Decision Science Letters 5 (2016) 535–550

Contents lists available at GrowingScience

Decision Science Letters

homepage: www.GrowingScience.com/dsl

Integrating the sequence dependent setup time open shop problem and preventive maintenance policies

K. Naboureh[a] and E. Safari[b*]

[a]*Department of Industrial Engineering, Iran University of Science and Technology, Tehran, Iran*
[b]*Research Faculty for ICT Policy Making and Strategic Management, Iran Telecommunication Research Center, Tehran, Iran*

C H R O N I C L E

Article history:
Received February 25, 2016
Received in revised format:
March 28, 2016
Accepted April 19, 2016
Available online
April 20 2016

Keywords:
Open shop
Meta heuristics
Preventive maintenance
SDST
Immune Algorithm

A B S T R A C T

In most industrial environments, it is usually considered that machines are accessible throughout the planning horizon, but in real situation, machines may be unavailable due to a scheduled preventive maintenance where the periods of unavailability are known in advance. The main idea of this paper is to consider different preventive maintenance policies on machines regarding open shop scheduling problem (OSSP) with sequence dependent setup times (SDST) using immune algorithm. The preventive maintenance (PM) policies are planned for maximizing availability of machines or keeping minimum level of reliability through the production horizon. The objective function of the paper is to minimize makespan. In total, the proposed algorithm extensively is compared with six adaptations of existing heuristic and meta-heuristic methods for the problem through data sets from benchmarks based on Taillard's instances with some adjustments. The results show that the proposed algorithm outperforms other algorithms for this problem.

1. Introduction

One of the most important tasks carried out in industry is scheduling. Scheduling problems are defined as allocating of resources to complete a set of activities in a period of time (Baker, 1974). In common sense, scheduling problems have been classified into those with batching and non-batching considerations, and with sequence-independent and sequence-dependent setup times. The other categories available in the literature have been described based on shop environments, including single machine, parallel machines, flow shop, no-wait flow shop, flexible flow shop, job shop, open shop, and others. In this paper, we are interested in studying the open shop problem which can be defined as follows: suppose there are a set of m machines $\{M_j \mid 1 \le j \le m\}$, a set of n jobs $\{J_i \mid 1 \le i \le n\}$, and a set of mn operations $\{O_{ij} \mid 1 \le i \le n, 1 \le j \le m\}$. Each operation O_{ij} is corresponded to a specific job j_i

* Corresponding author.
E-mail address: e.safari@itrc.ac.ir (E. Safari)

doi: 10.5267/j.dsl.2016.4.002

processed on a specific machine M_j for a given amount of time t_{ij}, which is a nonnegative integer. In open shop problem, there is no order for processing of operations of each job. In other word, this assumption is essential difference between open shop and other scheduling problems such as flowshop and jobshop (Allahverdi et al., 2008).

The majority of researches on the scheduling literature have assumed that the setup time is negligible or part of the job processing time. While this hypothesis makes easier the analyses of the problems, it seriously has an effect on solution quality of many scheduling problems that needs a clear treatment for setup times (Allahverdi et al., 2008). As stated earlier, we take into account the existence of sequence dependent setup times for the proposed study of this paper.

In most scheduling problems, it is usually assumed that the machines are permanently accessible in planning horizon, while in practice this does not hold for many reasons such as downtime and maintenance operations. In this paper, the main reason to machine unavailability is based on two popular preventive maintenances (PMs) (a set of operations to maintain equipment and facilities in satisfactory operating condition by providing systematic inspection, detection, and correction of incipient failures either before they occur or before they develop into major defects) techniques, which are extensively employed in industry.

In this article, we intend to schedule jobs and maintenance operations simultaneously in open shop problem with SDST constraint.

The main goal of the paper is to provide tools that allow openshop scheduling problem with SDST constraint by implicitly taking into assumption the necessary maintenance operations to achieve a high level of reliability in machines. Preventive maintenance (PM) operations are first computed and then sequenced along with the operations to be performed in the openshop. To the best of our knowledge, this is the first paper that integrates the PM policies in the openshop problem with SDST constraint. The main reason of doing this research is that considering the PM policies in the openshop problem affects the performance of the available algorithms. In other words, current algorithms for regular openshop problems may not necessarily be efficient in real-world and industrial environments.

There have been tremendous efforts to introduce different methods to solve open shop problem. In spite of some special cases of open shop scheduling problems which are still polynomially solvable, This problem is NP-Hard for M>2 in general (Garey & Johnson, 1979). The methods and approaches to solve open shop problem may be divided into two groups: The first one is searching for the exact optimal solution, while, in the second one, a heuristic is applied to obtain the near optimal solution. Heuristic techniques have been proposed to solve open shop characterized as either constructive or improvement. Constructive methods start the construction of solution by determining one operation schedule of specific jobs and then schedule other operations of jobs one by one until scheduling of all operations of jobs finish perfectly. Improvements methods start from a solution generated using some heuristic or randomly and, by means of proper schemes, improve the solution to obtain the global optimum. Improvement methods are generally derived from meta-heuristic methods such as Immune algorithms (IA), genetic algorithm (GA), Simulated annealing (SA) and so on. For exact methods, Brucker et al. provided a method based on branch and bound for the general m-machine OSSP (Brucker et al., 1997). Dorndorf et al. developed a branch and bound method to tackle OSSP in which the search space is reduced using constraint propagation methods (Dorndorf et al., 2001). For constructive methods, Blum and Sampels (2004) provided an ant colony optimization algorithm, which uses a strong neighborhood search for constructing solutions and a local search to improve the quality of solutions, for OSSP. In iterative based algorithms, we may provide some examples. Liaw (2000) developed a hybrid genetic algorithm based on integrating 9 TS and a basic GA to provide good quality solution for OSSP with makespan minimization criterion. Other proposed algorithm was provided by Prins (2000) in which the proposed GA is able to enhance the upper bound for the benchmark from the literature.

Sha and Hsu (2008) proposed a novel particle swarm optimization in which there are four decoding schemes based on the principal of active and non-delay schedules. Andersen et al. (2008) solved OSSP using two metaheuristics, simulated annealing and genetic algorithms, with some operators. Roshanei et al. (2010) considered non-preemptive open shop scheduling problem with SDST on each machine for minimizing makespan. These researchers were the first people who integrated the OSSP with STSD constraint. Also they proposed two new advanced metaheuristics: multi-neighborhood search, simulated annealing and hybrid simulated annealing to solve the problem at hand. Next stage, for the first time, they adapted two well-known constructive heuristics: longest total processing time and longest total remaining processing from the literature to consider the case of SDSTs. Their conducted experiments showed that the proposed algorithms could outperform other adapted algorithms.

