International Journal of Industrial Engineering & Production Research September 2026, Vol. 37, No. 3: 134-144
DOI: 10.22068/ijiepr.37.3.2645

RESEARCH PAPER

A Genetic-Firefly Algorithm for a Sequence-Dependent Setup Times Job Shop Scheduling Problem Considering Learning Effects and Availability Constraints

Seyed Hamed Mousavipour[1,*], Hiwa Farughi[2]

Received 07 December 2025; Revised 25 May 2026; Accepted 01 July 2026;
© Iran University of Science and Technology 2026

ABSTRACT
Humans are inherently learning beings. Due to learning effects, when an operation is performed repeatedly, workers gain experience, leading to a reduction in job processing times. In contemporary scheduling literature, both learning effects and restricted access to machines caused by maintenance activities are recognized as critical factors that have attracted extensive research from multiple perspectives. Despite this, relatively few studies have systematically examined the learning phenomenon within the context of the Job Shop Scheduling Problem (JSSP). Most prior research assumes that machines are continuously available, that set-up times are incorporated into processing times, and that transportation times are negligible.
This study introduces a novel JSSP framework incorporating Sequence-Dependent Set-up Times (SDSTs), classical position-based learning effects, flexible maintenance schedules, and transportation times. A corresponding mathematical model is formulated, and numerical instances of three different scales are generated. The model is solved exactly for small-sized instances using CPLEX within GAMS. To address medium- and large-sized instances, metaheuristic algorithms including Ant Colony Optimization for continuous domains (ACOR), Invasive Weed Optimization (IWO), and a hybrid Genetic–Firefly algorithm (GA-FA) are employed.

KEYWORDS: *Job shop scheduling problem; Sequence-dependent set-up times; Learning effects; Flexible maintenance; Transportation times; Genetic-firefly algorithm.*

1. Introduction

Scheduling is a critical function that enables companies, administrative systems, and manufacturing industries to operate efficiently. It involves the allocation of finite resources such as human labor, tools, and machinery to perform a set of jobs while optimizing one or more objectives. Effective scheduling can help manufacturing systems reduce costs, meet customer deadlines reliably, and transform inefficient and costly processes into streamlined and economical operations. Introduced by Muth and Thompson [1], the Job Shop Scheduling Problem (JSSP) consists of a set of n jobs, each comprising a sequence of operations that must be processed on m machines. The job routes vary, and at most one operation can be executed on a machine at any given time.
A review of the literature shows that most studies incorporate set-up times within processing times, with only a few explicitly modeling set-up times as sequence-dependent. Sequence-dependent set-up times (SDSTs) can be categorized as either anticipatory or non-anticipatory. In the anticipatory type, set-ups occur only when both the job and the machine are available, whereas in the non-anticipatory type, set-ups can proceed even if the job is unavailable, provided the machine is ready. In this study, anticipatory set-up times are considered.
Traditionally, job processing times are assumed to be known and fixed. However, recent empirical evidence across various industries indicates that processing times decrease as workers gain experience and refine their skills over repeated operations, a phenomenon referred to as learning effects [2].
Machine availability constraints have also become an important consideration in scheduling research. While many studies assume unrestricted access to machines, in practice, maintenance activities, equipment failures, and other factors often limit availability. Two main types of deterministic availability constraints are recognized in the literature: fixed preventive maintenance, in which start times and durations are predetermined, and flexible maintenance, in which start times fall

* *Corresponding author: Seyed Hamed Mousavipour*
h.mousavipour@yu.ac.ir

1. *Assistant professor, Department of computer engineering, Yasouj University, Iran*
2. *Ph.D, Department of industrial engineering, University of Kurdistan, Pasdaran Boulevard, Sanandaj, Iran*

within a defined time window.

In job shop environments, jobs are transferred between machines, and operations may experience waiting times. Transportation times are typically classified as either job-dependent or job-independent: job-independent times depend solely on the distance between consecutive machines, whereas job-dependent times also consider the specific job being transported. Transporter systems may be limited (single) or unlimited (multiple) [3]. Notably, when anticipatory SDSTs are employed, set-up and transportation can overlap, allowing job processing to commence once both are completed.

This paper introduces a novel SDSTs-JSSP framework that incorporates real-world considerations, including learning effects, multiple flexible availability constraints, and transportation times. The remainder of the paper is structured as follows: Section 2 reviews the literature; Section 3 presents a mathematical model based on learning effects, anticipatory SDSTs, flexible maintenance activities, and job-dependent transportation times; Section 4 outlines the solution techniques; Sections 5 and 6 detail data generation and comparison metrics; Section 7 evaluates algorithmic performance; and Section 8 concludes the study.

2. Literature Review

Job Shop Scheduling Problems with set-up times arise in numerous production environments, including aerospace, fabrication, printing, and semiconductor industries [4]. Allahverdi and Soroush [5] emphasize that scheduling heavily depends on the preparation of related facilities, making set-up times and costs critical. Sharma and Jain [4] provide a comprehensive review of JSSP with set-up times. Few studies address sequence-dependent set-up times (SDSTs) in conjunction with availability constraints. Naderi et al. [6] examined SDSTs alongside preventive maintenance in a job shop and proposed two metaheuristic algorithms without presenting a formal mathematical model. Biskup [7] was among the first to investigate learning effects in scheduling, demonstrating that processing times decrease as operations are repeated. He categorized learning effects into position-based and cumulative processing time-based types. In the position-based type, processing times depend on a job's sequence position, whereas in the cumulative type, they depend on previously completed jobs. Reviews of learning effects in scheduling are provided by Azzouz et al. [2] and Biskup [8]. Other studies, including Vahedi-Nouri et al. [9], Amirian and Sahraean [10], Behnamian and Zandieh [11], and Mousavi et al. [12], examine learning effects in flow shop environments. In recent years, many scholars, in order to bring research closer to real-world problems, have considered these effects in various types of scheduling problems. A concise guide to scheduling has been proposed by Pei et al. [13]. In this study, a comprehensive review and critical hints for future research have been provided. A novel classification scheme is designed based on four levels of different domains, i.e., learning effects, processing ways, processing time functions, and manufacturing environments. Hu, C et al. [14] in a paper propose a mathematical model for the time–dependent setup times group scheduling problem, where job processing times are simultaneously affected by both deterioration and learning effects. A two–machine no–wait proportionate flow shop scheduling problem with consideration of position-based learning effects and resource allocation is investigated to find the optimum amount of some objectives by Sun, Y. et al. [15]. Birgin, E.G. et al [16] consider the flexible job shop scheduling problem to minimize energy consumption. To bring the problem closer to real-world problems, two relevant factors, nonlinear routes and position-based learning effects, have been proposed.

