Advances in Production Engineering & Management
Volume 15 | Number 2 | June 2020 | pp 204–216
https://doi.org/10.14743/apem2020.2.359
ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

Hybrid evolution strategy approach for robust permutation flowshop scheduling

Khurshid, B.[a], Maqsood, S.[b,*], Omair, M.[b], Nawaz, R.[a], Akhtar, R.[a]

[a]Department of Industrial Engineering, University of Engineering and Technology, Peshawar, Pakistan
[b]Department of Industrial Engineering, Jalozai Campus, University of Engineering and Technology, Peshawar, Pakistan

ABSTRACT

In this paper, a robust schedule has been proposed to deal with uncertainities for *m*-machines permutation flow shop problems. A robust schedule ensures that the expected finish time is always less than the makespan. To use the global search ability of the evolution strategy (ES) and local search ability of Tabu Search (TS), a hybrid evolution strategy (HES) is proposed by combining Improved ES with TS to generate the robust schedules. The robust schedule is first generated using ES and then the solution is optimized using TS for maximum exploitation and exploration of the solution space. For maximum exploitation in ES, *(1+9)* reproduction operator and double swap mutation is used. Also variable mutation rate is used for fine tuning of the results. In TS, the length of Tabu list is fixed, also lower bound is used to save computational time. The hybrid algorithm is tested on Carlier and Reeves benchmark problems taken from the OR-library. Achieved results are compared with other famous techniques available in the literature, and the results show that HES performs better than other techniques and provides an affirmative percentage increase in the probability that the expected finish time is less than the makespan.

ARTICLE INFO

Keywords:
Permutation flowshop;
Scheduling;
Carlier problem;
Reeves problem;
Evolutionary computation;
Hybrid evolution stratgey;
Improved evolution strategy;
Tabu search

*Corresponding author:
smaqsood@uetpeshawar.edu.pk
(Maqsood, S.)

Article history:
Received 1 April 2020
Revised 12 July 2020
Accepted 14 July 2020

1. Introduction

Scheduling in services and manufacturing sectors refers to the allocation of tasks or activities to resources in a determined process for minimization of costs and optimizing operational efficiency [1]. Based on the structure of the environment, scheduling has many types. Flow shop scheduling has practical implementation in automobile, pharmaceutical, pump, chemical, steel industry and is addressed in this paper [2]. Permutation flow shop scheduling problems (PFSSP) has been the focus for most researchers in the recent past, however they have mostly focused on a deterministic environment. Study of deterministic PFSSP is relatively easier as all the parameters are known in advance [3].

However uncertain events arrive in real-life situations and the uncertainties are induced in processing times, which will affect the makespan of schedule [4]. Thus there are chances that some jobs will not be completed in the expected finish time and hence the production gets delayed. Hence more focus should be paid to uncertainties in PFSSP [5, 6].

In stochastic scheduling, processing times of jobs are random and till their finish they are unknown. Hence in scheduling, real-life situations should be able to deal with these stochastic events. A good scheduling environment should cope with these uncertain situations and produce a robust schedule. If the behavior of a schedule does not degrade significantly in the face of a disruption it is termed as a robust schedule. Hence a robust schedule is insensitive to the occurrence of uncertain events. Robust scheduling is currently being used in various real-life situations to, i.e. manufacturing industry, airline crew scheduling, project scheduling, and process industry. The present research is focused on robust scheduling for the manufacturing industry.

The two common approaches presently used in robust scheduling are reactive and proactive scheduling [7]. In proactive scheduling, future disruptions are taken into account while generating the initial schedule. However reactive scheduling does not take future disruptions in account while generating the initial schedule, and the schedule is revised after the occurrence of a disruption. In this paper, a proactive approach is used for robust scheduling.

Since the deterministic PFSSP is classified as an NP-hard problem [8], therefore the stochastic PFSSP is also an NP-hard problem. Hence exact methods cannot be used to solve the large size problem, therefore approximation methods should be used for their solutions [9]. Under uncertainties, González-Neira *et al.* [6] carried out a detailed literature review of flow shop problems. He defined role of scheduling and guided further areas of research. He surveyed 100 papers and in most of the papers makespan was the main objective, while very limited papers were focused on multi objectives.

The remainder of the paper is organized as follows. A detailed literature review is provided in Section 2. Section 3 explains the problem statement while methodology is presented in Section 4. The robust schedule is explained in Section 5 while computational results are explained in Section 6. Finally, conclusions and recommendations are suggested in Section 7.

2. Literature review

Robustness of a schedule can be defined using different measures [10]. Mostly researchers have minimized the variation of a performance indicator [11-13]. However, the robust schedules do not guarantee that the performance indicator will not exceed limits as uncertainty occurs. Initially, Daniels and Carrillo [14] proposed a robust solution to address the uncertainties of processing times in a single machine environment using a Branch and Bound (B & B) method and a heuristic approach. They discovered the risk of the inferior system, by considering the variance and mean performance in appraising alternative schedules. An exact dynamic program was used by Yang and Yu [11] to find a robust schedule for a single machine environment with three performance measures. Kuo and Lin [15] Studied robustness for a single machine environment with total flow time as the objective function. The authors used a fractional programming framework to find robust schedules. Using an algorithm they suggested worst case scenario and worst case relative deviation, then a parametric programming algorithm was used to find robust schedule.

A robust multi-objective problem was studied by Jia and Ierapetritou [16] incorporating robustness, economic expectation and expected unsatisfied demand. Based on a Tabu Search (TS) algorithm, a robust and stable schedule was proposed by Goren and Sabuncuoglu [13]. For a single machine scheduling problem, the proposed algorithm-generated robust and stable schedules for three performance parameters, i.e. total flow time, makespan and total tardiness. The algorithm constitutes two parts, i.e. sequence generator and sequence evaluator. In order to cope with uncertain situations in a single machine environment, a robust schedule was proposed by Wu *et al.* [17] using dominance rules. The objective was to minimize the risk that makespan will not exceed the maximum threshold. He proposed three models, i.e. combined, primal and a dual model to solve the problem, however, the combined method was the most efficient one.

Mostly the research has been focused on a single machine and it doesn't represent actual cases of PFSSP. By increasing the number of machines, the effect of uncertainty in scheduling environment increases. A robust schedule for m-machines PFSSP was first studied by Liu *et al.* [7] using an improved genetic algorithm (IGA). The IGA used a new generation scheme to preserve the good characteristics of parents. The IGA was tested on Carlier and Reeves problems, and compared with the results of NEH heuristic. Results showed that IGA performed better than NEH and also produced an affirmative percentage that the expected finish time is always less than the makespan.

