Hindawi Publishing Corporation
Advances in Operations Research
Volume 2015, Article ID 124258, 12 pages
http://dx.doi.org/10.1155/2015/124258

Research Article

Scheduling Jobs Families with Learning Effect on the Setup

Sergio Fichera,[1] Antonio Costa,[1] and Fulvio Cappadonna[2]

[1]*Department of Industrial and Mechanical Engineering, University of Catania, Viale A. Doria 6, 95126 Catania, Italy*
[2]*Department of Electrical Electronic Information Engineering, University of Catania, Viale A. Doria 6, 95126 Catania, Italy*

Correspondence should be addressed to Sergio Fichera; sfichera@dii.unict.it

Received 30 September 2014; Revised 29 December 2014; Accepted 10 January 2015

Academic Editor: Konstantina Skouri

The present paper aims to address the flow-shop sequence-dependent group scheduling problem with learning effect (FSDGSLE). The objective function to be minimized is the total completion time, that is, the makespan. The workers are required to carry out manually the set-up operations on each group to be loaded on the generic machine. The operators skills improve over time due to the learning effects; therefore the set-up time of a group under learning effect decreases depending on the order the group is worked in. In order to effectively cope with the issue at hand, a mathematical model and a hybrid metaheuristic procedure integrating features from genetic algorithms (GA) have been developed. A well-known problem benchmark risen from literature, made by two-, three- and six-machine instances, has been taken as reference for assessing performances of such approach against the two most recent algorithms presented by literature on the FSDGS issue. The obtained results, also supported by a properly developed ANOVA analysis, demonstrate the superiority of the proposed hybrid metaheuristic in tackling the FSDGSLE problem under investigation.

1. Introduction

Scheduling problems have received extensive attention since the middle of the last century. One of the basic assumptions of the classic scheduling theory is that job descriptors like processing times and setup times are a priori known and do not change during the time horizon. Nowadays, due to the availability of computational resources, it is possible to consider a more realistic situation where such descriptors may vary due to the learning effect. In fact, the workers improve their performance by repeating the provided operations and, as a result, processing or/and set-up times of a job may be reduced if it is scheduled later in the sequence.

The first studies about learning effect had been developed by Wright [1] and Biskup [2]. They stated that the production time of a job under learning effect decreases depending on the order the job is worked in. The corresponding learning effect model, which computes the processing time of job J_j when it is scheduled in the r_{th} position, is defined as

$$p_{j[r]} = p_j r^a, \tag{1}$$

where p_j is the normal processing time of job J_j and $a = \log_2 \text{LR} < 0$ is the learning index, which is a function of the learning rate $\text{LR} < 1$. The processing time needed decreases by the number of repetitions, meaning that learning is primarily based on the repetition of a task, such as machine setup.

In alternative to the above position based learning model, Kuo and Yang [3] proposed a sum-of-processing-time based learning model which is time dependent:

$$p_i = p_i \left(1 + \sum_{k=1}^{r-1} p_{[k]}\right)^a. \tag{2}$$

Starting from these two fundamental models, some authors proposed other learning models that are a modification of the above models or a combination of those. Cheng et al. [4] proposed model of learning effects in which the actual processing time of a job is a function of the total normal processing times of the jobs already processed and of the job's scheduled position. Yin et al. [5] developed a general learning effects model where the actual processing time of a job is not only a function of the total normal processing times of the jobs already processed, but also a function of the job's scheduled position. Soroush [6] considers a single machine scheduling problem with general past-sequence-dependent setup time and log-linear learning in which the setup times and learning effects are job-dependent.

Biskup [7] reviews extensively the literature on scheduling problems that considers the two types of learning effects.

The learning effect has been widely applied on the single machine scheduling problems. Recently, Lee [8] proposed a model where the setup time is past-sequence-dependent, and the actual job processing time is a general function of the processing times of the jobs, already processed, and its scheduled position. Costa et al. [9] consider the single machine total weighted completion time scheduling problem. The jobs have nonzero release time and processing time increases during the production due to the effect of deterioration on the machine. The setup time and the removal time are influenced by the ability of the worker, which depends on work experience and learning capacity: Zhang [10] studied a single machine model where the jobs are grouped in families and the learning effect influences both processing time of the jobs and set-up time of the groups.