Burton et al. (1989) presented a simulation based study of the performance of a job shop where machines were subject to failure. The effectiveness of some maintenance techniques under shop load, job sequencing rule, and preventive maintenance policy and maintenance capacity was evaluated. Li et al. (19998) studied dynamic job shop rescheduling where machines are not permanently available due to breakdown or preventative maintenance. A simulation model using the AWESIM discrete event modeling tools was developed. Also they developed the PMBD heuristic that relies only on predicted downstream machine failure times. Espinouse et al. (2001) studied two-machine and no-wait flow shop problem with machine availability constraint for minimizing of makespan as objective function. They only considered the deterministic case where the unavailable periods are known in advance. They showed that the problem is strongly NP-hard even if only one non-availability period occurs on one of machines. Finally they also provided heuristic algorithms with error bounding analysis. Bazewicz et al. (2002) considered open shop scheduling problems with limited machine availability. Three types of jobs were studied: non-preemtable, resumable and preemptible.

Aggoune (2004) considered the scheduling of a flow shop with availability constraints (FSPAC) due to a preventive maintenance activity. In this paper, he considered two variants of the non-preemptive FSPAC. Since the FSPAC is strongly NP-hard, a heuristic method based on a GA and TS was used to approximately solve the makespan minimization problem. Allaoui et al. (2004) investigated the hybrid flow shop scheduling problem with maintenance constraints with different objective functions such as flow time, due date, setup, cleaning and transportation times. Allaoui and Artiba (2004) considered a two-stage hybrid flow shop where each machine is subject to at most one unavailability period. They first discussed the complexity of the problem and then gave the Branch and Bound model for this problem. Zhou et al. (2007) incorporated production and preventive maintenance in flow shops, and built an optimum period model of preventive maintenances for maximizing the machines availability on production horizon. Also they proposed a heuristic algorithm to solve the provided model. Experimental results showed that the heuristic algorithm had good performance in resolving the problem.

Sortrakul et al. (2005) developed some heuristics using genetic algorithms to tackle an integrated optimization model for production scheduling and preventive maintenance planning. The experimental results on several problem sizes showed that the developed genetic algorithms were very effective to optimize the integrated problem. Guo et al. (2007) provided an experimental model to evaluate the impact of corrective and preventive maintenance on scheduling performance in the presence of machine breakdown with the objective of minimizing schedule duration. Ruiz et al. (2007) considered two most popular preventive maintenance policies to maximize the availability or to maintain a minimum reliability during the planning horizon. The results for six adaptations of existing heuristic and meta-heuristic methods showed that PACO and GA provided very effective solutions. Benbouzid-Sitayeb et al. (2008) considered integration of production and preventive maintenance scheduling problem in permutation flowshop with the objective of minimizing the makespan. They proposed a sequential algorithm including two steps.

Allaoui et al. (2008) studied the simultaneously scheduling of *n* jobs and the preventive maintenance in a two-machine flowshop with the objective of makespan under non-resumable case. They assumed that one of the both machines may be repaired during the first *T* periods of scheduling. Yulan et al. (2008) simultaneously considered five objectives, including minimizing maintenance cost, makespan, total weighted completion time of jobs, total weighted tardiness, and maximizing machine availability to optimize the integrated problem of PM and production scheduling for a single machine. Multi-objective genetic algorithm (MOGA) was employed to tackle this problem. A numerical example revealed the urgency and circumstance of integrating optimization of PM and production scheduling considering multiple objectives. Naderi et al. (2009) investigated the flexible flow line problem with sequence dependent setup times and different preventive maintenance policies. The optimization criterion is the minimization of makespan. In this paper, they provided a novel variable neighborhood search (VNS) as well as the adaptations of some existing high performing meta-heuristics in the literature.

Kubzin et al. (2009) considered the two-machine flow shop scheduling problem where the machines are unavailability during a period. They considered both the resumable scenario and the semi-resumable scenario and provided approximation algorithm for them. Ghodratnama et al. (2010) presented a new nonlinear multi-objective mathematical model for the single-machine scheduling problem where some of the constraints such as repairing and maintenance periods, deterioration of jobs, and learning effect of the work process were incorporated in the model. An algorithm based on simulated annealing (SA) was provided for solving the problem. The algorithm was compared with those results reported by the Lingo 8 to show the efficiency and capability of the proposed algorithm. Jabbarizadeh et al. (2009) considered the hybrid flexible flowshop with SDST and machine availability, caused by preventive maintenance, constraints. They proposed three heuristics and two metaheuristics to tackle the problem.

Safari et al. (2010, 2011) considered the flowshop problem where machines are subject to condition based maintenance. They provided two hybrid meta-heuristic algorithms to tackle the NP-hardness of the problem. Overall, some adaptations of the existing meta-heuristic and the heuristic methods were evaluated and they showed that the proposed meta-heuristic algorithms had good performance with respect to other existing methods. Zammoria et al. (2014) focused on the scheduling problem with SDST in which the single machine is subject to failures. To increase availability level, they scheduled the maintenance activities as well as jobs. Furthermore, the integration of harmony search and genetic algorithms was used to solve the problem. Sarkera et al. (2013) first provided a hybrid evolutionary algorithm to solve the job scheduling problem and then examined the effect of machine maintenance including preventive and breakdown on the job scheduling. For the breakdown case, the algorithm was revised to combine a rescheduling option after the breakdown occurs. The experimental provided a well understanding of job scheduling and the needed rescheduling actions under process disruption. Vahedi-Nouria et al. (2014) proposed a mixed integer linear programming model for the non-permutation flow shop scheduling problem with learning effects and machine availability constraints. They also presented an effective heuristic to find a proper non-permutation solution. Kaplanoğlu (2014) provided a collaborative multi-agent based method to schedule single machine with maintenance and SDST constraints. The problem was solved under the case of both regular and irregular maintenance activities. The obtained solutions were compared with several static single machine scheduling problem sets available in the literature.

In this article, since our review shows that none of the researchers has investigated the OSSP with STSD and PM constraints, we obviate the gap between the theory and the practice in the OSSP with SDST constraint by the integration of PM operation in our problem. Since this problem is strongly NP-hard, we propose an effective meta-heuristics based on Immune Algorithm to solve the problem effectively. The next stage is devoted to conduct experiments for tuning the algorithm parameters using ANOVA. At the end proposed algorithm is compared with existing meta-heuristics in literature.

In Section 2, we explain the preventive maintenance techniques along with the assumptions of problem which are used to this paper. The proposed algorithm is explained in section 3. Section 4 is devoted to explore the experiments results. Finally, Section 5 includes conclution of the paper and provides some directions for future works.