For machine breakdown uncertainty, Fazayeli *et al.* [18] proposed a Hybrid Meta heuristic for maximizing β-robustness of makespan, he hybridized genetic algorithm with simulated annealing and outperformed six other heuristics. For uncertain setup and processing times, Gholami-Zanjani *et al.* [19] suggested a Robust optimization and a Fuzzy Optimization approaches to minimize the weighted mean completion time. For two machine problem, Rahmani [20] used surrogate measures for uncertain processing times and machine failures. The algorithm worked in two step structure, initially a schedule was generated for processing times and then a reactive schedule was generated after machine failure. In view of the robust starting times, Cui *et al.* [21] Proposed a Weibull distribution considering machine failures to minimize makespan. The algorithm proceeded in two loop, first local serach is initialized using NEH and then genetic algorithm is used in second loop.

For the uncertainties of processing times in m-machines, Ma *et al.* [22] proposed a robust schedule based on an artificial bee colony algorithm (ABC). To enhance the exploitation and exploration of ABC, an improved local search scheme was introduced in ABC. The algorithm was tested on famous Carlier and Reeves problems and compared with the results of IGA and NEH. Results show that ABC finds a better robust schedule as compared to IGA and NEH. For stochastic processing times, González-Neira *et al*. [23] proposed a simheuristic approach to solve multi objective PFSSP. The objectives are minimization of expected deviation of tardiness and tardiness. Tabu Search is hybridized with a Monte Carlo simulation and a Pareto archived evolution strategy in the simheuristic approach and tested on 540 benchmark instances. In addition he applied the simheuristic approach on an optical laboratory case and found better results for expected and standard deviation as compared to the sequences used by the laboratory.

Recently various meta-heuristics have been used by numerous researchers for PFSSP and other related problems, e.g. GA, ABC, TS, ES [24-31]. Although GA is fast, however, their searchability is reduced due to their lack of local search adaptability. ABC algorithms are simple however they often get trapped in local optimum [22]. TS is a local search technique and is widely used for the discrete optimization problem, however, it becomes slow for large-sized problems. ES was developed by Rechenberg in 1970 for parameter optimization problems, however recently they have been applied to discrete problems. ES performs better for global search and contains information of the previous solutions, however sometime it gets trapped in a local minima after certain iterations, in this condition the ES jumps out of the local minima by using large mutation rate.

The two key parameters for the performance of any algorithm are: i) Exploiting the best solutions and ii) Exploring the search space. Hence the global search ability of ES should be combined with local search ability of TS for better performance. Hence in this paper, a hybrid evolution strategy (HES) is proposed by combining an ES with TS to find robust schedules for m-machines PFSSP. The ES is improved by using, *(1+9)* reproduction operator and double swap mutation operator, also variable mutation rate is used for fine tuning of the results. To save the computation time of TS, the length of Tabu list is fixed and lower bound is used. With all these improvements, the HES finds optimal solution in minimum computational time.

3. Problem statement

In a PFSSP, n-jobs are processed in fixed order on m-machines. At time zero all the jobs are available and they are independent. The first machine has no idle time and each machine can simultaneously process one job. The processing time of the job includes setup and transportation times. Each job has M operations and they are performed on different machines. The processing order of the job is the same on every machine. All machines are continuously available and preemption is not allowed. The due date for all jobs is the same and there is unlimited buffer space. Processing time of job I on machine j is represented at Pij and the Corresponding finish time is represented at Ci_j. The finish time for the last job is termed as makespan and is represented at C_{max}. Mathematically PFSSP can be represented as below:

$$C_{11} = P_{11} \tag{1}$$

$$Ci_1 = \sum_{r=1}^{i} P_{r_1} \qquad \text{where } i = 1,2,\dots,N \tag{2}$$

$$C_{1_j} = \sum_{r=1}^{j} P_{1_r} \qquad \text{where } j = 1,2,\dots,M \tag{3}$$

$$C_{max} = \max\{C_i(j-1), \quad C(i-1)\} + P_{ij} \tag{4}$$

$$Objective = max\{Probability(C_{max} \leq Expected\ Finish\ Time)\} \tag{5}$$

4. Robust schedule

When the processing times for jobs are uncertain, then for m-machines PFSSP and n-jobs, a schedule which ensures maximum probability that the expected finish time is less that the make-span is termed as robust schedule. Mathematically, for a robust schedule the probability ($C_{\max} \leq X$) is maximum.

The processing time uncertainty in a robust schedule can be presented by fuzzy logic or probabilistic technique. The probabilistic technique is normally used for processing time uncertainty, where the distribution describes the processing time uncertainty. Actual data is taken from the jobs processing time, and then it is analyzed to know the distribution of processing time. Ample processing time data through repeat production is required to know the distribution of processing time. The processing time uncertainty is an independent random variable and according to the central limit theorem, its distribution formed is a normal distribution. Hence job processing time follows normal distribution according to the central limit theorem. If two numbers A and B follow normal distribution then their sum $A + B$ also follows a normal distribution. Hence makespan computed from normally distributed processing times follows a normal distribution as depicted in Fig. 1.

For normally distributed makespan, the probability that the expected finish time is less than makespan is calculated using Eq. 4.

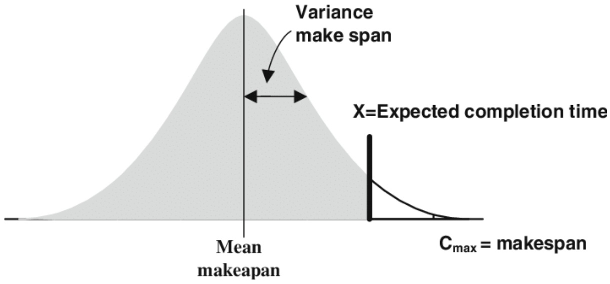

Fig. 1 Normal distribution curve for makespan

$$\text{Probability } (Cmax \leq X) = \frac{1}{2} + \varphi(z) \tag{6}$$

$$\text{where} \quad z \geq 0 \text{ and } z = \frac{X - \mu Cnm}{6CnMx} \tag{7}$$

$$\text{and} \qquad \varphi(z) = \frac{1}{\sqrt{2p}} \int_{-\propto}^{z} e^{-(t2/2)}\, dt \tag{8}$$

$$\emptyset(z) \approx \varphi(z) = \begin{cases} 0.1z(4.4-z) & (0 \leq z \leq 2.2) \\ 0.49 & (2.2 \leq z \leq 2.6) \\ 0.50 & (z \geq 2.6) \end{cases} \tag{9}$$

It is difficult to find the value of z through the exact method. However, Hayter [32] proposed a mathematical equation to find its approximate value as shown in Eq. 9.