Mousavipour et al. [17, 18] surveyed JSSP with learning effects for the first time, incorporating flexible maintenance and job-dependent transportation times, and proposed a MILP model under these assumptions. Additionally, they studied position-based learning in SDSTs-JSSP, considering multiple fixed maintenance periods. Tayebi Araghi et al. [19] addressed learning effects on set-up times in flexible job shop scheduling. Ma et al. [20] and Schmidt [21] reviewed scheduling problems considering machine availability constraints. They consider fixed and flexible preventive maintenance activities as an availability constraint in single-machine scheduling problems.

Tavakkoli-Moghaddam et al. [22] investigated SDSTs-JSSP with fixed preventive maintenance to minimize tardiness, proposing hybrid metaheuristic algorithms without formal mathematical modeling. This study considers flexible availability constraints similar to those of Vahedi-Nouri et al. [23], where a predetermined number of maintenance activities are scheduled for each machine and are considered delayed if completed after their earliest possible finish time.

3. Model Description

In the JSSP, a set of n jobs must be processed by a set of m machines. Containing L operations, each job J_j has a specific predefined sequence. There are k positions on each machine M_i. Only one job can be processed at a time by each machine, and only one machine can process each job at a time. The jobs are not allowed to be preempted, in the sense that the jobs are processed without interruption. In this section, the job shop problem is detailed with the assumptions made above. All the jobs are available at time zero, and there are no buffer constraints between the machines. Each job J_j has a predetermined normal processing time of p_{ij} on machine M_i. The jobs are affected by the classic position-based learning effects. The real processing time of the job J_j on machine M_i depends on the position of the job in sequence k and the learning index $\alpha_i \leq 0$, which is obtained from the equation $p_{ijk} = p_{ij}k^{\alpha_i}$.

The set-up times have been assumed to be anticipatory and sequence-dependent, and $S_{jij'}$ represents the time required for the preparation of the machine M_i when job J_j is placed before $j_{j'}$, It is also assumed that S0ij= 0. By the way, in the literature, it has been assumed that sequence-dependent setup times satisfy a triangular inequality, so that $S_{jij'} + S_{j'ij''} \geq S_{jij''}$ When this condition seems a rational condition according to the practical applications. Transportation times $tp_{iji'}$ representing the time needed for transferring a job J_j from machine M_i to machine $M_{i'}$. Along the same lines as the set-up times, the transportation times have to meet the triangle inequality, i.e. $tp_{iji'} + tp_{i'ji''} \geq tp_{iji''}$; This is, of course, not a real restriction [3].

Machine M_i requires r maintenance activities, such that the rth maintenance operation PM_{ir} is completed in a time window of $[EF_{ir}, LF_{ir}]$ where r= 1, 2, …, r. C_{ijk} is the completion time of the job M_i if processed on the kth position of the machine J_j, and C_j is the completion time of the jth job. The binary variable X_{ijk} is one if job J_j is placed in the kth position of the machine M_i, and is 0 otherwise. The variable Zijkj'k + 1 = X_{ijk}. Xij'k + 1 is one if the job J_j is placed in the kth position of the machine M_i, and job $j_{j'}$ is placed on the (k+1)th position of that machine. The assumptions, notations, and decision variables that are utilized in the present model are mentioned below:

3.1. Assumptions

- There is access to all machines and jobs at time zero.
- The jobs cannot be preempted.
- A job is processed on a machine just once.
- Processing times are decreased by the effects of position-based learning.
- Sequence -dependent set-up times are considered.
- Each machine is serviced by multiple flexible maintenance activities.
- Intra-machine job-dependent transportation times have been considered.
- There is no limitation on transporters.

3.2. Notations

Indices:

i: Index for machines
j: Index for jobs
k: Index for positions
r: Index for maintenance activities
L: Index for operations

Parameters and scalars:

n: Number of jobs
m: Number of machines
V: A big positive number
r: Required maintenance activities for each machine
P_{ij}: Normal processing time of a job J_j on machine M_i
$P_{ijk} = p_{ij}k^{\alpha_i}$: Processing time of job J_j on the kth position of machine M_i
α_i: learning index on a machine M_i ($\alpha_i \leq 0$)
PM_{ir}: r^{th} maintenance task on machine M_i
EF_{ir}: Earliest possible finish time of maintenance activity PM_{ir}
LF_{ir}: Latest possible finish time of maintenance activity PM_{ir}
t_{ir}: Duration time of PM_{ir}
q_{ijl}: A binary parameter that is 1 if the lth operation of the jth job is performed on machine M_i, and is 0 otherwise
$tp_{iji'}$: Required time for moving job J_j from machine M_i to machine $M_{i'}$
$S_{jij'}$: Required time for the preparation of machine M_i when job J_j is placed before $j_{j'}$

3.3. Decision variables

X_{ijk}: Is 1 if job J_j is processed on the kth position of machine M_i, and is 0 otherwise, refers to a binary variable

C_{ijk}: Completion time of job J_j if scheduled on the kth position of the machine M_i and 0 otherwise, refers to a binary variable

ZM_{ijr}: Is 1 if maintenance activity PM_{ir} is performed after job j is processed, and is 0 otherwise

FM_{ir}: Finish time of maintenance activity PM_{ir}

$Z_{ijkj'k+1} = X_{ijk} . X_{ij'k+1}$ is equal to 1 if the job J_j is processed in the kth position of the machine M_i and job $J_{j'}$ is processed in the position k+1 of this machine.