Some authors applied the principle of learning effect to other manufacturing models. Vahedi-Nouri et al. [11] investigated a nonpermutation flow shop scheduling problem with the objective of minimizing the total flow time. Each job has nonzero release time and the processing time depends on its position in the sequence, because of the learning effect. Liu [12] studied the scheduling problems of jobs on identical parallels machines where delivery times are past-sequence-dependent and the learning effect on processing times is considered. Sun et al. [13] studied a permutation flow shop scheduling, where the actual processing time of a job is defined by a general nonincreasing function of its scheduled position. To solve this problem, several algorithms derived from the corresponding single machine scheduling problem are presented.

To the best of our knowledge, the learning effect has not been investigated so far in the context of flow shop where the jobs are grouped in families. This scheduling problem is known as flow-shop group scheduling (FSGS) problem. A set of N jobs has to be processed through M serial machines arranged in a flow-shop layout. According to group technology (GT) manufacturing principles, the whole set of jobs to be worked may be partitioned into G smaller subsets, called groups or families, made by jobs sharing the same technological requirements in terms of tooling and setups. A major setup is required when switching from one family to another, whilst the setup time between jobs belonging to the same family either is assumed to be negligible or it can be included along with the run time. The solution to the problem is represented by the permutation of the jobs of each family and the permutation order of the groups. In a more realistic context as printed circuit board (PCBs) manufacturing [14, 15] or automotive sector [16] the set-up time depends on the sequence of the group in the schedule, and this problem is addressed as flow-shop sequence-dependent group scheduling (FSDGS).

The FSDGS problem has been taken into consideration by a significant number of researchers: Zhu and Wilhelm [17] proposed a complete review. Salmasi et al. [18] presented hybrid ant colony optimization (HACO) algorithm for minimizing makespan in a flow-shop sequence-dependent group scheduling problem. Hajinejad et al. [19] proposed a hybrid particle swarm optimization (PSO) algorithm that outperforms the HACO proposed by Salmasi et al. [18]. Recently, Naderi and Salmasi [20] proposed two different MILP formulations, together with a metaheuristic technique hybridising genetic and simulated annealing algorithm, called GSA, to cope with the FSDGS issue under the total completion time minimization viewpoint.

The aim of this paper is to propose the flow-shop sequence-dependent group scheduling problem with learning effect (FSDGSLE). The objective function to be minimized is the total completion time, that is, the makespan. In this scheduling problem the workers are required to manually carry out the set-up operations on each group to be loaded on the generic machine. An anticipatory set-up is assumed; that is, the set-up of each group can be started by a worker even though the first job of the group is not yet available on the machine to be set-up. The workers are not a critical resource: that is, a worker from the crew is always available to perform a set-up activity on a machine, thus preventing machine starvation and consequent delays in the makespan. The ability of operators increases over time due to the learning effect; therefore the set-up time of a group under learning effect decreases according to the order the group is worked in. The jobs processing time does not change because they are automatically worked on every machine of the flow shop. A new mathematical model for this problem and an efficient heuristic algorithm to solve also large-sized problems are proposed.

The remainder of the paper is organized as follows: Section 2 presents the mixed integer programming mathematical model for the proposed FSDGSLE problem. Section 3 describes the proposed heuristic algorithm based on the evolutionary principle. Section 4 deals with an extensive comparison between the proposed optimization procedure and other two algorithms in the field of FSDGS problems. Finally, Section 5 concludes the paper.

2. The FSDGSLE Mathematical Model

The proposed mathematical model integrates the learning effect on the group setup time with the flow-shop sequence-dependent group scheduling problem (FSDGS) by means of a mixed integer linear programming (MILP) approach. According to the formalization proposed in Pinedo [15], this problem can be denoted as: $F_m \mid fmls, Splm, prmu, Splm = Splm\, r^a \mid C_{\max}$, where F_m indicates a flow-shop with m machines, $fmls$ indicates that the jobs are assigned to different groups, $Splm$ means that the set-up of each group is sequence-dependent, $prmu$ refers to a permutation type process (i.e., all the jobs and the groups are processed by respecting the same order on each machine), and $Splm = Splm\, r^a$ means that the setup time depends on the position of the group in the sequence order of the groups, while $C_{\max}$, that is, the makespan, is the objective to be minimized.