5. Methodology

5.1 Evolution strategy

Evolution strategy (ES) is a subclass of an evolutionary algorithm and was developed was Rechenberg [33]. ES is based on the Darwinian Paradigm of evolution and its performance depends on the strength of its various genetic operators, i.e. selection, reproduction, recombination, evaluation. ES operates with a population of size *(μ + λ)*, where μ represents individual parent and λ represents the offspring. ES is an iterative process developed for the numerical optimization process and the solution space is searched through the population of individual solutions. The mutation operator is the main genetic operator in ES [34]. In Literature different reproduction operators have been used for ES, i.e. *(1+1), (1+4), (1+9), (1+16)* ([35, 36]).

Two selection operators are normally used in ES, i.e. *(μ+λ)-ES* and *(μ, λ)-ES*. In *(μ+λ)-ES*, both the parents and offspring's take part in the selection process and it is recommended for combinatorial optimization problems. While in *(μ, λ)-ES* only the offspring's take part in the selection process and it is recommended for real value parameter optimization problems.

The performance of ES is heavily dependent on the strength of its mutation operator and is the main source of genetic variation in ES. The mutation operator is problem-dependent, and their appropriate selection is an art. Various mutation operators can be used for PFSSP, however, swap operators are best for PFSSP [37].

5.2 Improved evolution strategy (IES)

In order to maximize the exploration and exploitation of the solution space, the following improvements have been made (procedure for the IES is in Fig. 2):

Pseudocode for the IES

Input Parameters:
Total Number of generations
Mutation rate
Population size
Number of off springs to be produced from parent(λ=9)

Record Parameters: Total number of generations and best makespan

```
1:   Parent population randomly generated, Pp
2:   for gen =1: k
3:   Evaluate parent population
4:   If number of generations (k) is not achieved go to step 7
5:         else
6:   end
7:         for v= 2: μ
8:   Produce offsprings and apply mutation operator (double swap mutation operator)
9:         Update population, Pi
10:        Evaluate all offsprings
11:        end for
12:  Evaluate Pi and fittest individual should be termed as parent for new iteration
13:  If an offspring has minimum makespan it is termed as candidate (OS1-OS9=C)
14:        else
15:        Parent is termed as candidate (P=C)
16:        end
17:  Until number of iterations are achieved go to step 2
18:  Record Total number of generations and best makespan
```

Fig. 2 Pseudocode for IES

- For maximum exploitation of the solution space, *(1+9)* reproduction operator is used. From 1 parent 9 offspring's are generated.
- The selection scheme used is *(1+9)* instead of *(1, 9)*, hence the selection pool consists of 10 entities, and the best offspring is selected from the pool, so the parent can survive for many generations. In *(1, 9)* parents die out of the selection pool and only children are available for selection.
- In order to increase the genetic variation, the large mutation rate is used initially and then it is reduced for fine-tuning of global minima.
- To save computational time and for maximum exploitation of search space, double swap mutation operator is used.

5.3 Tabu search (TS)

TS is a local search technique where the neighborhood is deterministically selected. For combinatorial optimization problems, TS is one of the most effective local search techniques for finding near-optimal solutions [38]. The local search technique starts with a solution and then find the best solution in the neighborhood. TS is good to avoid local minima and improving the solution through iterations. TS helps in exploring the solution beyond local optimality. TS starts from a basic schedule and by searching its neighborhood, moves generate a set of iterations with the lowest makespan. The exploration is then started from the previous best iteration as a new iteration and the search process continues.

The key steps of the TS algorithm are an initial solution, evaluation, move, neighborhood search, memory, searching strategy and termination criteria.

To avoid duplication of job swaps, a record of moves is kept in the list termed as Tabu List. The key benefit of TS is the use of Tabu List to avoid duplication and overcoming local optima and guiding the solution to the region which have not been explored. The infinite length of the tabu list is not recommended as it slows down the algorithm. For finite Tabu List, FIFO strategy is used, so as new attributes are inserted the old attributes are removed. The algorithm is terminated if the number of iterations have reached, or the computational time has reached, or there is no improvement in makespan after significant iterations, or the neighborhood is empty. The performance and speed of convergence for any TS algorithm is based on the accurate design of its components.

Following improvements have been made in the basic TS algorithm:

- To save the computational time of our TS algorithm, we have used lower bound and a number of iterations instead of explicitly finding the best makespan.
- To avoid local minima, Tabu List having fixed length is used.
- Since the TS improves the solution found by IES, hence global search ability of ES is united with local search ability of TS to find the best results.

Move and neighborhood

Every solution in our algorithm is represented by iteration. Using a set of moves, the neighborhood of a solution is generated. Then, using the Move function, the solution is transformed into another solution. The subset of moves associated with a given solution produces a set of solutions known as neighborhood. At each iteration, the neighborhood is searched to find the best in the neighborhood. Again move is performed, and a solution is generated from the previous best solution and becomes the current solution and the iteration continues.

The two main types of moves for TS are i) E-move, Exchange Jobs at the a^{th} and b^{th} position. ii) *L*-move, Remove a job at a^{th} position and put it at b^{th} position. Although E-move is complex however it provides a better result and has been used in this paper. Since ample computational time is required for the large neighborhood, hence to save computation time lower bounds are used instead of calculating makespan explicitly.

Tabu list

The primary purpose of Tabu list is used to prevent cycling during the search. In literature various methods are available for implementation of Tabu list: i) Pairs of executed iterations, ii) Job and its position, iii) Makespan of executed iterations, iv) Pair of jobs along with their positions. The design of the appropriate Tabu list is a major factor for the performance and convergence of the TS algorithm.

Based on problem type the length of Tabu List can be fixed or dynamic. In this paper fixed length has been used. Let $TL= (TL_1,\ldots, TL_t)$ be a fixed Tabu List with length t, and $Tj=(g, h)$ is a pair of jobs. Initially, the Tabu List is started with zero elements $Tj= (0, 0)$, where $j=1,.., t$. Let $\upsilon= (a, b)$ a move performed from an iteration π, after this move the Tabu List will be updated $Tj=Tj+1$ where $j=1,.., t-1$. Then set $Tt= (\pi (a), \pi (a+1))$ if $a<b$ and $Tt= (\pi (a-1), \pi (a))$ if $a>b$. An iteration β cannot be performed from a move $\upsilon= (a, b)$ if minimum one pair $(\beta (j), \beta (a)), j=a+1,\ldots, b$ is in T if $a<b$ and minimum one pair $(\beta (a), \beta (j)), j=b,..a-1$ is in T.