3.4. Mathematical model

Based on above-mentioned descriptions, the mathematical model is as follow:

$$\text{Min } Z = \sum_{j=1}^{n} C_j + \sum_{i=1}^{m}\sum_{r=1}^{r}(FM_{ir} - EM_{ir}) \quad (1)$$

S.t

$$\sum_k X_{ijk} = 1 \quad \forall i = 1,\dots,m \quad j = 1,\dots,n \quad (2)$$

$$\sum_j X_{ijk} = 1 \quad \forall i = 1,\dots,m \quad k = 1,\dots,n \quad (3)$$

$$\sum_j C_{ijk} + \sum_j X_{ij\,k+1} \times P_{ij\,k+1} + \sum_j \sum_{j'} Z_{ijkj'k+1} \times S_{jij'} \le \sum_j C_{ijk+1} \quad \forall i = 1,\dots,m\, k = 1,\dots,n-1 \quad (4)$$

$$\sum_i q_{ijl} \times C_{ijk} + \sum_{i'} q_{i'j\,l+1} \times X_{i'jk} \times P_{i'jk} + \sum_i \sum_{i'} q_{ijl} \times q_{i'j\,l+1} \times tp_{iji'} \le V \times \left(1 - \sum_i q_{ijl} \times X_{ijk}\right) + V \times \left(1 - \sum_j q_{ij\,l+1} \times X_{ijk}\right) + \sum_{i'} q_{ij\,l+1} \times C_{i'jk}$$

$$\forall j = 1,\dots,n \quad L = 1\dots, m-1 \quad K = 1\dots, n \quad i', i = 1,\dots,m \quad (5)$$

$$\sum_k C_{ijk} - \sum_k X_{ijk} \times P_{ijk} - FM_{ir} + (ZM_{ijr} \times v) \ge 0$$

$$\forall i = 1,\dots,m \quad j = 1,\dots,n \quad r = 1,\dots,r \quad (6)$$

$$FM_{ir} - t_{ir} + V\left(1 - ZM_{ijr}\right) \ge \sum_k C_{ijk}$$

$$\forall i = 1,\dots,m \quad j = 1,\dots,n \quad r = 1,\dots,r \quad (7)$$

$$EF_{ir} \le FM_{ir} \quad \forall i = 1,\dots,m \quad r = 1,\dots,r \quad (8)$$

$$FM_{ir} \le LF_{ir} \quad \forall i = 1,\dots,m \quad r = 1,\dots,r \quad (9)$$

$$Z_{ijkj'k+1} \le X_{i,j,k} \quad \forall i = 1,\dots,m \quad j, j' = 1,\dots,n \quad k = 1,\dots,n-1 \quad (10)$$

$$Z_{ijkj'k+1} \le X_{i'jk+1} \quad \forall i = 1,\dots,m \quad j, j' = 1,\dots,n \quad k = 1,\dots,n-1 \quad (11)$$

$$Z_{ijkj'k+1} \ge \left(X_{ijk} + X_{i'jk+1}\right) - 1 \quad \forall i = 1,\dots,m \quad j, j' = 1,\dots,n \quad k = 1,\dots,n-1 \quad (12)$$

$$C_{ijk} \le V \times X_{ijk} \quad \forall i = 1,\dots,m \quad j = 1,\dots,n \quad k = 1,\dots,n \quad (13)$$

$$\sum_k C_{[1]jk} \ge \sum_k X_{[1]}jk \times P_{[1]jk} \quad \forall j = 1,\dots,n \quad (14)$$

$$C_j \ge \sum_k C_{[\text{Last machine}]jk} \quad \forall j = 1,\dots,n \quad (15)$$

$$\forall j = 1,\dots,n \quad C_j \ge 0 \quad (16)$$

$$X_{ijk}, Z_{ijkj'k+1}, ZM_{ijr} \in \{0,1\} \quad \forall i = 1,\dots,m \quad j, j' = 1\dots,n \quad k = 1,\dots,n \quad r = 1\dots,r \quad (17)$$

The first equation of the model defines the objective function, which aims to minimize the total completion times of jobs as well as delays associated with maintenance activities. Constraints (1) and (2) ensure that each job is assigned to a single position and that each position accommodates only one job. Constraint (3) incorporates position-based learning effects to determine the actual processing times for each job on each machine. Constraint (4) enforces the precedence relationships among operations, taking transportation times into account and ensuring that all operations of a job are executed in the prescribed sequence. Constraints (5) through (10) calculate the completion times of maintenance activities on each machine, considering both the real job processing times and the time windows of flexible maintenance tasks. Constraints (10) to (12) implement the linearization of product terms in the model. Constraint (13) computes the completion time of a job if it is scheduled in the k-th position on a machine; otherwise, it is zero. Constraints (14) and (15) determine job completion times, while Constraints (16) and (17) define the non-negativity and binary conditions of the decision variables.