2. Preventive maintenance and problem assumptions

2.1 Preventive maintenance

In recent years, progress in technology results in development of highly sophisticated products, and increasing customer expectations provide new pressures on manufacturers to produce high-quality products. So to achieve these, production systems should have high reliability which may be defined as the probability that a system would perform its intended function under operating conditions for a specified period of time. Therefore, the reliability function *R(t)* may be expressed as follows (Rausand & Høyland, 2004; Ushakov, 1994; Nakagawa, 2006: Birolini, 2007; Meeker & Escobar, 2014):

$$\Pr\{X > t\} = 1 - F(t) = \int_t^{\infty} f(u)du \tag{1}$$

where X, f and F are failure time, failure density function and failure time distribution, respectively.

Maintenance operations (set of operations is done to restore a condition of equipment in to previous or initial state) have very important role in achieving systems with high reliability. System reliability can be assessed by reliability of unit and configuration of system, and can be enhanced by accepting some appropriate maintenance policies. In particular, the following three policies are generally used.

(1) Repair of failed units
(2) Provision of redundant units
(3) Maintenance of units before failure

The first policy is termed corrective maintenance (CM) and accepted in the case where units can be repaired and their failures do not unfavorably influence a whole system. If units fail then they may start to be repaired instantly or may be scrapped. After the repair is finished, units can work again. The second policy is done in the case where system reliability can be enhanced by supplying surplus and spare units. In particular, standby and parallel systems are well-known and used in practice. The third is called PM and defined as scheduling of planned maintenance actions aimed at the prevention of breakdowns and failures. The main goal of PM is to prevent the failure of equipment before it truly takes place. It is designed to maintain and improve equipment reliability by substituting damaged components before they actually fail. PM operations contain equipment checks, partial or complete overhauls at specified periods, oil changes, lubrication and so on. In general, the ideal PM program would prevent all equipment failure before it occurs.

One of the most popular distribution function used to model time to failure of an equipment (*X)* is Weibull distribution function with two parameters, $X \approx W(\theta, \beta)$, where β and θ are called shape parameter and scale parameter, respectively. Based on Weibull distribution function, policy II and policy III are formulated for employing in remaining of this paper.

Policy I: *preventive maintenance at fixed predefined time intervals*
In this type of policy, the PM operations are executed according to predefined time intervals without considering probabilistic models for the time to failure.

Policy II: *optimum period model for the preventive maintenance maximizing the machines' availability*

In this type of policy, PM is done according to the optimal maintenance period. In this paper the time to failure is considered to follow Weibull probability distribution, with $\beta>1$. t_r is time needed to rapier a machine and t_p is the time needed to maintain a machine. The PM action recovers machine state to the as-good-as-new condition, which permits to model the operation cycle and the PM as a renovation process. According this assumption and Cassady and Kutanoglu (2003), the optimal maintenance interval T_{PMop} is obtained by means of:

$$T_{PMop} = \theta(\frac{t_p}{t_r(\beta-1)})^{1/\beta} \quad (2)$$

Policy III: maintaining a minimum reliability threshold for a given production period t

In this policy, it is supposed that the failure rate of a system is rising with time and therefore it can be influenced by failures due to aging or wear. This policy provides a systematic PM after a time T_{PM} to ensure a minimum reliability of the system from time $t = 0$. In this paper, it is considered that PM restores the machine to the as-good-as-new state. In this case, PM will be done at regular intervals $0, 1T_{PM}, 2T_{PM}, 3T_{PM}, \ldots, nT_{PM}$ that are considered as renovation points. Based on Ruiz et al. calculation (Ruiz et al., 2007), the time between PM is calculated as follows:

$$T_{PM} = [-\theta^{\beta} \frac{\ln R_0(t)}{t})^{1/(\beta-1)} \quad (3)$$

2.2 Problem assumptions

Main assumptions of the problem can be described as:

- A machine cannot process two jobs at the same time.
- A job includes several operations, each of which has to be done on a particular machine.
- All jobs are accessible for processing at time zero.
- Setup times between operations are not negligible.
- A processing of jobs is done without error.
- All parameters of problem are known in advance.
- Each job is processed on a machine at the same time.
- There are no preference constraints among the jobs.
- Infinite buffers exist between machines.
- Machines are not available at some times because of PM operations.
- Level of system reliability is specified in advance.

3. Immune algorithm

Artificial Immune System (AIS) was invented in the mid-1980s by Farmer et al. (1986). However, it was in the mid 90s that AIS became a subject area in its own right. The immune system is the basic and remarkable that protects body against bacteria, viruses. To begin, we are going to bring a few characteristics of the human adaptive immune system. A number of concepts and technical terms can be introduced to make the clear the terminology for reader from Janeway et al. (2000). Many AIS algorithms have been proposed to find solutions to a widespread class of complex problems. AIS has been applied to the areas such as clustering (Karaboga & Ozturk, 2011; Abdi et al., 2012; Duma et al., 2013), anomaly detection (Gonzalez & Dasgupta, 2003), network security (Forrest & Hoffmeyr, 2000), optimization (Gao & Fang, 2009), scheduling (Engin & Döyen, 2004; Anandaraman et al., 2012), etc. The immune system is a dramatic and complex set of cells, molecules and organs with the primary role of defending of the host organism by pathogens (called antigens, Ag), which brings out an immune response. One of the most important types of the response is the secretion of antibody molecules by B cells. Antibodies (Ab) are Y-shaped receptor molecules posed on the outside of B cell with the basic role of recognizing and binding, during a complementary match, with an antigen. The Ab recognizes a portion of the Ag called its epitope. An idiotype is defined as the set of epitopes displayed by the

variable regions of a set of Abs. While each B cell has a single type of Ab, thus being called mono-specific, an Ag naturally has several different types of epitopes, and can be identified by different antibodies. The antibody portion responsible for recognizing an antigen is termed paratope, also known as V-region, for variable region. Since Ab can alter its shape to achieve a better match with a given Ag, It is variable. The power and specificity of the Ag-Ab interaction is determined by the affinity of their match. Fig. 1 illustrates an Ag with its many epitopes and an Ab with its paratope and idiotope.

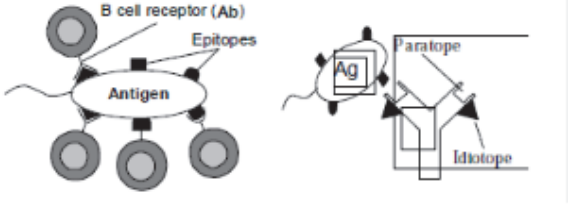

Fig. 1. B cell, antigen, antibody, epitopes, paratopes, idiotopes

It can generate millions of antibodies from hundreds of antibody genes and can keep animals which are infected by antigens. It has complex mechanisms that recombine the gene to deal with the foreign antigens, generate the antibodies and bind the antigens. The AIS and IA were motivated by the immune system. Similar to Genetic Algorithm (GA), IA begins by generating antibodies randomly in a solution space, and finally reaches the optimum via natural selection, crossover and mutation. Compared with GA, IA has an affinity function, which is used to describe the relationship between antibodies. The higher affinity, the stronger binding, thus the better immune detection and response.