Procedure for the TS is in Fig. 3:

Pseudocode for the TS

Input:
Number of iterations, k
Size of move list ((a1,b1), (ai+1,bi+1) where i=…n
Total generations, k

```
1:    Use the final sequence from ES as a starting sequence(So) for TS
          So= (J1, J2,.. Jn) where n=number of jobs
2:    for gen =1: k
3:    Evaluate sequence, G(So)
4:    Interchange adjacent wise pairs of So to generate new sequence(Sc) (Exchange Jobs at the a^th and b^th
position)
5:    Evaluate candidate sequence G(Sc)
6:    IF
7:            G(Sc)<G(So)
          Then        So=Sc
8:                    Update move in Tabu List(a1≠a1+1≠b1≠b1+1)
9:            end
10:  IF
11:           Cmax=LB
12:           Go to step 16
13:           Else Continue
14:           End
15:   repeat until loop (Go to step 2)
16:   Output Cmax and Total generations
```

Fig. 3 Pseudocode for TS

6. Computational results and discussion

To evaluate the robustness of HES, results are compared with other well-known techniques available in the literature. Probability ($C_{\max} \leq X$) of HES is compared with the probability ($C_{\max} \leq X$) of other known techniques. Robust schedule dealt is the paper was first presented by Liu, Ullah [7] for m-machines PFSSP, he used IGA to find a robust schedule and compared his results with the robust schedule generated by NEH heuristic [39]. The IGA performed better than NEH based on the objective of probability ($C_{\max} \leq X$). An ABC algorithm was suggested by Ma, Wang [22] for m-machines PFSSP and found better results than IGA and NEH. Hence in order to validate HES for robust schedules it results will be compared with IGA, NEH, and ABC. Results of IGA and ABC are taken from their original papers. Results of HES cannot be compared with other technqiues as robust schedules for benchmark problems of Carlier [40] and Reeves [41] are only generated by IGA, NEH and ABC.

The expected finish time is required to compare the probability ($C_{max} \leq X$). Hence in order to compare the probability ($C_{max} \leq X$) of HES, IGA, NEH and ABC, some assumptions are made. The probability value for NEH schedule is assumed and the analogous z-value is computed using Eqs. 6 and 9. Using the z-value the expected finish time is calculated using Eq. 7. Then the expected finish time is used to compute the probability ($C_{max} \leq X$) of HES. Finally, the probability of HES, ABC, and IGA are compared with the probability of NEH.

The technique has been evaluated on famous benchmark problems of Carlier [40] and Reeves [41] and their data set has been taken from OR-Library. The processing times in the data set are deterministic, hence the deterministic times are assumed as the mean processing times. While the variance of processing times is calculated from an assumed interval [1, μij/9], where μij is the mean processing time. The mean and variance of processing times for Car01 and Rec01 problems are shown in Tables 1 and 2.

The hybrid ES was coded in MATLAB and tested on Laptop with 2.1 GHz processor and 4 GB Ram. While the IGA, ABC, and NEH were all coded in Visual C. Three probabilities of 0.80, 0.85 and 0.90 were assumed for NEH Heuristic, hence for each problem three instances are used. The z-value for each instance is calculated using Eqs. 6 and 9. Each instance is first solved using NEH to find mean and variance and then expected makespan as shown in Tables 3 and 4. The expected finish time calculated using NEH is assumed to be the same for the HES, IGA and ABC. The mean and variance makespan is then calculated using HES. Then the z-values are calculated using Eq. 9. Finally, probability ($C_{max} \leq X$) is calculated using Eq. 9 for each instance.

Table 1 Processing times (mean and variance) for Car 1 (11×5) instance

Car 1 Problem										
Jobs	Mean processing times					Variance processing times				
	M1	M2	M3	M4	M5	M1	M2	M3	M4	M5
1	375	12	142	245	412	11	1	11	3	6
2	632	452	758	278	398	15	4	1	9	8
3	12	876	124	534	765	1	1	7	16	2
4	460	542	523	120	499	2	10	6	7	2
5	528	101	789	124	999	11	8	2	2	8
6	796	245	632	375	123	4	6	12	10	12
7	532	230	543	896	452	11	10	2	9	5
8	14	124	214	543	785	1	9	3	16	17
9	257	527	753	210	463	19	10	3	2	1
10	896	896	214	258	259	1	4	3	8	13
11	532	302	501	765	988	8	2	13	1	13

Table 2 Processing times (mean and variance) for Rec 1 (20×5) problem

Rec 1 Problem (20×5)										
Jobs	Mean Processing Times					Variance Processing Times				
	M1	M2	M3	M4	M5	M1	M2	M3	M4	M5
1	5	76	74	99	26	1	8	8	11	2
2	74	21	83	52	90	8	2	9	5	10
3	67	48	6	66	38	7	5	1	7	4
4	97	36	71	68	81	10	4	7	7	9
5	87	86	64	11	31	9	9	7	1	3
6	1	42	20	90	23	1	4	2	10	2
7	69	32	99	26	57	7	3	11	2	6
8	69	12	54	80	16	7	1	6	8	1
9	11	63	24	16	89	1	7	2	1	9
10	87	52	43	10	26	9	5	4	1	2
11	25	59	88	87	40	2	6	9	9	4
12	50	42	72	77	29	5	4	8	8	3
13	58	76	71	82	94	6	8	7	9	10
14	79	48	20	63	97	8	5	2	7	10
15	35	57	78	99	80	3	6	8	11	8
16	70	76	53	2	19	7	8	5	1	2
17	79	22	77	74	95	8	2	8	8	10
18	34	99	49	3	61	3	11	5	1	6
19	37	24	32	25	4	4	2	3	3	1
20	50	88	46	63	76	5	9	5	7	8

In Eq. 10, The proportion increase in probability ($C_{\max} \leq X$) for each instance is introduced while using Eq. 11 the proportion decline in risk can be calculated. Both these parameters will be used to compare the proposed HES, IGA and ABC.

$$\%_{increase}\ Probability = \frac{\varphi(z)_{ES} - \varphi(z)_{NEH}}{\varphi(z)_{NEH}} \tag{10}$$

$$\%_{increase}\ Risk = \frac{(1-\varphi(z)_{NEH}) - (1-\varphi(z)_{ES})}{(1-\varphi(z)_{ES})}\ x\ 100 \tag{11}$$

6.1 Carlier problems

From Table 3 it is evident that HES finds better robust schedules than NEH, IGA and ABC for Carlier Problems. HES finds minimum makespan for all instances as compared to NEH and IGA. Except Car-03 instance, makepan values found by HES for all other instances is minimum as compared to ABC. The probability found by HES is higher than the probabilities of HES and IGA. For instance Car-03, probability of ABC is better, while for all other instances probability of HES is higher than ABC. Probability is almost 100 % that the makespan is less than the expected finish time. Almost 100 % probability is attained due to the assumptions used for computing $\phi(z)$. With HES, the minimum percent increase in the probability of 15 % and 13.09 % decrease in risk for Car-01 problem. While the maximum increase in the probability of 66.67 % is obtained for Car-02, Car-04 and Car-08 problems. A maximum percent decrease in risk of -42.88 % is obtained for Car-02 problem.