4. Proposed Solution Methods

The Job Shop Scheduling Problem (JSSP) is well-known to be strongly NP-hard [24]. In this study, the proposed MILP model is solved exactly using the CPLEX solver within the GAMS environment for small-scale instances. To obtain near-optimal solutions for medium- and large-scale instances within reasonable computational times, three metaheuristic algorithms are employed.

4.1. Solution representation

To present the solution, a string as long as "the number of machines" × "the number of jobs" has been applied. At the beginning of the algorithm for each machine, an initial solution is generated randomly. Each string contains number of machines, assigned values for each section are ordained in ascending order. This order specifies the sequence of jobs processing on each machine. Fig. 1 demonstrates decoded scheme

In the subsequent step, the duration and scheduling of maintenance activities on each machine are represented using a solution structure with dimensions equal to “the number of machines × the number of maintenance activities × the number of positions”. In the proposed approach, the optimal position k is determined for each machine and maintenance task, thereby specifying the assignment of maintenance operations. Figure 2 illustrates how maintenance activities are allocated across machines.

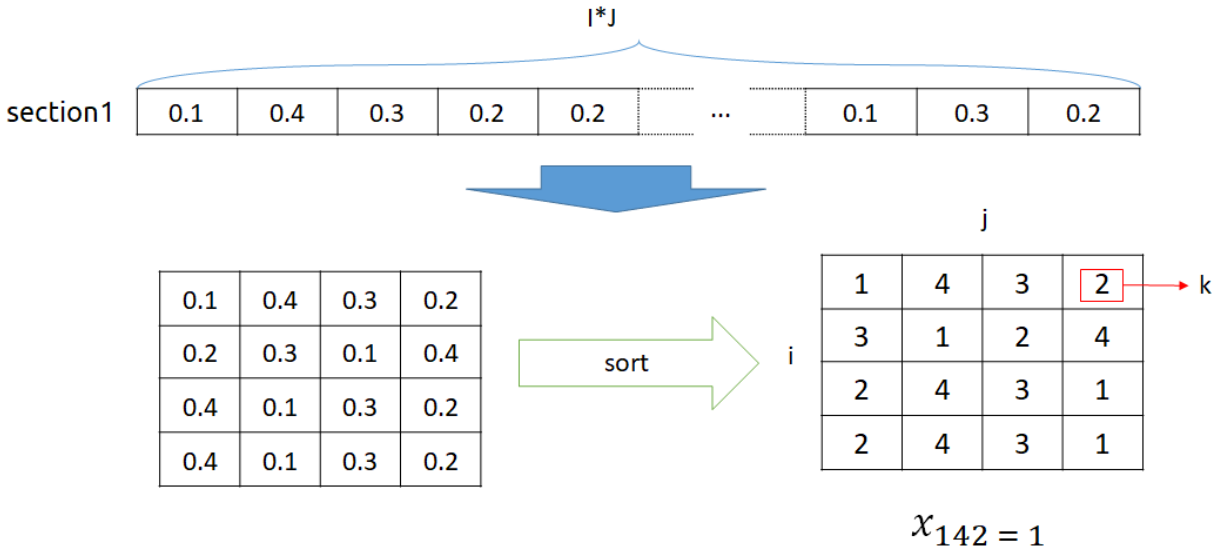

Fig.1. Decoded scheme

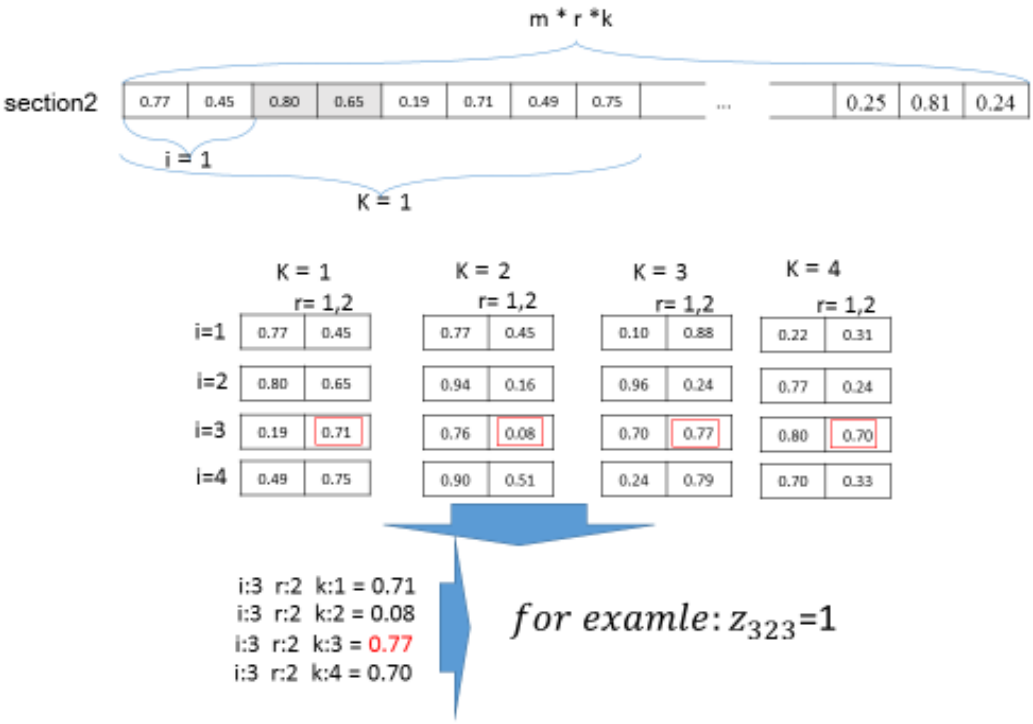

Fig.2. Determining the position of maintenance activity

For each feasible solution, the objective function is evaluated based on job positions on machines, set-up times, maintenance durations, and transportation times, and the solution yielding the best objective value is selected.