The following notation has been adopted where a slot is a position of the sequence of groups that should be occupied

just by a single group; that is, each group should be assigned to one slot.

Indices/Parameters

M: number of machines,

G: number of groups,

$k, l, p = 0, 1, \ldots, G$: indexes of groups,

N_l: number of jobs in group l,

$i = 0, 1, \ldots, G$: index of group slots,

$j = 1, 2, \ldots, N_l$: index of jobs in group l,

$r = 1, 2, \ldots, N_l$: index of job slots in group l,

$m = 1, 2, \ldots, M$: index of machines,

P_{ljm}: processing time of job j of group l on machine m,

S_{plm}: setup time of group l preceded by group p on machine m,

a: learning index,

B: a big number.

Binary Variables. Consider

$$W_{ipl} \begin{cases} 1 & \text{if group } p \text{ is assigned to slot } i \\ & \text{and immediately precedes group } l \\ 0 & \text{otherwise,} \end{cases}$$

$$i = 0, 1, \ldots, G-1, \quad p = 0, 1, \ldots, G, \quad l = 1, 2, \ldots, G, \tag{3}$$

$$X_{ljr} \begin{cases} 1 & \text{if job } j \text{ occupies slot } r \text{ in group } l \\ 0 & \text{otherwise,} \end{cases}$$

$$l = 1, 2, \ldots, G, \quad j = 1, 2, \ldots, N_l, \quad r = 1, 2, \ldots, N_l.$$

Continuous Variables

C_{lrm}: completion time of job processed in slot r of group l on machine m,

ST_{lm}: starting time of group l on machine m,

F_{lm}: finishing time of group l on machine m,

$C_{\max}$: makespan.

Objective. Consider

$$\text{minimize } C_{\max} \tag{4}$$

subject to

$$\sum_{p=0}^{G} \sum_{l=1}^{G} W_{ipl} = 1, \quad i = 0, 1, \ldots, G-1, \tag{5}$$

$$\sum_{i=0}^{G-1} \sum_{p=0}^{G} W_{ipl} = 1, \quad l = 1, 2, \ldots, G, \tag{6}$$

$$\sum_{i=0}^{G-1} \sum_{l=1}^{G} W_{ipl} \le 1, \quad p = 0, 1, \ldots, G, \tag{7}$$

$$\sum_{p=0}^{G} W_{(i-1)pl} = \sum_{k=1}^{G} W_{ilk}, \quad i = 1, 2, \ldots, G-1, \quad l = 1, 2, \ldots, G, \tag{8}$$

$$\sum_{j=1}^{N_l} X_{ljr} = 1, \quad l = 1, 2, \ldots, G, \ r = 1, 2, \ldots, N_l, \tag{9}$$

$$\sum_{r=1}^{N_l} X_{ljr} = 1, \quad l = 1, 2, \ldots, G, \ j = 1, 2, \ldots, N_l, \tag{10}$$

$$\mathrm{ST}_{lm} \ge F_{pm} + \sum_{i=0}^{G-1} W_{ipl} \cdot S_{plm} \cdot (i+1)^a - B \cdot \left(1 - \sum_{i=0}^{G-1} W_{ipl}\right), \quad p = 0, 1, \ldots, G, \quad l = 1, 2, \ldots, G, \quad m = 1, 2, \ldots, M, \tag{11}$$

$$C_{l1m} \ge ST_{lm} + \sum_{j=1}^{N_l} X_{lj1} \cdot P_{ljm}, \quad l = 1, 2, \ldots, G, \quad m = 1, 2, \ldots, M, \tag{12}$$

$$C_{lrm} \ge C_{l(r-1)m} + \sum_{j=1}^{N_l} X_{lj1} \cdot P_{ljm}, \tag{13}$$

$$l = 1, 2, \ldots, G, \quad r = 2, 3, \ldots, N_l, \quad m = 1, 2, \ldots, M,$$

$$C_{lrm} \ge C_{lr(m-1)} + \sum_{j=1}^{N_l} X_{lj1} \cdot P_{ljm}, \tag{14}$$

$$l = 1, 2, \ldots, G, \quad r = 1, 2, \ldots, N_l, \quad m = 2, 3, \ldots, M,$$

$$F_{lm} \ge C_{lN_lm}, \quad l = 1, 2, \ldots, G, \quad m = 1, 2, \ldots, M, \tag{15}$$