From Figs. 4 and 5 it is evident that for schedules generated using HES, there is the maximum probability that the makespan is less than the expected finish time and also ensures that the decrease the risk of exceeding makespan from a large gap.

Table 3 Carlier problem results

Prob	Size	Nawaz-Enscore-Ham Heuristic				Improved genetic algorithm			Artificial bee colony algorithm			Hybrid evolution strategy			Percent increase in probability with ES	Percent decrease in risk with ES
		Prob	ø(z)	Cmax	X	X	Cmax	Prob	X	Cmax	Prob	X	Cmax	Prob		
Car 01	11×5	0.8	0.3	7038	7152.73	7152.73	7050	0.87	7152.73	7040	0.862	7152.73	7038	0.88	26.67	13.46
		0.85	0.35	7038	7179.78	7179.78	7050	0.917	7179.78	7040	0.913	7179.78	7038	0.93	22.86	14.38
		0.9	0.4	7038	7212.56	7212.56	7059	0.934	7212.56	7040	0.957	7212.56	7038	0.96	15.00	13.09
Car 02	13×4	0.8	0.3	7376	7534.52	7534.52	7180	1	7534.52	7166	1	7534.52	7166	1	66.67	-4.97
		0.85	0.35	7376	7571.89	7571.89	7176	1	7571.89	7166	1	7571.89	7166	1	42.86	-24.55
		0.9	0.4	7376	7617.19	7617.19	7200	1	7617.19	7186	1	7617.19	7166	1	25.00	-42.88
Car 03	12×5	0.8	0.3	7399	7543.52	7543.52	7403	0.922	7543.52	7312	0.95	7543.52	7399	0.92	40.00	21.25
		0.85	0.35	7399	7577.59	7577.59	7312	0.925	7577.59	7312	0.972	7577.59	7399	0.96	31.43	21.23
		0.9	0.4	7399	7618.9	7618.90	7401	0.947	7618.90	7312	0.984	7618.90	7399	0.98	20.00	16.24
Car 04	14×4	0.8	0.3	8129	8270.02	8270.02	8024	1	8270.02	8011	1	8270.02	8003	1	66.67	32.63
		0.85	0.35	8129	8303.26	8303.26	8004	1	8303.26	8014	1	8303.26	8003	1	42.86	16.17
		0.9	0.4	8129	8343.56	8343.56	8018	1	8343.56	8106	1	8343.56	8003	1	25.00	-3.98
Car 05	10×6	0.8	0.3	7835	7946.51	7946.51	7758	0.969	7946.51	7727	0.966	7946.51	7720	0.98	60.00	34.83
		0.85	0.35	7835	7972.8	7972.80	7798	0.97	7972.80	7727	0.979	7972.80	7720	1	42.86	25.89
		0.9	0.4	7835	8004.67	8004.67	7758	0.99	8004.67	7727	0.99	8004.67	7720	1	25.00	13.71
Car 06	8×9	0.8	0.3	8773	8893.55	8893.55	8570	1	8893.55	8505	1	8893.55	8210	0.99	63.33	35.66
		0.85	0.35	8773	8921.97	8921.97	8570	1	8921.97	8505	1	8921.97	8210	0.99	40.00	25.74
		0.9	0.4	8773	8956.43	8956.43	8570	1	8956.43	8505	1	8956.43	8210	1	25.00	15.31
Car 07	7×7	0.8	0.3	6590	6702.77	6702.77	6590	0.754	6702.77	6590	0.881	6702.77	6490	0.95	50.00	27.80
		0.85	0.35	6590	6729.35	6729.35	6590	0.8	6729.35	6590	0.93	6729.35	6490	0.97	34.29	22.90
		0.9	0.4	6590	6761.58	6761.58	6590	0.85	6761.58	6590	0.968	6761.58	6490	0.98	20.00	16.00
Car 08	8×8	0.8	0.3	8564	8694.23	8694.23	8366	1	8694.23	8424	1	8694.23	7790	1	66.67	34.11
		0.85	0.35	8564	8724.94	8724.94	8366	1	8724.94	8420	1	8724.94	7790	1	42.86	23.41
		0.9	0.4	8564	8762.16	8762.16	8429	1	8762.16	8424	1	8762.16	7790	1	25.00	12.30

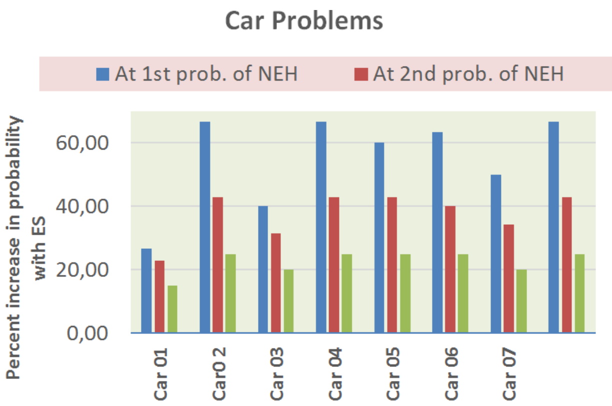

Fig. 4 Comparison of Prob. on Carlier problems

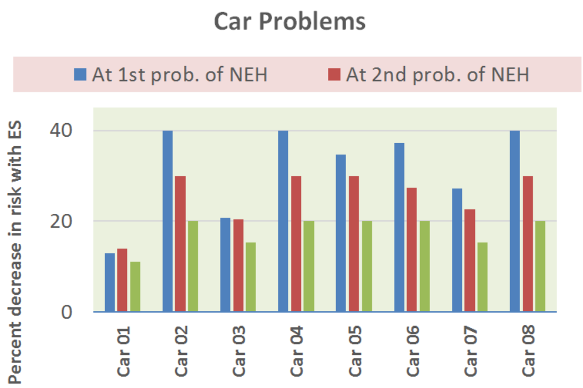

Fig. 5 Comparison of risk on Carlier problems

6.2 Reeves problems

The results of HES for Reeves's problems are shown in Table 4. Results of the HES are better as compared to NEH, IGA and ABC. HES finds minimum makespan for all instances as compared to NEH, IGA and ABC. Also the probability found by HES is higher than the probabilities of HES, IGA and ABC. Using HES, the probability ($C_{max} \leq X$) for all Rec problems is 100 %. Minimum increase in the probability of 25 % and a decrease in risk of 20 % is observed for Rec-01 problem. While maximum probability of 66.67 % is observed for Rec-01, Rec-03, Rec-05, Rec-07, Rec-09, Rec-11, Rec-13, Rec-15 and Rec-17 problems and maximum decrease in risk of 40 % is observed for Problem 1, 3, 5, 7, 9, 11, 13, 15 and 17 respectively.

