1986), pp. 187–204.

[54] Shahrizal Muhamad, A., & Deris, S., *"An artificial immune system for solving production scheduling problems: a review"*, Artificial Intelligence Review, Vol. 39, (2013), pp. 97-108.

[55] Dorigo, M., & Stützle, T., "*Ant colony optimization: overview and recent advances"*, International Series in Operations Research and Management Science, Vol. 146, (2010), pp. 227-263.

[56] Keskinturk, T., Yildirim, M.B., & Barut, M., "*An ant colony optimization algorithm for load balancing in parallel machines with sequence-dependent setup times"*, Computers and Operations Research, Vol. 39, No. (6), (2012), pp. 1225-1235.