3.1 The proposed Immune algorithm

In this paper, the generated solutions for this problem are generally named antibodies. Initial antibodies are randomly generated on a solution space. The cross over and mutation are used to obtain new antibodies in an iterative manner until a pre-determined time have been satisfied. An affinity calculation between antibodies is also located within the algorithm to suppress similar antibodies. Through IA procedure, the fittest antibody is considered as the solution to the SDST open shop scheduling with preventive maintenance.

1. Initialization:
   a) Parameter setting: set the number of initial population (*pop_size*), probability of crossover (*pc*), the number of clone, affinity threshold (*at*), and affinity adjustment (*aa*).
   b) Randomly generate an initial population of (*pop_size*) antibodies.
2. Objective function calculation: calculate the fitness function for each of the antibodies.
3. Mating pool creation:
   a) The best one selection: throw the best known antibody based on fitness in mating pool.
   b) Affinity calculation: calculate the affinity between each antibody with the best known antibody obtained so far.
   c) Similar antibodies suppression: if some antibodies have an affinity value higher than a predetermined threshold (*at*), then decrease the probability allocated to those and normalizes the probability.
   d) Mating pool expansion: select with replacement (*pop_size*) antibodies from the full population. The antibodies are chosen based on their fitness. Those antibodies have the higher fitness value being selected with high probability.
4. Crossover operation: select *pop_size* * *pc* pairs of parents from mating pool using tournament selection mechanism, and perform crossover operators on the selected parents.

5. Select *pop_size /2* solutions from generated offspring using crossover operator, and then select *pop_size* /2 of solution from mating pool.
6. Create 3 clone of *pop_size* selected solution.
7. Apply 20 times swap, shift and multi point mutations on each of clones and keep best obtained solution.
8. Fitness calculation: compute the fitness values for the new generated antibodies.
9. Select pop_size of best antibodies among total of available antibodies (old antibody and new antibodies).
10. Termination test: terminate the algorithm if the predefined stopping time is met; else return to step 3.

3.1.1 Antibody representation

In this paper, we encode OSSP schedules in permutations schema. In permutation representation, a number between 1 to *nm* is assigned to each of the operations of all jobs. In this assignment, numbers from 1 to *n* are assigned to the operations of job1; *n+1* to *2n* to the operations of job 2 and so on. For example: Let *n=m=3*, the *X= (4, 8, 6, 1, 9, 7, 2, 3, 5)* denotes a solution in which for example 4 , 8, 6 represent the first , second, third operations of job1, respectively .

3.1.2 Fitness evaluation

In general, fitness function denotes the effectiveness of an antibody or a solution provided by an IA. Antibodies are evaluated using an objective function to obtain their fitness values. Fitness values are used to generate selection probability where an antibody with high fitness value has higher probability for selection (minimization problem). Sometimes, the problem structure is in such a way that an antibody with low fitness value has higher probability for selection. Thus a transformation is necessary to convert the antibodies fitness value to before mentioned problem structure. In our case, the makespan is to be minimized; a candidate solution with higher makespan is assigned lower fitness value. Here, for antibody *i*, it is calculated as follow:

$$f(i) = \frac{\frac{1}{C_{\max}(i)}}{\sum_{j=1}^{po\_size} \frac{1}{C_{\max}(j)}}$$

where *f(i)* and $C_{max(i)}$ are fitness value and makespan for antibody *i*, respectively.

3.1.3 Crossover operator

The crossover operator generates new sequences or offspring by combining two other sequences or parents. The goal is to generate “better” offspring, i.e. creating better sequences after crossing the parents. Many different general and specific crossover operators have been proposed for the OSSP. There should be neither repeated nor missing elements otherwise the sequence would be illegal. The first operator is called “Similar Job Order Crossover” or SJOX and can be explained as follows: First, both parents are examined on a position-by-position basis. Identical jobs at the same positions are copied over to both offspring. Then, each offspring directly inherits all jobs from one of the parents up to a randomly chosen cut point. That is to say, Offspring 1 inherits directly from Parent 1 and Offspring 2 from Parent 2. Lastly, the missing elements of each offspring are copied in the relative order of the other parent. The second crossover operator is called “two point crossovers” which can be expressed as: A pair of random points is selected along the length of first parents. Then, the jobs outside the

selected two points from first parent are copied into child and the remaining jobs of the child are copied from the second parent in the order of their appearance.

3.1.4 Mutation operator

A mutation operator is applied to avoid premature convergence to local optimum. This operator can also be seen as a simple form of local search. Doing a mutation operator along with an individual, we slightly change the sequence, thus allowing a new but similar permutation. Mainly, three different mutation operators are proposed in the literature for permutation encodings:

- SWAP mutation: Two positions are randomly chosen and their corresponding operations swapped.
- SHIFT mutation: In this case, a randomly selected operation in the sequence is relocated to another randomly picked position. The operations between these two positions move along.
- Multi point mutation: In this case, three randomly selected positions in the sequence is relocated to another three randomly selection position. The operations between them move along.

3.1.5 Selection scheme and similar antibodies suppression

Selection is a process by which some of the current populations are selected for improvement. The selection process firstly ranks solutions based on the objective function. Then, a selection function uses each antibody's rank to decide which antibodies will survive to the next generation. The method that is used to select members for survival is important to achieve the best solution, and seriously influences the diversity of solution and solution convergence. Two popular selection procedures mentioned in literature are: tournament and roulette wheel.