$$F_{0m} = 0, \quad m = 1, 2, \ldots, M, \tag{16}$$

$$C_{\max} \geq F_{lM}, \quad l = 1, 2, \ldots, G, \tag{17}$$

$$W_{ipl}, X_{ljr} \in \{0; 1\}. \tag{18}$$

Constraints (5), (6), and (7) ensure that each group is assigned to only one slot, being preceded by one and only one group and preceding one other group at most. Constraint (8) states that if group l is assigned to slot i, being $i < G$, there must be a group k preceded by group l. Constraints (9) and (10) ensure that each job of a given group is assigned to one only slot and that each slot does not hold more than one job. Constraint (11) defines the starting time of each group on the basis of the finishing time of the previous one. Constraint (12) ensures that the first job of each group starts after the starting point of the group. Constraint (13) forces each job to be processed after the previous job of the same group is finished. Constraint (14) states that each job must be processed after it is concluded on the previous machine. Constraint (15) links the finishing time of the group to the completion time of the last job processed in the group itself. Constraint (16) forces the finishing time of dummy group 0 to be null. Constraint (17) defines the makespan.

3. The Proposed Genetic Hybrid Algorithm

The FSDGSLE problem is $\mathcal{NP}$-hard and the exact solution of the mathematical model can be reached for problems with a limited number of groups. For this reason a metaheuristic algorithm based on evolutionary principle has been proposed. Starting from a genetic algorithm properly adapted to a group scheduling problem, the proposed genetic hybrid algorithm aims to enhance the efficiency of the genetic procedure by embedding a local search algorithm.

Generally, a genetic algorithm works with a set of problem solutions called *population*. At every iteration, a new population is generated from the previous one by means of two operators, *crossover* and *mutation*, applied to solutions (*chromosomes*) selected on the basis of their *fitness*, that is, the objective function value; thus, best solutions have greater chances of being selected. Crossover operator generates new solutions (*offspring*) by coalescing the structures of a couple of existing ones (*parents*), while mutation operator brings a change into the scheme of selected chromosomes, with the aim of avoiding that the procedure remains trapped in the local optima. The algorithm proceeds by evolving the population through successive *generations*, until a given stopping criterion is reached.

Whenever a real problem should be addressed through an evolutionary algorithm, the choice of a proper *encoding* scheme (i.e., the way a solution is represented by a string of *genes*) plays a key role under both the quality of solutions and the computational burden viewpoints [21]. In addition, a valid *decoding* procedure able to transform a given string into a feasible solution should be provided.

The following subsections provide a detailed description of the proposed GA-based optimization procedure, named GHA.

3.1. Problem Encoding. Problem encoding is the way a given problem to be optimized through a metaheuristic procedure can be represented by means of a numerical chromosome. With reference to the proposed GHA, a matrix-encoding scheme has been employed. Following the same notation adopted in Section 2 and indicating by n_k the number of jobs within each group k ($k = 1, 2, \ldots, G$), each solution is described by a $(G + 1) \times n_{\max}$ matrix, being $n_{\max} = \max_{k=1,\ldots,G}\{n_k\}$. The first G rows are made by the permutation vectors π^k indicating the sequence of jobs within each group k, while the last row is the permutation vector Ω representing the sequence of groups to be processed:

$$\left[\begin{array}{c} \pi_1^1, \ldots, \pi_{n_1}^1 \\ \vdots \\ \pi_1^k, \ldots, \pi_{n_k}^k \\ \vdots \\ \pi_1^G, \ldots, \pi_{n_G}^G \\ \hline \Omega_1, \ldots, \Omega_G \end{array}\right]. \tag{19}$$

Each row r ($r = 1, 2, \ldots, G$) of the partitioned matrix codes a specific schedule concerning the problem in hand and it is worth pointing out that it is independent from the other sequences; hereinafter it will be denoted as a *subchromosome*. Thus, a certain subchromosome r corresponds to the sequence of jobs scheduled within group r; subchromosome $r = G + 1$ identifies the sequence Ω of groups. For the sake of clarity, a feasible solution for a problem in which $G = 5$ and $n_{\max} = 5$ could be represented by the following chromosome $[C_1]$:

$$[C_1] = \left[\begin{array}{ccccc} 3 & 1 & 2 & 0 & 0 \\ 2 & 5 & 1 & 4 & 3 \\ 2 & 1 & 0 & 0 & 0 \\ 3 & 4 & 1 & 2 & 0 \\ 1 & 2 & 3 & 0 & 0 \\ \hline 5 & 1 & 3 & 2 & 4 \end{array}\right]. \tag{20}$$