From Figs. 6 and 7 it is evident that for schedules generated using HES, there is a maximum probability that the makespan is less than the expected finish time and also ensures that the decrease the risk of exceeding makespan from a large gap for Reeves problems.

Table 4 Reeves problem results

Prob	Size	Nawaz-Enscore-Ham heuristic				Improved genetic algorithm			Artifical bee colony algorithm			Hybrid evolution strategy			Percent increase in probability with ES	Percent decrease in risk with ES
		Prob	ø(z)	Cmax	X	Cmax	ø(z)	Prob	Cmax	ø(z)	Prob	Cmax	ø(z)	Prob		
Rec 1	20×5	0.8	0.3	1320	1340.82	1271	0.5	1	1279	0.5	1	1249	0.5	1	66.67	40.00
		0.85	0.35	1320	1345.72	1268	0.5	1	1291	0.5	1	1249	0.5	1	42.86	30.00
		0.9	0.4	1320	1351.67	1288	0.5	1	1293	0.5	1	1249	0.5	1	25.00	20.00
Rec 3	20×5	0.8	0.3	1116	1133.02	1115	0.312	0.812	1111	0.452	0.952	1109	0.5	1	66.67	40.00
		0.85	0.35	1116	1137.03	1111	0.397	0.897	1112	0.475	0.975	1109	0.5	1	42.86	30.00
		0.9	0.4	1116	1141.89	1111	0.437	0.937	1113	0.49	0.99	1109	0.5	1	25.00	20.00
Rec 5	20×5	0.8	0.3	1296	1315.80	1255	0.5	1	1261	0.5	1	1245	0.5	1	66.67	40.00
		0.85	0.35	1296	1320.47	1247	0.5	1	1266	0.5	1	1245	0.5	1	42.86	30.00
		0.9	0.4	1296	1326.13	1268	0.5	1	1277	0.5	1	1245	0.5	1	25.00	20.00
Rec 7	20×10	0.8	0.3	1626	1645.89	1584	0.5	1	1584	0.5	1	1572	0.5	1	66.67	40.00
		0.85	0.35	1626	1650.58	1599	0.5	1	1584	0.5	1	1572	0.5	1	42.86	30.00
		0.9	0.4	1626	1656.26	1601	0.5	1	1596	0.5	1	1572	0.5	1	25.00	20.00
Rec 9	20×10	0.8	0.3	1583	1602.40	1580	0.340	0.840	1557	0.5	1	1537	0.5	1	66.67	40.00
		0.85	0.35	1583	1606.98	1575	0.448	0.948	1557	0.5	1	1537	0.5	1	42.86	30.00
		0.9	0.4	1583	1612.52	1574	0.464	0.964	1547	0.5	1	1537	0.5	1	25.00	20.00
Rec 11	20×10	0.8	0.3	1550	1569.91	1502	0.5	1	1502	0.5	1	1431	0.5	1	66.67	40.00
		0.85	0.35	1550	1574.60	1509	0.5	1	1509	0.5	1	1431	0.5	1	42.86	30.00
		0.9	0.4	1550	1580.29	1491	0.5	1	1491	0.5	1	1431	0.5	1	25.00	20.00
Rec 13	20×15	0.8	0.3	2002	2025.90	1981	0.465	0.965	1981	0.5	1	1935	0.5	1	66.67	40.00
		0.85	0.35	2002	2031.54	1969	0.5	1	1969	0.5	1	1935	0.5	1	42.86	30.00
		0.9	0.4	2002	2038.37	1979	0.5	1	1979	0.5	1	1935	0.5	1	25.00	20.00
Rec 15	20×15	0.8	0.3	2025	2045.59	1986	0.49	0.99	1986	0.5	1	1962	0.5	1	66.67	40.00
		0.85	0.35	2025	2050.45	1998	0.49	0.99	1998	0.5	1	1962	0.5	1	42.86	30.00
		0.9	0.4	2025	2056.33	1997	0.5	1	1997	0.5	1	1962	0.5	1	25.00	20.00
Rec 17	20×15	0.8	0.3	2019	2044.40	1992	0.484	0.984	1992	0.5	1	1939	0.5	1	66.67	40.00
		0.85	0.35	2019	2050.39	1986	0.5	1	1986	0.5	1	1939	0.5	1	42.86	30.00
		0.9	0.4	2019	2057.65	1992	0.5	1	1994	0.5	1	1939	0.5	1	25.00	20.00

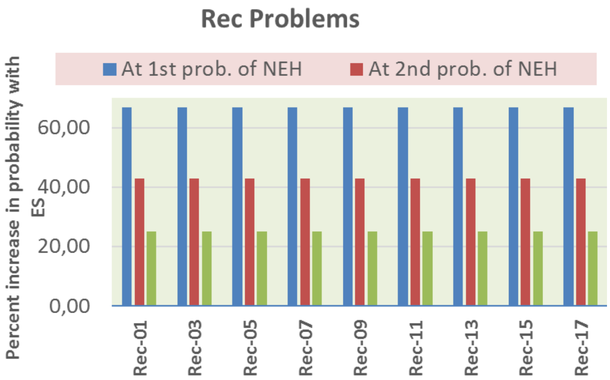

Fig. 6 Comparison of probabilities on Reeves problems

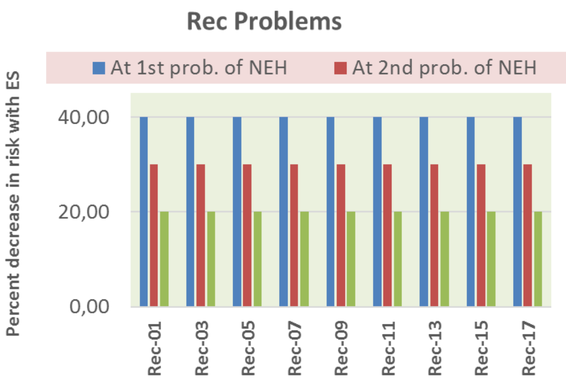

Fig. 7 Comparison of risk on Reeves problems

6.3 Overall results

Fig. 8 demonstrates the average rise in probability and average percentge decline in risk for all Carlier and Reeves problems. It is evident that for all Carlier and Reeves problems, HES provides better probability that the expected finish time is less than the makespan for Carlier and Reeves instances (more than 38 % and 42 % respectively). Also HES ensures a decline in risk that expected finish time will not exceed the makespan for Carlier and Reeves instances (more than 22 % and 29 % respectively). The overall results show that HES outperforms IGA, NEH and ABC and ensures that the robust schedules generated by HES gives higher probability that they will not exceed the expected finish time for Carlier and reeves problem. Also robust schedules generated using HES ensures the decline in risk that the makespan will exceed the expected finish time for Carlier and Reeves instances.