Similarity of the candidate solutions are compared with the best-known solutions obtained so far. This allows an affinity value, which expresses the similarity between candidate solutions with the known best solution, to be computed using entropy theory. The details of the entropy theory are outlined in the later section. Candidate solutions with the higher fitness value are given the higher probability to be selected for the production of the next generation of candidate solutions. The assignment of probability values is performed using the roulette wheel method. To assist the search, an IA uses both accelerating mechanism and restraining mechanism. Basically, the accelerating mechanism works as follows; a candidate solution with the best fitness is recorded, which will be used to seed the mating pool. This mechanism is to ensure that the mating pool contains a large proportion of candidate solutions with good properties. On the other hand, the restraining mechanism performs the following tasks. If a candidate antibody has an affinity value higher than a prescribed threshold value, then these two antibodies are called similar ones. To maintain diversity, it is assigned a lower probability by multiplying the probability obtained previously from the fitness value criterion with a factor of less than one. This will reduce the probability of being selected. Such a mechanism is built into an IA to prevent a good candidate antibody from becoming overly dominant. Without the restraining mechanism, a good candidate antibody may be able to dominate the mating pool prematurely. As a result, it may restrict the search space for candidate antibodies.
Entropy theory is employed here to estimate the probability of the recurrence of patterns from an information source. Abramson (1963) defined the information entropy, H(x), of a discrete random variable $X = \{x1,x2, \ldots ,xn\}$ with probability mass function $P(X = xi) = pi,\ i = 1,2,\ldots ,n$ as:

$$H(x) = -\sum_{i=1}^{n} p_i \log p_i$$

Using information entropy, the similarity of antibody *i* (or sequence i) in relation to a reference antibody (or sequence) can be expressed as follows:

$$aff(i) = \frac{1}{1 + \frac{1}{k}\sum_{j=1}^{k} h_{ij}}$$

where *aff(i)* is the measure of similarity, k is the sequence size, and $h_{ij} = p_{ij}\ log p_{ij}$.
If $x_j = x_j$ then $p_{ij} = 1$; thus, $h_{ij} = -log\ 1 = 0$. On the other hand, if $xj \neq xj$ then $p_{ij} = 0.5$; this implies that $hij = 0.5\ log\ 0.5 \approx 0.151$.

4. Computational results

In this section, we evaluate the proposed algorithm, i.e. IA, and other existing algorithms for OSSP without considering preventive maintenance constraint in literature i.e MNSSA, HSA, SA and random (a simple algorithm that creates random solutions and returns the best sequence generated as a result). Thus, in this section, first we describe how to generate data for experiments, second IA algorithm is tuned and then we apply IA algorithm to efficient comparison. We implement the algorithms in MATLAB 7.0 and run on a PC with Core2 Duo 3.0 GHz and 2GB of RAM.

Data required for the problem includes the number of jobs, the number of machines, the range of processing times, the range of the sequence-dependent setup times, the maintenance interval and PM duration. We generate processing time for three combinations of number of machines (m) and number of jobs (n): (10, 10), (15, 15) and (20, 20) based on a well-known Taillard (1993)'s instances. The processing times are obtained from a uniform distribution over (1, 99). For each combination (n,m), there are 10 instances. Since Taillard's benchmark does not include SDST, maintenance interval and PM duration, we randomly generate SDST and PM duration. Also maintenance interval is generated using presented formulation for these parameters. We generate five instance sets for the SDST, ratios of the setup times to the processing times are varied such that the sequence-dependent setup times are at most 25%, 50%,75%, 100% or 125%, respectively, of the maximum processing times. For each combination, $\beta = \{2,10,100\}$ is defined and the duration of the PM operations (D_{PM}) as U[1, 24], U[1, 49], *U[1, 74], U[1, 99], U[1, 124]* and U[1, 149]. That is, there are six ranges which the average T_{PM} is 25%, 50%, 75%, 100%, 125% or 150% of the average processing times.

For the case of policy II, t_p is set to 1 and t_r to 8 for all the experiments. θ *is* set according to the number of jobs and SDST ranges (Table 1). The levels of θ are selected to make sure that the considerable number of PM operations would be performed in each machine. Therefore, we have total of 2700 instances for different combinations of n, m, β and D_{PM}.

For the case of policy III, The objective of policy III is to maintain a minimum level of reliability for a production period t. The interval of the PM operations *(T_{PM})* is given by Eq. (3). In this case, we have four parameters: θ, β, $R_0(t)$ and t. Since both policies II and III are not comparable, it is necessary to define a new set of instances to evaluate the effectiveness of the different adapted methods. The same configurations of n, m, θ, β and D_{PM} as in the case of policy II are considered, and therefore a set of 2700 instances is also obtained. The aim is 95% reliability after the production period t, therefore $R_0(t)$=0.95. In order to calculate T_{PM} it is still necessary to determine period t, which can be easily obtained from the processing times (t_{ij}) of the instances. If we suppose that the average processing time and average STSD are t and p, the t parameter can be calculated using $t=n(t+p)$ Thus, we set different values for t according to Table 2.

Table 1
different θ values

n	m	expected STSD	θ
10	10	25	400
10	10	50	525
10	10	75	625
10	10	100	800
10	10	125	900
15	15	25	575
15	15	50	750
15	15	75	900
15	15	100	1150
15	15	125	1400
20	20	25	750
20	20	50	1000
20	20	75	1300
20	20	100	1550
20	20	125	1800

Table 2
different θ and $n(p+t)$ values

m	n	n(p+t)	θ
10	10	750	400
10	10	1000	525
10	10	1250	625
10	10	1500	800
10	10	1750	900
15	15	1125	575
15	15	1500	750
15	15	1875	900
15	15	2250	1150
15	15	2625	1400
20	20	1500	750
20	20	2000	1000
20	20	2500	1300
20	20	3000	1550
20	20	3500	1800

Once the makespan C_{max} of each algorithm has been obtained for all the instances, we evaluate the best solution obtained for each instance by any of the 5 algorithms, which we call S_{best}. With this, we transform the results to calculate the relative percentage deviation with respect to this best solution with the following expression:

$$\text{relative percentage deviation (RPD)} = \frac{S - S_{best}}{S_{best}}.100$$

where S is the C_{max} obtained for a given algorithm and instance. This *RPD* measure is straightforward way for comparing algorithms. Obviously, lower values of *RPD* are preferred.

4.1 Parameter tuning

For the proposed IA algorithm, four parameter pop_size, pc, al and aa are considered to tune. The considered levels for these factors are as follows:

pop_size: 10; 20; 30.
pc: 0.8 0.9 0.95.
at: 0.94; 0.96;0.98.
aa: 0.1; 0.2; 0.5.

A set of 100 instances is selected randomly among total instances. All the 100 instances are solved by IA for all levels of considered factors. We use ANOVA test to analyze the results. It is necessary to notice that for using ANOVA, three main hypotheses, normality, homogeneity of variance and independence of residuals are checked. The means plot and LSD intervals for the different levels of parameters *pop_size*, *pc*, *at* and *aa* are shown in Figs. (2-5). As it could be seen, the levels that are selected to *pop_size*, *pc*, *al* and *aa* to generate the best solution are 10, 0.8, 0.98 and 0.5, respectively.