4.2. Metaheuristic algorithms

As we mentioned before, to tackle medium and large-sized instances, we applied three metaheuristic algorithms: Ant Colony Optimization for continuous domains (ACOR) [25], Invasive Weed Optimization (IWO) [26] and a hybrid algorithm based on the Genetic and Firefly algorithms (GA-FA).

The Ant Colony Algorithm was presented by Colorni et al. [27], inspired by the foraging behavior of Ants. In ACO, artificial ants traverse graphs to construct solutions. This solution-construction process is stochastic and guided by a pheromone-based model. The algorithm operates within a discrete framework, where each decision variable is divided into a finite set of states within a predefined domain. This discretization constrains

the algorithm and may reduce its solution precision Ant Colony Optimization for continuous domains applies a continuous probability distribution function to turn space of decision variables in to continuous space.

Based on the concept of grass colonies, IWO was introduced as a stochastic mathematical optimization algorithm. grasses are a class of plants that pose significant challenges to agriculture due to their aggressive nature and resilience. They adapt quickly and effectively to environmental changes, making the algorithm inspired by their characteristics highly flexible. The grass Optimization Algorithm emulates the strength, adaptability, and stochastic behavior observed in grass colonies.

4.2.1. Genetic–firefly algorithm

The Genetic Algorithm (GA) is a widely used evolutionary optimization method [28]. It generates an initial population according to specific rules and evolves new generations through crossover and mutation operations, iterating until termination criteria are met.

Firefly Algorithm (FA) was introduced by Yang [29] and inspired by the flashing behavior of fireflies. Silkworm is a winged insect or cockroach that generates light or blinks during the night. The light does not contain an infrared or ultraviolet frequency and is generated chemically by the lower abdomen, known as bioluminescence. A silkworm generates light to attract mates or prey. The flashlight is also used as a protective warning mechanism for reminding silkworms of potential hunters. The algorithm begins to solve the problem with a random population, and in each iteration, the GA operators are first applied to the problem, and FA then begins with the current population. Once the FA operators finish their operation, the population obtained from the GA and FA operators is placed in a pool of solutions, and as many solutions as the size of the initial population are then selected from among the best solutions in the set, and the algorithm is iterated until the termination conditions are met. The Pseudo code of GA-FA is as follows:

5. Parameter Generation

Since this model is original and we have no access to any benchmark, we generated problems randomly in three scales: small, medium, and large. If a, represents "the number of machines × the number of jobs × the number of maintenance operations on each machine" if a< 30 instance is small. If 30< a<60 problem is considered as a medium, otherwise is large.

```
GA-FA
begin
Choose an initial population
repeat
Evaluate individual fitness for a certain proportion of the population
Select pairs of highest-ranking individuals for reproduction
Apply the crossover operator
Apply the mutation operator
repeat
        for i= 1 to n (all of the n fireflies)
                for j=1 to n (all of the n fireflies)
                        if (Ij > Ii), move firefly i towards firefly j|: end if
                                Attractiveness varies with distance
Evaluate the new solutions, and update light intensely
                End for j
        End for i
Rank the fireflies
End While
until FA pop is reached
until termination condition
end
```

Fig.3. Pseudo code of the GA-FA algorithm

Tab.1. Parameter generation for random instances

Parameter	Notation	Value
N	Number of jobs	{2…10}
M	Number of machines	{3…12}
R	Number of maintenances on each machine	2
α_i	Learning index of machine M_i	Uniform distribution (-0.9, -0.1)
t_{ir}	Normal maintenance execution time	Uniform distribution (5, 20)
EF_{ir}	Earliest possible finish time of maintenance activity PM_{ir}	$K(\sum_{j=1}^{n} P_{ij})/(k+1) + 25(i-1)$
LF_{ir}	Latest possible finish time of maintenance activity PM_{ir}	$K(\sum_{j=1}^{n} P_{ij})/(k+1) + 25(i)$
$tp_{iji'}$	Transportation times	Uniform distribution (1, 30)
$S_{jij'}$	Set-up times	Uniform distribution (5, 40)
P_{ij}	Processing time of job J_j on machine M_i	Uniform distribution (10, 100)

6. Survey on the Solutions

Three criteria have been used for comparing the solution methods. Relative Percentage Deviation (RPD). RPD of each solution is calculated by Eq. (15), where Alg_{sol} is the value of the objective function gained by solving an instance using the considered algorithm, and Min_{sol} is the minimum objective value obtained by solving that instance using all the all-proposed algorithms. Furthermore, Improvement (Imp) as the second metric indicates the amounts of improvement occurred in the initial solution, obtained by Eq. (16), where $Alg_{initialsol}$ is the objective value of the initial solution of the algorithm, and $Alg_{initialsol}$ is the objective value of its final solution. CPU time of each algorithm is the third metric for comparison the efficiency of each method.

$$RPD = \frac{Alg_{sol} - Min_{sol}}{Min_{sol}} \times 100 \quad (18)$$

$$Imp = \frac{Alg_{initialsol} - Alg_{finalsol}}{Alg_{initialsol}} \times 100 \quad (19)$$

As Table 2 shows, instances have been solved by considering three different learning rates. Results reveal that the more learning rate is applied, the less objective function is obtained. On the other hand, the values of RPD indicate that all three metaheuristic algorithms can gain optimal solutions for all small-sized instances which confirm the validity of these algorithms.