Subchromosomes from 1 to 5 hold the schedules of jobs within each group (i.e., schedule 3-1-2 for group 1, schedule 2-5-1-4-3 for group 2, schedule 2-1 for group 3, schedule 3-4-1-2 for group 4, and schedule 1-2-3 for group 5); subchromosome $r = G + 1$ fixes the sequence of groups Ω = 5-1-3-2-4. All the digits equal to zero do not take part either in the solution decoding or in the genetic evolutionary process. Once the problem encoding is defined, the fitness function of N_{pop} individuals pertaining to the genetic population may be computed.

3.2. Crossover Operator. Through crossover operator, the genetic material of two properly selected parent chromosomes is recombined in order to generate two offspring. The selection mechanism employed by the proposed GHA is the well-known roulette wheel scheme [22], which assigns to each solution a probability of being selected inversely proportional to the makespan value. Once two parent chromosomes have

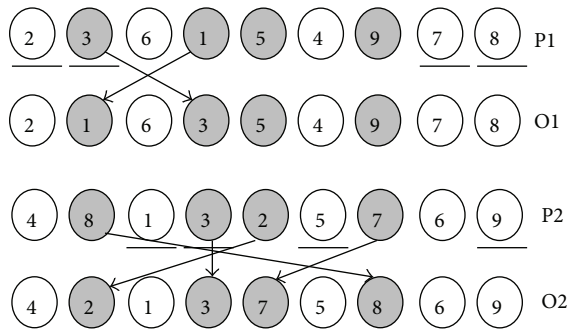

Figure 1: Position based crossover (PBC).

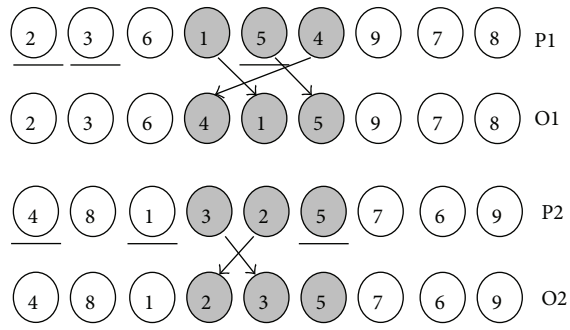

Figure 2: Two-point crossover (TPC).

been selected, each couple of subchromosomes belonging to the parent solutions is selected for crossover according to an a priori fixed probability, hereinafter called p_{cr}. Two methods of crossover operators have been adopted to recombine alleles within each couple of subchromosomes: they are denoted as *position based crossover* (PBC) and *two-point crossover* (TPC), respectively. Both of these two crossover operators have been largely adopted by literature within GAs applied to combinatorial problems [23]; PBC generates offspring by considering the relative order in which some alleles are positioned within the parents. Indeed, it works on a couple of subchromosomes (P1) and (P2) as follows: (1) one or more alleles are randomly selected; (2) the alleles genetic information of *parent 1* (P1) are reordered in *offspring 1* (O1) in the same order as they appear within *parent 2* (P2); (3) remaining elements are positioned in the sequence by copying directly from *parent 1* the unselected alleles. The same procedure is followed in the second parent, that is, *parent 2*, to obtain offspring (O2). Figure 1 shows the implementation of the PBC on a couple of parents where alleles in positions {2}, {4}, {5}, and {7} have been selected. As far as the TPC method is concerned, two positions are randomly selected and each subchromosome parent is divided into three blocks of alleles: both head and tail blocks are copied directly in the corresponding offspring, while the alleles belonging to the middle block are reordered within the offspring in the same order as they appear in the other parent subchromosome (see Figure 2). A "fair coin toss" probability equal to 0.5 has been chosen for selecting either PBC or TPC crossover.