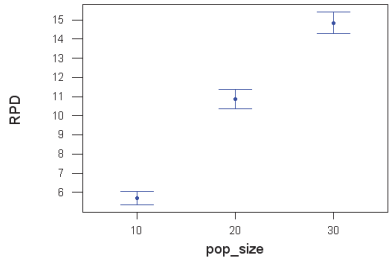

Fig. 2. Means plot and LSD intervals (at the 95% confidence level) for the different levels of parameter pop_size

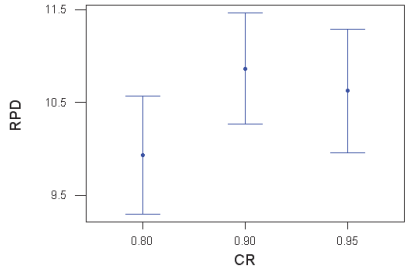

Fig. 3. Means plot and LSD intervals (at the 95% confidence level) for the different levels of parameter CR

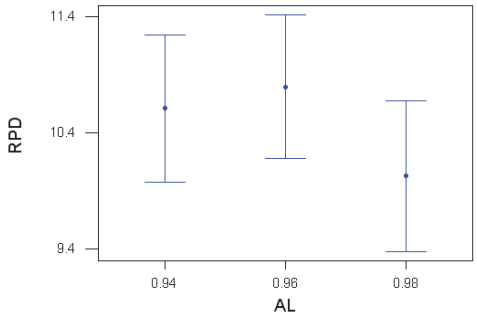

Fig. 4. Means plot and LSD intervals (at the 95% confidence level) for the different levels of parameter AL

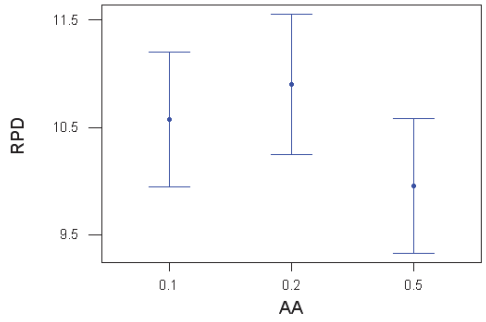

Fig. 5. Means plot and LSD intervals (at the 95% confidence level) for the different levels of parameter AA

4.2 Experimental results

In this subsection, we present the results obtained from the comparison of the algorithms mentioned in previous section. The stopping criterion used when testing all the instances with the algorithms is set to a CPU time limit fixed to mn^2 ms .

4.2.1 Policy II

As it is obvious, the RANDOM rule obtains the worst average results. The SA obtains much better results, which even exceed those obtained by MNSA and HAS which have been provided superior results than SA algorithm. Also IA has outperformed all the other algorithms.

Table 3
Average relative percentage deviation *(RPD)* for the five algorithms grouped by *n, m and STSD* for the PM policy II

Instances	STSD	IA	MNSA	HAS	SA	RADOM
20×20	U[1,24]	1.44	6.82	6.27	2.87	9.29
20×20	U[1,49]	0.13	9.58	4.66	1.45	7.52
20×20	U[1,74]	0.03	11.00	4.12	1.25	6.68
20×20	U[1,99]	0.06	12.11	3.58	1.00	5.90
20×20	U[1,124]	0.02	13.78	3.44	1.06	5.57
15×15	U[1,24]	0.91	2.46	6.69	2.67	10.15
15×15	U[1,49]	0.32	3.53	5.48	1.86	8.65
15×15	U[1,74]	0.28	4.59	4.84	1.61	7.65
15×15	U[1,99]	0.15	6.42	4.41	1.44	7.05
15×15	U[1,124]	0.10	7.53	4.02	1.26	6.40
10×10	U[1,24]	1.08	2.39	8.29	3.22	12.63
10×10	U[1,49]	0.63	3.47	7.55	2.70	11.50
10×10	U[1,74]	0.58	4.58	6.72	2.40	10.33
10×10	U[1,99]	0.22	5.68	5.92	1.87	9.25
10×10	U[1,124]	0.29	5.89	5.49	1.85	8.54
Avearage		0.42	6.65	5.43	1.90	8.47

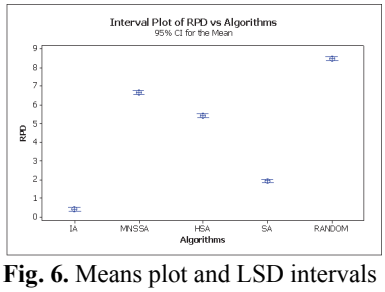

Fig. 6. Means plot and LSD intervals (at the 95% confidence level) for the type of algorithm factor in the PM policy

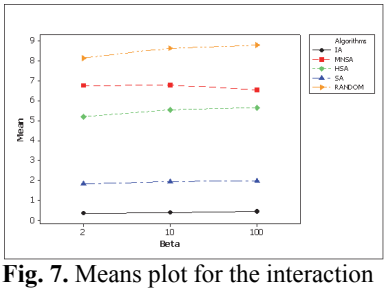

Fig. 7. Means plot for the interaction between the factors type of algorithm and *β(beta)* in the PM policy II

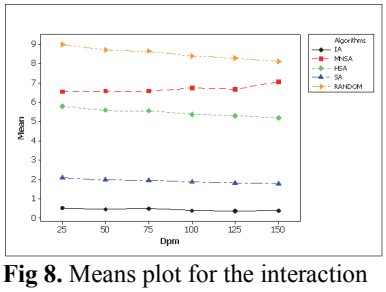

Fig 8. Means plot for the interaction between the factors type of algorithm and D_{pm} in the PM policy II

To confirm the statistical validity of the results shown in Tables 3 and to determine which is the best algorithm, a multifactor ANOVA has been performed, where the response variable is *RPD* and the factors are *n*, *SDST*, *β*, *DPM* and type of algorithm. Considering *RPD* as a response function satisfies ANOVA's three important hypotheses; normality, homogeneity of variance and independence of the residuals. The mean plot and least significant differences (LSD) intervals for the type of algorithm factor is shown in Fig. 6. As it can be seen, there are statistically significant differences between the five algorithms. The proposed IA is the best algorithm for the PM policy II.

4.2.2 Policy III

The results obtained for the five algorithms in policy III are shown in Table 4. Another ANOVA has been performed to confirm the statistical validity of the results. The LSD plot for the type of algorithm factor is shown in Fig. 9. Once again, there are statistically significant differences between the five algorithms considered, IA being the best algorithm for the PM policy III. Also we can study β and D_{PM} effects on the different algorithms when the durations of the PM operations or β are increased (Fig. 10-11).