Table 3 shows the values of two performance metrics i.e., Imp and RPD besides CPU times of solution methods for three medium-sized instances and four large-sized instances. Fig. 3 shows that the most improvement in the initial solution is 0.379 which has been made by GA-FA. As Fig. 4 reveals, given the mean RPD values, GA-FA enjoys the best performance in comparison to IWO and ACOR, it means that GA-FA can find a better solution rather than other two algorithms. However, Fig. 5 demonstrates that the highest amount of CPU time belongs to GA-FA, IWO and ACOR respectively. It is necessary to mention that parameters of these applied algorithms have been tuned by trial-and-error method.

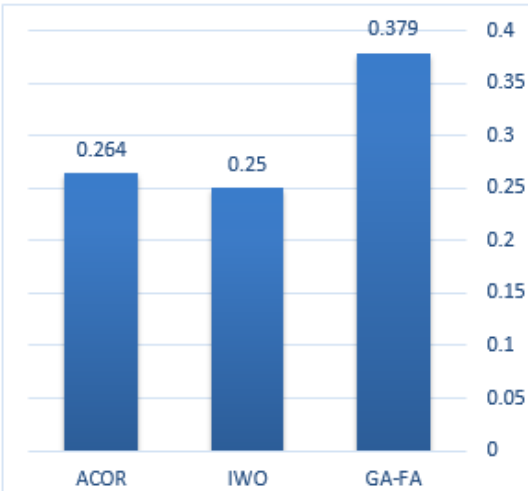

Fig.4. Mean values of Imp for metaheuristic solution methods

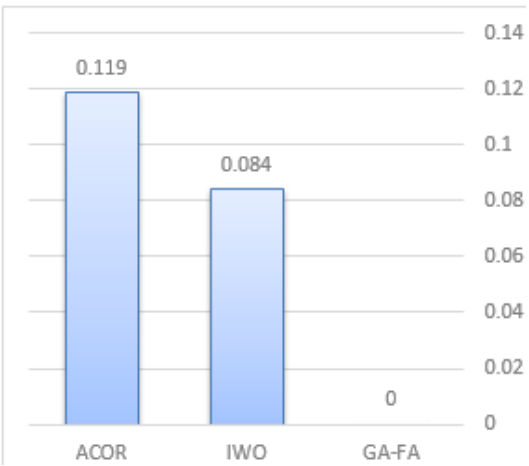

Fig.5. Mean values of RPD for metaheuristic solution method

Tab.2. Computational results for small-sized instances

Machine Job maintenance	α	CPU time	RPD	Imp	Objective function values
Exact MIP	70	0.258	0	———	896.02
	80	0.345	0	———	882.55
	90	0.448	0	———	852.81
3 ×3 × 2 IWO	70	9.852	0	0.98	896.02
	80	11.88	0	0.175	882.55
	90	12.35	0	0.92	852.81
GA-FA	70	84.88	0	0.20	896.02
	80	81.97	0	0.30	882.55
	90	83.48	0	0.07	852.81
ACOR	70	9.97	0	0.91	896.02
	80	9.18	0	0.92	882.55
	90	9.13	0	0.92	852.81
Exact MIP	70	2.88	0	———	788.12
	80	3.41	0	———	760.49
	90	2.336	0	———	744.03
IWO	70	9.18	0	0.93	788.12
	80	9.30	0	0.93	760.49
	90	11.27	0	0.93	744.03
3 × 4 × 2 GA-FA	70	60.94	0	0.98	788.12
	80	57.79	0	0.18	760.49
	90	57.56	0	0.40	744.03
ACOR	70	10.53	0	0.93	788.12
	80	11.01	0	0.93	760.49
	90	10.61	0	0.93	744.03
Exact MIP	70	10.32	0	———	1040.79
	80	11.05	0	———	1000.32
	90	8.00	0	———	980.15
IWO	70	45.47	0	0.91	1040.79
	80	44.79	0	0.92	1000.32
	90	37.86	0	0.92	980.15
4 × 4 × 2 GA-FA	70	92.73	0	0.92	1040.72
	80	69.54	0	0.92	1000.32
	90	61.34	0	0.92	980.15
ACOR	70	12.95	0	0.92	1040.79
	80	12.47	0	0.92	1000.32
	90	13.33	0	0.92	980.15
Exact MIP	70	557	0	———	1159.05
	80	1006	0	———	1145.18
	90	987	0	———	1107.77
3 ×5 × 2 IWO	70	84.60	0	0.33	1159.05
	80	96.78	0	0.90	1145.18
	90	103.53	0	0.31	1107.77
GA-FA	70	68.68	0	0.90	1159.05
	80	54.37	0	0.27	1145.18
	90	56.112	0	0.9	1107.77
ACOR	70	42.17	0	0.90	1159.05
	80	79.9	0	0.90	1145.18
	90	41.42	0	0.90	1107.77

Tab.3. Computational results for medium and large-sized instances

	GA-FA			IWO			ACOR		
Representation	**CPU time (s)**	**Imp**	**RPD**	**CPU time (s)**	**Imp**	**RPD**	**CPU time (s)**	**Imp**	**RPD**
4 × 5 × 2	136.45	0.16	0	36.66	0.15	0	61.25	0.16	0.001
5 × 5 × 2	154.48	0.186	0	60.58	0.235	0	69.84	0.198	0.047
5 × 6 × 2	186.36	0.384	0	86.58	0.313	0.003	156.92	0.355	0.019
6 × 7 × 2	218.49	0.324	0	248.12	0.236	0.01	167.56	0.231	0.119
8 × 9 × 2	381.65	0.504	0	353	0.055	0.252	454.32	0.277	0.199
10 × 9 × 2	728.23	0.578	0	631.87	0.392	0.221	684.67	0.342	0.276
12 × 10 × 2	881.39	0.521	0	765.69	0.373	0.102	883.12	0.29	0.291
Mean	366.292	0.379	0	311.78	0.250	0.084	353.91	0.264	0.119
* The metaheuristics values are the best values in 5 runs									