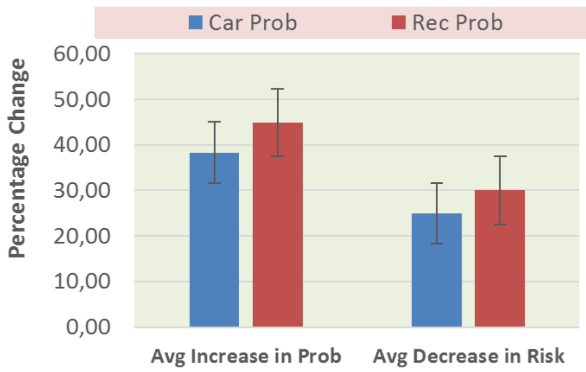

Fig. 8 Avg. percentage increase in Prob. and Avg. percentage decrease in risk by HES

7. Conclusion and recommendations

Robust schedules are generated for *m*-machines PFSSP to address the uncertainty of processing times. The objective is to ensure that the expected finish time is less than the makespan. As per the central limit theorem, processing time for uncertain jobs is normally distributed. A hybrid ES has been proposed and evaluated on famous benchmark problems of Carlier and Reeves. First ES is executed for 30,0000 iterations and then the solution is optimized using TS. The hybrid algorithm ensures maximum exploration as well as maximum exploitation of solution space. Results are compared with the NEH, IGA and ABC algorithms, and HES has outperformed all of them for all the Carlier and Reeves instances. The present research is focused on robust scheduling for the manufacturing industry, and it ensures that the expected finish time of jobs will not exceed the deadline as there are fluctuations in processing times of jobs in manufacturing industry. Hence the research is applicable to other industrial cases, i.e. process and chemical industry, pharmaceutical industry, steel industry etc. as the processing times in these industries are uncertain.

For simplicity it is assumed that finish time is same for all jobs, however, in actual flow shop, the finish times are different. Hence future research should be focused on PFSSP with all job have different expected finish times. Also, the research can be extended to flow shop with other objectives, i.e. tardiness, and flowtime. So far robust schedules were generated for small size flow shop problems. Hence in the future, robust schedules should be generated for large size problems of Reeves, Taillard, and Valllada benchmark instances. Also robust schedules for job shop problems are also still pendent. In addition robust schedules can be generated for other performance measures, i.e. tardiness, flowtime etc.

References

[1] Pinedo, M.L. (2015). *Scheduling*, *Theory, Algorithms, and Systems,* Fifth Edition*,* Springer, New York, USA, doi: 10.1007/978-3-319-26580-3.

[2] Sturrock, D. (2012). New solutions for production dilemmas, *Industrial Engineer,* Vol. 44, No. 12, 47-52.

[3] Gourgand, M., Grangeon, N., Norre, S. (2003). A contribution to the stochastic flow shop scheduling problem, *European Journal of Operational Research*, Vol. 151, No. 2, 415-433, doi: 10.1016/S0377-2217(02)00835-4.

[4] Juan, A.A., Barrios, B.B., Vallada, E., Riera, D., Jorba, J. (2014). A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times, *Simulation Modelling Practice and Theory*, Vol. 46, 101-117, doi: 10.1016/j.simpat.2014.02.005.

[5] Elyasi, A., Salmasi, N. (2013). Stochastic flow-shop scheduling with minimizing the expected number of tardy jobs, *The International Journal of Advanced Manufacturing Technology*, Vol. 66, No. 1-4, 337-346, doi: 10.1007/s00170-012-4328-4.

[6] González-Neira, E.M., Montoya-Torres, J.R., Barrera, D. (2017). Flow-shop scheduling problem under uncertainties: Review and trends, *International Journal of Industrial Engineering Computations*, Vol. 8, No. 4, 399-426, doi: 10.5267/j.ijiec.2017.2.001.

[7] Liu, Q., Ullah, S., Zhang, C. (2011). An improved genetic algorithm for robust permutation flowshop scheduling, *The International Journal of Advanced Manufacturing Technology*, Vol. 56, No. 1-4, 345-354, doi: 10.1007/s00170-010-3149-6.

[8] Vallada, E., Ruiz, R., Minella, G. (2008). Minimising total tardiness in the *m*-machine flowshop problem: A review and evaluation of heuristics and metaheuristics, *Computers & Operations Research*, Vol. 35, No. 4, 1350-1373, doi: 10.1016/j.cor.2006.08.016.

[9] Yenisey, M.M., Yagmahan, B. (2014). Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends, *Omega*, Vol. 45, 119-135, doi: 10.1016/j.omega.2013.07.004.

[10] Sabuncuoglu, I., Goren, S. (2009). Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research, *International Journal of Computer Integrated Manufacturing*, Vol. 22, No. 2, 138-157, doi: 10.1080/09511920802209033.

[11] Yang, J., Yu, G. (2002). On the robust single machine scheduling problem, *Journal of Combinatorial Optimization*, Vol. 6, No. 1, 17-33, doi: 10.1023/A:1013333232691.

[12] Liu, L., Gu, H.-Y., Xi, Y.-G. (2007). Robust and stable scheduling of a single machine with random machine breakdowns, *The International Journal of Advanced Manufacturing Technology*, Vol. 31, No. 7-8, 645-654, doi: 10.1007/s00170-005-0237-0.

[13] Goren, S., Sabuncuoglu, I. (2008). Robustness and stability measures for scheduling: Single-machine environment, *IIE Transactions,* Vol. 40, No. 1, 66-83, doi: 10.1080/07408170701283198.

[14] Daniels, R.L., Carrillo, J.E. (1997). β-Robust scheduling for single-machine systems with uncertain processing times, *IIE transactions*, Vol. 29, No. 11, 977-985, doi: 10.1080/07408179708966416.

[15] Kuo, C.-Y., Lin, F.-J. (2002). Relative robustness for single-machine scheduling problem with processing time uncertainty, *Journal of the Chinese Institute of Industrial Engineers*, Vol. 19, No. 5, 59-67, doi: 10.1080/10170660209509359.

[16] Jia, Z., Ierapetritou, M.G. (2007). Generate Pareto optimal solutions of scheduling problems using normal boundary intersection technique, *Computers & Chemical Engineering*, Vol. 31, No. 4, 268-280, doi: 10.1016/j.compchemeng.2006.07.001.

[17] Wu, C.W., Brown, K.N., Beck, J.C. (2009). Scheduling with uncertain durations: Modeling β-robust scheduling with constraints, *Computers & Operations Research*, Vol. 36, No. 8, 2348-2356, doi: 10.1016/j.cor.2008.08.008.