Table 4
Average relative percentage deviation *(RPD)* for the five algorithms grouped by *n, m and STSD* for the PM policy III

Instances	STSD	IA	MNSA	HAS	SA	RADOM
20×20	U[1,24]	0.02	7.39	5.00	1.55	8.04
20×20	U[1,49]	0.03	9.26	4.60	1.38	7.46
20×20	U[1,74]	0.03	11.11	4.16	1.27	6.72
20×20	U[1,99]	0.03	12.09	3.71	1.10	5.99
20×20	U[1,124]	0.02	13.56	3.41	1.03	5.53
15×15	U[1,24]	0.92	2.34	6.56	2.55	10.05
15×15	U[1,49]	0.50	2.76	5.63	2.01	8.78
15×15	U[1,74]	0.20	4.35	4.85	1.57	7.64
15×15	U[1,99]	0.09	5.46	4.34	1.36	6.90
15×15	U[1,124]	0.07	6.82	3.97	1.22	6.36
10×10	U[1,24]	0.43	3.22	7.89	2.62	12.29
10×10	U[1,49]	0.21	4.44	7.24	2.32	11.32
10×10	U[1,74]	0.14	5.53	6.44	2.03	10.12
10×10	U[1,99]	0.15	7.22	5.98	1.85	9.42
10×10	U[1,124]	0.04	8.50	5.51	1.71	8.64
Avearage		0.19	6.94	5.29	1.70	8.35

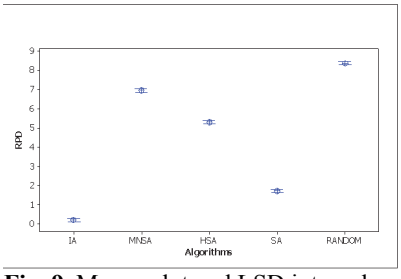

Fig. 9. Means plot and LSD intervals (at the 95% confidence level) for the type of algorithm factor in the PM policy

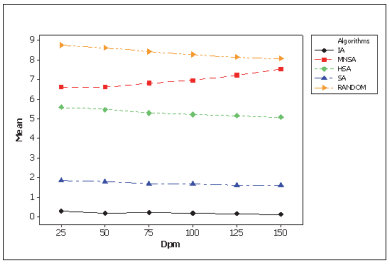

Fig. 10. Means plot for the interaction between the factors type of algorithm and Dpm in the PM policy III.

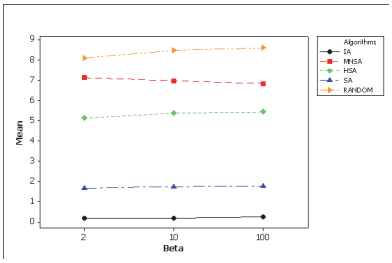

Fig. 11. Means plot for the interaction between the factors type of algorithm and β in the PM policy III

5. Conclusion

In most industrial environments, it is usually considered that the machines are available through the planning horizon, but this assumption is rarely correct in real world circumstances. Another important

assumption is to consider the setup times as a part of processing times, while set up time for a job may be depended on its immediate preceding job on the same machine, known as sequence-dependent setup times. The main idea of this paper was to consider different preventive maintenance policies on machines in the openshop problem with sequence dependent setup times using immune algorithm. Considered preventive maintenance policies have been planned for maximizing the machines availability or keeping a minimum level of reliability through the production horizon. The paper has proposed a simple criterion to schedule preventive maintenance operations to the production sequence. The objective function of this paper was to minimize makespan. In total, the proposed algorithm has been extensively compared with four adaptations of existing heuristic and meta-heuristic methods by considering preventive maintenance through data sets from benchmarks based on Taillard's instances with some adjustments. The results have shown that the proposed IA algorithm provides superior solution compared with other algorithms that obtain good quality solutions for open shop without PM constraint. Our Results also shown that the efficiency of other considered algorithms from literature respect to themselves were completely different which are available in open shop without PM constraint. The effect of different parameters on algorithms efficiency has been explored and the results showed that performance of proposed algorithm was better than other algorithms.

References

Abdi, K., Fathian, M., & Safari, E. (2012). A novel algorithm based on hybridization of artificial immune system and simulated annealing for clustering problem. *The International Journal of Advanced Manufacturing Technology*,60(5-8), 723-732.

Aggoune, R. (2004). Minimizing the makespan for the flow shop scheduling problem with availability constraints. *European Journal of Operational Resea*rch, *153*(3), 534-543.

Allahverdi, A., Ng, C. T., Cheng, T. E., & Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, *187*(3), 985-1032.

Allaoui, H., & Artiba, A. (2004). Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints. *Computers & Industrial Engineering*, *47*(4), 431-450.

Allaoui, H., & Artiba, A. (2006). Scheduling two-stage hybrid flow shop with availability constraints. *Computers & Operations Research*, *33*(5), 1399-1419.

Allaoui, H., Lamouri, S., Artiba, A., & Aghezzaf, E. (2008). Simultaneously scheduling n jobs and the preventive maintenance on the two-machine flow shop to minimize the makespan. *International Journal of Production Economics*, *112*(1), 161-167.

Anandaraman, C., Vikram, A., Sankar, M & Natarajan, R. (2012). Evolutionary approaches for scheduling a flexible manufacturing system with automated guided vehicles and robots.*International Journal of Industrial Engineering Computations* , *3*(4), 627-648.

Andresen, M., BräSel, H., MöRig, M., Tusch, J., Werner, F., & Willenius, P. (2008). Simulated annealing and genetic algorithms for minimizing mean flow time in an open shop. *Mathematical and Computer Modelling*, *48*(7), 1279-1293.

Baker, K. R. (1974). Introduction to sequence and scheduling. New York: John wiley& Sons.

Benbouzid-Sitayeb, F., Ammi, I., Varnier, C., & Zerhouni, N. (2008, April). Applying ant colony optimization for the joint production and preventive maintenance scheduling problem in the flowshop sequencing problem. In Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on (pp. 1-6). IEEE.

Birolini, A. (2007). *Reliability engineering* (Vol. 5). Berlin: Springer.

Błażewicz, J., & Formanowicz, P. (2002). Scheduling jobs in open shops with limited machine availability. *RAIRO-Operations Research*, *36*(02), 149-156.

Blum, C., & Sampels, M. (2004). An ant colony optimization algorithm for shop scheduling problems. *Journal of Mathematical Modelling and Algorithms*, *3*(3), 285-308.

Brucker, P., Hurink, J., Jurisch, B., & Wöstmann, B. (1997). A branch & bound algorithm for the open-shop problem. *Discrete Applied Mathematics*, *76*(1), 43-59.

Burton, J. S., Banerjee, A., & Sylla, C. (1989). A simulation study of sequencing and maintenance decisions in a dynamic job shop. *Computers & industrial engineering*, *17*(1), 447-452.

Cassady, C. R., & Kutanoglu, E. (2003). Minimizing job tardiness using integrated preventive maintenance planning and production scheduling. *IIE transactions*, *35*(6), 503-513.

Dorndorf, U., Pesch, E., & Phan-Huy, T. (2001). Solving the open shop scheduling problem. *Journal of Scheduling*, *4*(3), 157-174.