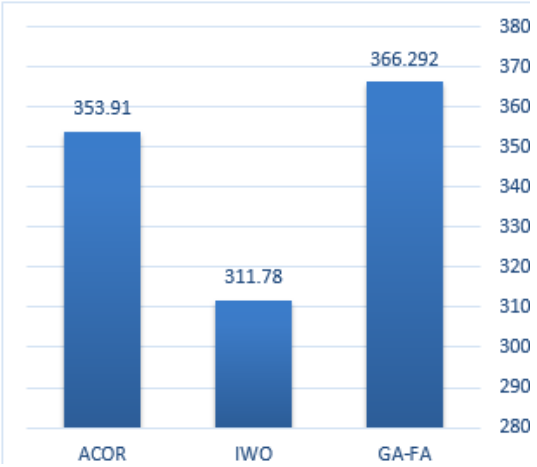

Fig.6. Mean values of CPU times for metaheuristic solution methods

6.1. Comparative statistical analysis

The tables below report the p-values obtained from pairwise comparisons using paired t-tests at a 95% confidence level. Values below 0.05 are highlighted in bold red, indicating that the null hypothesis of equal mean differences is rejected. These results indicate a statistically significant difference between the GA-FA and ACOR methods across both performance metrics. Furthermore, boxplot analyses reveal that the GA-FA algorithm demonstrates superior performance relative to ACOR with respect to the RPD metric.

Tab.4. statical analysis of IMP for methods

ACOR	IWO	GA-FA	
0.037	0.079	-	GA-FA
0.722	-	-	IWO
-	-	-	ACOR

Tab.5. Statical analysis of RPD for methods

ACOR	IWO	GA-FA	
0.025	0.091	-	GA-FA
0.130	-	-	IWO
-	-	-	ACOR

7. Conclusion

Incorporating assumptions such as anticipatory sequence-dependent set-up times (SDSTs), position-based learning effects, job-dependent transportation times, and flexible maintenance activities renders the Job Shop Scheduling Problem (JSSP) more representative of real-world manufacturing systems. In this study, a novel MILP model is proposed based on these assumptions, aiming to minimize the total completion times of jobs and delays associated with maintenance activities. To evaluate the model, instances of three different sizes were generated randomly. The model was solved exactly for small-scale instances using GAMS software. Classic position-based learning effects were incorporated, and numerical examples demonstrate their positive impact on the objective function.

Given the computational complexity of medium- and large-scale instances, a hybrid metaheuristic algorithm combining Genetic Algorithm and Firefly Algorithm (GA-FA), together with Invasive Weed Optimization (IWO) and Ant Colony Optimization (ACO), was applied. These algorithms successfully identified optimal solutions for small instances and were further employed for medium- and large-scale problems. Performance was assessed using RPD, Imp, and CPU time metrics. Although GA-FA required the highest computational time compared to IWO and ACO, it consistently achieved the best solution quality and the greatest improvement over initial solutions.

Future research may explore other types of learning effects, including induced learning, random failures, or non-anticipatory SDSTs, which represent promising directions for further investigation.

References

[1]. Muth, J.F., Thompson, G.L., Industrial Scheduling, Englewood Cliffs, New Jersey, 1963.

[2]. Azzouz, A., Ennigrou, M., Ben Said, L., "Scheduling problems under learning effects: classification and cartography", International Journal of Production Research, Vol. 56, No. 4, 2018, pp. 1642–1661. doi: 10.1080/00207543.2017.1355576.

[3]. Ahmadizar, F., Shahmaleki, P., "Group-shop scheduling with sequence-dependent set-up and transportation times", Applied Mathematical Modelling, Vol. 38, No. 21, 2014, pp. 5080–5091. doi: 10.1016/j.apm.2014.03.035.

[4]. Sharma, P., Jain, A., "A review on job shop scheduling with setup times", Journal of Engineering Manufacture, Vol. 230, No. 3, 2015, pp. 517–533. doi: 10.1177/0954405414560617.

[5]. Allahverdi, A., Soroush, H., "The significance of reducing set-up times/set-up costs", European Journal of Operational Research, Vol. 187, No. 3, 2008, pp. 978–984. doi: 10.1016/j.ejor.2006.09.010.

[6]. Naderi, B., Fatemi Ghomi, S.M.T., Aminnayeri, M.A., "A high-performing metaheuristic for job shop scheduling with sequence-dependent set-up times", Applied Soft Computing, Vol. 10, No. 3, 2010, pp. 703–710. doi: 10.1016/j.asoc.2009.08.039.

[7]. Biskup, D., "Single-machine scheduling with learning considerations", European Journal of Operational Research, Vol. 115, No. 1, 1999, pp. 173–178. doi: 10.1016/S0377-2217(98)00246-X.

[8]. Biskup, D., "A state-of-the-art review on scheduling with learning effect", European Journal of Operational Research, Vol. 188, No. 2, 2008, pp. 315–329. doi: 10.1016/j.ejor.2007.05.040.

[9]. Vahedi-Nouri, B., Fattahi, P., Tavakkoli-Moghaddam, R., Ramezanian, R., "A general flow shop scheduling problem with consideration of position-based learning effect and multiple availability constraints", International Journal of Advanced Manufacturing Technology, Vol. 73, No. 5, 2014, pp. 601–611. doi: 10.1007/s00170-014-5841-4.

[10]. Amirian, H., Sahraeian, R., "Augmented ε-constraint method in multi-objective flow shop problem with past sequence set-up times and a modified learning effect", International Journal of Production Research, Vol. 53, No. 19, 2015, pp. 1–15. doi: 10.1080/00207543.2015.1033033.