3.3. Mutation Operator. After a new population has been generated from the previous one by means of crossover, mutation operator is applied according to an a priori fixed probability p_m. Whether mutation occurs, a chromosome is randomly chosen from the population; within such chromosome, a subchromosome is randomly selected for mutation. Two kinds of operators have been adopted in the present research: an *allele swapping operator* (ASO), which performs an exchange of two randomly selected alleles of the subchromosome, and a *block swapping operator* (BSO), which performs a block exchange (see Figure 3). To avoid any loss of the current best genetic information, the survival of the two current fittest individuals within the population is ensured by an elitist strategy.

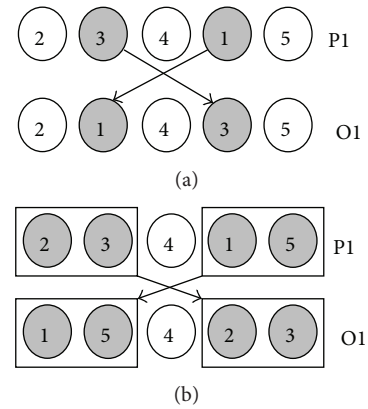

Figure 3: (a) Allele swapping (ASO) and (b) block swapping (BSO) mutation operators.

A "fair coin toss" probability equal to 0.5 has been chosen for selecting either ASO or BSO mutation operator.

3.4. Diversity Operator. A population diversity control technique has been embedded within the proposed optimization procedure, in order to mutate those identical chromosomes exceeding a preselected value D_{max}. In the present research, a D_{max} value equal to 2 has been selected, thus avoiding having more than two identical solutions within the current population.

3.5. Local Search and Termination Rule. In order to improve the performances of the proposed metaheuristic procedure, a local search algorithm has been embedded within the evolutionary optimization strategy of the proposed GHA. Such procedure operates only on a subpopulation, having size N_{best}, of the best individuals obtained after each generation. For each selected individual C_s ($s = 1, 2, \ldots, N_{best}$), a sample of N_{LS} neighbour solutions is generated by modifying the sequence Ω of groups, that is, the subchromosome $r = G + 1$, pertaining to C_s. In the present research N_{LS} has been set equal to 4. In the new sequence each group is casually ordered, each group is drawn with different probability depending on the initial position. The new sequence replaces the starting sequence if it leads to a better makespan value. Such procedure is executed for all the N_{best} individuals

Table 1: Benchmark of instances for the FDSGSLE problem.

Factor	Level	Value
Number of groups	1	$U[1, 5]$
	2	$U[6, 10]$
	3	$U[11, 16]$
Number of jobs in a group	1	$U[2, 4]$
	2	$U[5, 7]$
	3	$U[8, 10]$
Number of machines	1	2
	2	3
	3	6

originally selected. Then, the newly obtained population undergoes the next generation cycle.

The termination rule of the proposed GHA consists in $N \cdot M$ seconds of CPU time, similarly being done by Naderi and Salmasi [20].

4. Computational Experiments and Results

In order to evaluate the efficiency of the proposed GHA algorithm a benchmark of problems has been generated using the scheme provided by Salmasi et al. [18]. Processing times of each job on each machine has been randomly drawn from the range [1, 20]. The instances are generated according to three factors varied to three levels, namely, the number of groups, the number of machines, and the number of jobs within each group. The levels of setup times are three in the case of problems with 2 and 6 machines and nine in the case of problems with 3 machines. The value corresponding on each level of the benchmark is showed in Tables 1 and 2, where symbol $U[a, b]$ denotes a value extracted by a uniform distribution between a and b.

Two distinct replicates have been randomly generated for each problem of the proposed benchmark.

Therefore, a total of 54 (two machines) + 162 (three machines) + 54 (six machines) = 270 separate instances have been created.

The results of the proposed GHA have been compared with those obtained solving the mathematical model and two effective algorithms from the relevant literature on the flow shop group scheduling. The first algorithm is a hybrid particle swarm optimization (hereinafter coded as PSH) devised by Salmasi et al. [18]. Such method employs a real number matrix-encoding scheme and makes use of a properly developed transformation procedure able to convert the components of each solution to integer numbers, so as to obtain the sequence of groups and jobs within groups to be scheduled. Furthermore, the authors equipped the algorithm with a neighborhood search strategy, called individual enhancement, aimed to enhance the search by balancing exploration and exploitation power.