[18] Fazayeli, M., Aleagha, M.-R., Bashirzadeh, R., Shafaei, R. (2016). A hybrid meta-heuristic algorithm for flowshop robust scheduling under machine breakdown uncertainty, *International Journal of Computer Integrated Manufacturing*, Vol. 29, No. 7, 709-719, doi: 10.1080/0951192X.2015.1067907.

[19] Gholami-Zanjani, S.M., Hakimifar, M., Nazemi, N., Jolai, F. (2017). Robust and fuzzy optimisation models for a flow shop scheduling problem with sequence dependent setup times: A real case study on a PCB assembly company, *International Journal of Computer Integrated Manufacturing*, Vol. 30, No. 6, 552-563, doi: 10.1080/0951192X.2016.1187293.

[20] Rahmani, D. (2017). A new proactive-reactive approach to hedge against uncertain processing times and unexpected machine failures in the two-machine flow shop scheduling problems, *Scientia Iranica*, Vol. 24, No. 3, 1571-1584, doi: 10.24200/sci.2017.4136.

[21] Cui, W., Lu, Z., Li, C., Han, X. (2018). A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops, *Computers & Industrial Engineering*, Vol. 115, 342-353, doi: 10.1016/j.cie.2017.11.020.

[22] Ma, S., Wang, Y., Li, M. (2019). A novel artificial bee colony algorithm for robust permutation flowshop scheduling, In: Li, X., Wong, K.C. (eds.), *Natural computing for unsupervised learning, Unsupervised and semi-supervised learning,* Springer, Cham, Switzerland,163-182, doi: 10.1007/978-3-319-98566-4_8.
[23] González-Neira, E.M., Urrego-Torres, A.M., Cruz-Riveros, A.M., Henao-García, C., Montoya-Torres, J.R., Molina-Sánchez, L.P., Jiménez, J.-F. (2019). Robust solutions in multi-objective stochastic permutation flow shop problem, *Computers & Industrial Engineering*, Vol. 137, Article No.106026, doi: 10.1016/j.cie.2019.106026.
[24] Lebbar, G., El Abbassi, I., Jabri, A., El Barkany, A., Darcherif, M. (2018). Multi-criteria blocking flow shop scheduling problems: Formulation and performance analysis, *Advances in Production Engineering & Management*, Vol. 13, No. 2, 136-146, doi: 10.14743/apem2018.2.279.
[25] Xu, H., Bao, Z.R., Zhang, T. (2017). Solving dual flexible job-shop scheduling problem using a Bat Algorithm, *Advances in Production Engineering & Management*, Vol. 12, No. 1, 5-16, doi: 10.14743/apem2017.1.235.
[26] Rathinam, B., Govindan, K., Neelakandan, B., Raghavan, S.S. (2015). Rule based heuristic approach for minimizing total flow time in permutation flow shop scheduling, *Tehnički Vjesnik – Technical Gazette,* Vol. 22, No. 1, 25-32, doi: 10.17559/TV-20130704132725.
[27] Ma, D.Y., He, C.H., Wang, S.Q. Han, X.M., Shi, X.H. (2018). Solving fuzzy flexible job shop scheduling problem based on fuzzy satisfaction rate and differential evolution, *Advances in Production Engineering & Management,* Vol. 13, No. 1, 44-56, doi: 10.14743/apem2018.1.272.
[28] Maqsood, S., Noor, S., Khan, M.K., Wood, A. (2012). Hybrid genetic algorithm (GA) for job shop scheduling problems and its sensitivity analysis, *International Journal of Intelligent Systems Technologies and Applications*, Vol. 11, No. 1-2, 49-62, doi: 10.1504/IJISTA.2012.046543.
[29] Khalid, Q.S,, Arshad, M., Maqsood, S., Jahanzaib, M., Babar, A.R., Khan, I., Mumtaz, J., Kim, S. (2019). Hybrid particle swarm algorithm for products' scheduling problem in cellular manufacturing system, *Symmetry*, Vol. 11, No. 6, 729, doi: 10.3390/sym11060729.
[30] Sarkar, B., Omair, M., Choi, S.-B. (2018). A multi-objective optimization of energy, economic, and carbon emission in a production model under sustainable supply chain management, *Applied Sciences*, Vol. 8, No. 10, 1744, doi: 10.3390/app8101744.
[31] Zubair, M., Maqsood, S., Omair, M., Noor, I. (2019). Optimization of material handling system through material handling equipment selection, *International Journal of Progressive Sciences and Technologies*, Vol. 15, No. 2, 235-243.
[32] Hayter, A. (2012). *Instructor solution manual, Probability and statistics for engineers and scientists,* 3rd edition, Nelson Education, Scarborough, Canada.
[33] Rechenberg, I. (1973). *Optimierung technischer systeme nach prinzipien der biologischen evolution*, Phd thesis, Technische Universität Berlin, Germany, doi: 10.1002/fedr.19750860506.
[34] Beyer, H.-G. (2001). *The theory of evolution strategies*, Natural Computing Series, Springer-Verlag, Berlin, Germany, doi: 10.1007/978-3-662-04378-3.
[35] Ahmad, A.M., Khan, G.M., Mahmud, S.A. (2013). Classification of arrhythmia types using cartesian genetic programming evolved artificial neural networks, In: Iliadis, L., Papadopoulos, H., Jayne, C. (eds.), *Engineering applications of neural networks, EANN 2013, Communications in computer and information science,* Vol. 383, Springer, Berlin, Germany, doi: 10.1007/978-3-642-41013-0_29.
[36] Paris, P.C.D., Pedrino, E.C., Nicoletti, M.C. (2015). Automatic learning of image filters using Cartesian genetic programming, *Integrated Computer-Aided Engineering*, Vol. 22, No. 2, 135-151, doi: 10.3233/ICA-150482.
[37] Murata, T., Ishibuchi, H., Tanaka, H. (1996). Genetic algorithms for flowshop scheduling problems, *Computers & Industrial Engineering*, Vol. 30, No. 4, 1061-1071, doi: 10.1016/0360-8352(96)00053-8.
[38] Grabowski, J., Pempera, J. (2007). The permutation flow shop problem with blocking. A tabu search approach, *Omega*, Vol. 35, No. 3, 302-311, doi: 10.1016/j.omega.2005.07.004.
[39] Nawaz, M., Enscore Jr., E.E., Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem, *Omega*, Vol. 11, No. 1, 91-95, doi: 10.1016/0305-0483(83)90088-9.
[40] Carlier, J. (1978). Ordonnancements à contraintes disjonctives, *RAIRO-Operations Research*, Vol. 12, No. 4, 333-350, doi: 10.1051/ro/1978120403331.
[41] Reeves, C.R. (1995). A genetic algorithm for flowshop sequencing, *Computers & Operations Research*, Vol. 22, No. 1, 5-13, doi: 10.1016/0305-0548(93)E0014-K.