[11]. Behnamian, J., Zandieh, M., "Earliness and tardiness minimizing on a realistic hybrid flowshop scheduling with learning effect by advanced metaheuristic", Arab Journal of Science and Engineering, Vol. 38, 2013, pp. 1229–1242. doi: 10.1007/s13369-012-0347-6.

[12]. Mousavi, S.M., Mahdavi, L., Rezaeian, J., Zandieh, M., "Bi-objective scheduling for the re-entrant hybrid flow shop with learning effect and setup times", Scientia Iranica E, Vol. 25, No. 4, 2018, pp. 2233–2253. doi: 10.24200/SCI.2017.4451.

[13]. Pei, J., Zhou, Y., Yan, P., & Pardalos, P. M. (2023). A concise guide to scheduling with learning and deteriorating effects. International Journal of Production Research, 61(6), 2010–2031. https://doi.org/10.1080/00207543.2022.2049911.

[14]. Hu, C., Zhai, Y., Lu, S. et al. Parallel machine group scheduling with time-dependent setup time and combined effects of deterioration and learning. Memetic Comp. 18, 8 (2026).

[15]. Sun, Y., Lv, D. Y., & Huang, X. (2026). Properties for due window assignment scheduling on a two-machine no-wait proportionate flow shop with learning effects and resource allocation. Journal of the Operational Research Society, 77(2), 599–615. https://doi.org/10.1080/01605682.2025.2492817.

[16]. Birgin, E.G., Riveaux, J.A. and Ronconi, D.P. (2026), Energy-aware flexible job shop scheduling problem with nonlinear routes and position-based learning effect. Intl. Trans. in Op. Res., 33: 860-891. https://doi.org/10.1111/itor.70057.

[17]. Mousavipour, S.H., Farughi, H., Ahmadizar, F., "A job shop scheduling problem with sequence-dependent setup times considering position-based learning effects and availability constraints", International Journal of Industrial Engineering & Production Research, Vol. 30, 2019, pp. 329–340.

[18]. Mousavipour, S.H., Farughi, H., Ahmadizar, F., "Job shop scheduling problem based on learning effects, flexible maintenance activities and transportation times", Journal of Industrial and Systems Engineering, Vol. 12, No. 3, 2019, pp. 107–119.

[19]. Tayebi Araghi, M.E., Jolai, F., Rabiee, M., "Incorporating learning effect and deterioration for solving an SDST flexible job-shop scheduling problem with a hybrid meta-heuristic approach", International Journal of Computer Integrated Manufacturing, Vol. 27, No. 8, 2014, pp. 733–746. doi: 10.1080/0951192X.2013.834465.

[20]. Ma, Y., Chu, C., Zuo, C., "A survey of scheduling with deterministic machine availability constraints", Computers & Industrial Engineering, Vol. 58, No. 2, 2010.

[21]. Schmidt, G., "Scheduling with limited machine availability", European Journal of Operational Research, Vol. 121, No. 1, 2000, pp. 1–15. doi: 10.1016/S0377-2217(98)00367-1.

[22]. Tavakkoli-Moghaddam, R., Khalili, M., Naderi, B., "A hybridization of simulated annealing and electromagnetic-like mechanism for job shop problems with machine availability and sequence-dependent setup times to minimize total weighted tardiness", Soft Computing, Vol. 13, 2009, pp. 995–1006.

doi: 10.1007/s00500-008-0367-z.

[23]. Vahedi-Nouri, B., Fattahi, P., Ramezanian, R., "Hybrid firefly-simulated annealing algorithm for the flow shop problem with learning effects and flexible maintenance activities", International Journal of Production Research, Vol. 51, No. 12, 2014, pp. 3501–3515. doi: 10.1080/00207543.2012.750771.

[24]. Dehnavi-Arani, S., Sabaghian, A., Fazli, M., "A job shop scheduling and location of battery charging storage for the automated guided vehicles (AGVs)", Journal of Optimization in Industrial Engineering, Vol. 12, No. 2, 2019, pp. 123–131.

[25]. Bilchev, G., Parmee, I., "The ant colony metaphor for searching continuous design spaces", In Fogarty, T.C. (ed.), Evolutionary Computing, AISB Workshop, Springer, Sheffield, UK, 1995, pp. 25–39.

[26]. Mehrabian, A.R., Lucas, C., "A novel numerical optimization algorithm inspired from weed colonization", Ecological Informatics, Vol. 1, 2006, pp. 355–366.

[27]. Colorni, A.M., Dorigo, M., Maniezzo, V., "Distributed optimization by ant colonies", European Conference on Artificial Life, Paris, France, Elsevier Publishing, 1992, pp. 134–142.

[28]. Hossini, S., "Modeling and solving the job shop scheduling problem followed by an assembly stage considering maintenance operations and access restrictions to machines", Journal of Optimization in Industrial Engineering, Vol. 12, No. 1, 2019, pp. 63–78.

[29]. Yang, X.S., "Firefly algorithms for multimodal optimization", In Watanabe, O., Zeugmann, T. (eds.), Stochastic Algorithms: Foundations and Applications, SAGA 2009, Lecture Notes in Computer Science, Vol. 5792, Springer, Berlin, Heidelberg, 2009, pp. 169–178. doi: 10.1007/978-3-642-04944-6_14.

Follow this article at the following site:

Seyed Hamed Mousavipour, Hiwa Farughi. "A Genetic-Firefly Algorithm for a Sequence-Dependent Setup Times Job Shop Scheduling Problem Considering Learning Effects and Availability Constraints" IJIEPR 2026; 37 (3): 134-144
URL: http://ijiepr.iust.ac.ir/article-1-2645-en.html