Duma, M., Marwala, T., Twala, B., & Nelwamondo, F. (2013). Partial imputation of unseen records to improve classification using a hybrid multi-layered artificial immune system and genetic algorithm. *Applied Soft Computing*, *13*(12), 4461-4480.

Engin, O., & Döyen, A. (2004). A new approach to solve hybrid flow shop scheduling problems by artificial immune system. *Future generation computer systems*, *20*(6), 1083-1095.

Espinouse, M. L., Formanowicz, P., & Penz, B. (2001). Complexity results and approximation algorithms for the two machine no-wait flow-shop with limited machine availability. *Journal of the Operational Research Society*, *52*(1), 116-121.

Farmer, J. D., Packard, N. H., & Perelson, A. S. (1986). The immune system, adaptation, and machine learning. *Physica D: Nonlinear Phenomena*, *22*(1), 187-204.

Forrest, S., Hoffmeyr, S.A. (2000). Engineering an immune system. *Graft, 4*(5), 5 –9.

Gao, J., & Fang, L. (2009). A novel artificial immune system for multiobjective optimization problems. *In Advances in Neural Networks–ISNN 2009*. Springer Berlin Heidelberg.

Garey, M. R., & Johnson, D. S. (1979). Computers and intractability: a guide to the theory of NP-completeness. 1979. San Francisco, LA: Freeman.

Ghodratnama, A., Rabbani, M., Tavakkoli-Moghaddam, R., & Baboli, A. (2010). Solving a single-machine scheduling problem with maintenance, job deterioration and learning effect by simulated annealing. *Journal of Manufacturing Systems*, *29*(1), 1-9.

Gonzalez, F., & Dasgupta, D. (2003). A study of artificial immune systems applied to anomaly detection (Doctoral dissertation, University of Memphis).

Guo, Y., Lim, A., Rodrigues, B., & Yu, S. (2007). Machine scheduling performance with maintenance and failure. *Mathematical and computer modelling*, *45*(9), 1067-1080.

Jabbarizadeh, F., Zandieh, M., & Talebi, D. (2009). Hybrid flexible flowshops with sequence-dependent setup times and machine availability constraints. *Computers & Industrial Engineering*, *57*(3), 949-957.

Janeway, C. A., Travers, P., Walport, M., Capra, J. D., 2000. Immunobiology The Immune System in Health and Disease (4th ed.). Artes Médicas (in Portuguese).

Kaplanoğlu, V. (2014). Multi-agent based approach for single machine scheduling with sequence-dependent setup times and machine maintenance. *Applied Soft Computing*, *23*, 165-179.

Karaboga, D., & Ozturk, C. (2011). A novel clustering approach: Artificial Bee Colony (ABC) algorithm. *Applied soft computing*, *11*(1), 652-657.

Kubzin, M. A., Potts, C. N., & Strusevich, V. A. (2009). Approximation results for flow shop scheduling problems with machine availability constraints. *Computers & Operations Research*, *36*(2), 379-390.

Li, Y. C. E., & Shaw, W. H. (1998). Simulation modeling of a dynamic job shop rescheduling with machine availability constraints. *Computers & Industrial Engineering*, *35*(1), 117-120.

Liaw, C. F. (2000). A hybrid genetic algorithm for the open shop scheduling problem. *European Journal of Operational Research*, *124*(1), 28-42.

Meeker, W. Q., & Escobar, L. A. (2014). *Statistical methods for reliability data*. John Wiley & Sons.

N. Abramson, 1963. Information Theory and Coding, McGrawHill, New York.

Naderi, B., Zandieh, M., & Ghomi, S. F. (2009). A study on integrating sequence dependent setup time flexible flow lines and preventive maintenance scheduling. *Journal of intelligent manufacturing*, *20*(6), 683-694.

Nakagawa, T. (2006). Maintenance theory of reliability. Springer Science & Business Media.

Prins, C. (2000). Competitive genetic algorithms for the open-shop scheduling problem. *Mathematical Methods of Operations Research*, *52*(3), 389-411.

Rausand, M., & Høyland, A. (2004). *System reliability theory: models, statistical methods, and applications* (Vol. 396). John Wiley & Sons.

Roshanaei, V., Esfehani, M. S., & Zandieh, M. (2010). Integrating non-preemptive open shops scheduling with sequence-dependent setup times using advanced metaheuristics. *Expert Systems with Applications*, *37*(1), 259-266.

Ruiz, R., García-Díaz, J. C., & Maroto, C. (2007). Considering scheduling and preventive maintenance in the flowshop sequencing problem. *Computers & Operations Research*, *34*(11), 3314-3330.

Safari, E., & Sadjadi, S. J. (2011). A hybrid method for flowshops scheduling with condition-based maintenance constraint and machines breakdown. *Expert Systems with Applications*, *38*(3), 2020-2029.

Safari, E., Sadjadi, S. J., & Shahanaghi, K. (2010). Scheduling flowshops with condition-based maintenance constraint to minimize expected makespan. *The International Journal of Advanced Manufacturing Technology*, *46*(5-8), 757-767.

Sarker, R., Omar, M., Hasan, S. K., & Essam, D. (2013). Hybrid Evolutionary Algorithm for job scheduling under machine maintenance. *Applied Soft Computing*, *13*(3), 1440-1447.

Sha, D. Y., & Hsu, C. Y. (2008). A new particle swarm optimization for the open shop scheduling problem. *Computers & Operations Research*, *35*(10), 3243-3261.

Sortrakul, N., Nachtmann, H. L., & Cassady, C. R. (2005). Genetic algorithms for integrated preventive maintenance planning and production scheduling for a single machine. *Computers in Industry*, *56*(2), 161-168.

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research*, *64*(2), 278-285.

Ushakov, I. A. (Ed.). (1994). *Handbook of reliability engineering*. John Wiley & Sons.

Vahedi Nouri, B., Fattahi, P., & Ramezanian, R. (2013). Hybrid firefly-simulated annealing algorithm for the flow shop problem with learning effects and flexible maintenance activities. *International Journal of Production Research*, *51*(12), 3501-3515.

Yulan, J., Zuhua, J., & Wenrui, H. (2008). Multi-objective integrated optimization research on preventive maintenance planning and production scheduling for a single machine. The *International Journal of Advanced Manufacturing Technology*, *39*(9-10), 954-964.

Zammori, F., Braglia, M., & Castellano, D. (2014). Harmony search algorithm for single-machine scheduling problem with planned maintenance. *Computers & Industrial Engineerin*g, *76*, 333-346.

Zhou, B. H., Jiang, S. Y., Wang, S. J., Wu, B., & Xi, L. F. (2007). Integrated production and preventive maintenance scheduling algorithm for flow shops. *Journal of Dalian Maritime University*, *33*(3), 32-35.