The second algorithm is a metaheuristic procedure hybridizing genetic and simulated annealing algorithms (hereinafter coded as GSA) proposed by Naderi and Salmasi [20]. Similar to the proposed GHA, such algorithm works with an integer number matrix-encoding scheme. It employs a twofold optimization strategy: a genetic algorithm is used to find the sequence of groups, while a simulated annealing-based local search engine drives the search towards better job sequences.

The overall set of instances has been solved by means of the three optimization procedures to be compared, namely, GHA, PSH, and GSA. All the heuristic algorithms has been coded in MATLAB language; the mathematical model has been solved through MILP solver into ILOG CPLEX commercial tool. The optimization procedures are executed on a 2 GB RAM virtual machine embedded on a workstation powered by two quad-core 2.39 GHz processors. The stopping criterion was set to $N \cdot M$ seconds of CPU time for all algorithms tested. The MILP solver has been stopped to 3,600 sec of CPU time: the exact solutions are those obtained before this time and correspond to a subset of 130 instances, in particular the first 30 instances of the problems with 2 and 6 machines and the first 70 instances of the problems with 3 machines.

The values of the learning effects are 90% and 80%, which correspond to a learning index of −0.152 and −0.322 according to Biskup's [2] model, respectively. Thus, a total of (270 (instances) × 3 (algorithms) + 130 (mathematical model)) × 2 (learning index) = 1880 runs have been taken into account. The key performance indicator used to compare the alternative metaheuristics is the relative percentage deviation (RPD), calculated as follows:

$$\mathrm{RPD} = 100 \cdot \frac{\mathrm{ALG}_{\mathrm{sol}} - \mathrm{BEST}_{\mathrm{sol}}}{\mathrm{BEST}_{\mathrm{sol}}}, \tag{21}$$

where $\mathrm{ALG}_{\mathrm{sol}}$ is the makespan solution provided by a given algorithm with reference to a certain instance and $\mathrm{BEST}_{\mathrm{sol}}$ is the exact solution or the lowest makespan value among those obtained by the provided optimization procedures.

Before starting the experimental design a preliminary tuning of the GHA algorithm was performed to determine the optimal parameter configuration. Table 3 reports the selected values.

Tables 4, 5, and 6 show the numerical results by the tested algorithms in terms of RPD values, for 2, 3, and 6 machines, respectively. Notably, every table refers to a given number of machines and two levels of learning effect equal to −0.152 and −0.323, respectively. The bold values indicate the minimum value reached by the algorithms. Both italic and bold values denote the aforementioned $\mathrm{BEST}_{\mathrm{sol}}$. If the minimum value is obtained by only one algorithm than it is underlined.

In the bottom of each table are reported four performance indicators to evaluate the effectiveness of the algorithms:

(i) $\mathrm{RPD}_{\mathrm{average}}$ is the average value of all RPDs;

(ii) N_{opt} denotes the number of times each optimization procedure reaches the best solution;

(iii) N_{exact} represents the number of times an algorithm reaches the global minimum;

Table 2: Benchmark of instances for the FDSGSLE problem.

Factor	Level	Value					
Setup times of machine M_i	1	$M_1 \to U[1,50]$	$M_2 \to U[17,67]$				
	2	$M_1 \to U[1,50]$	$M_2 \to U[1,50]$				
	3	$M_1 \to U[17,67]$	$M_2 \to U[1,50]$				
Factor	Level	Value					
Setup times of machine M_i	1	$M_1 \to U[1,50]$	$M_2 \to U[17,67]$	$M_3 \to U[45,95]$			
	2	$M_1 \to U[17,67]$	$M_2 \to U[17,67]$	$M_3 \to U[17,67]$			
	3	$M_1 \to U[45,95]$	$M_2 \to U[17,67]$	$M_3 \to U[1,50]$			
	4	$M_1 \to U[1,50]$	$M_2 \to U[17,67]$	$M_3 \to U[17,67]$			
	5	$M_1 \to U[1,50]$	$M_2 \to U[17,67]$	$M_3 \to U[1,50]$			
	6	$M_1 \to U[17,67]$	$M_2 \to U[17,67]$	$M_3 \to U[45,95]$			
	7	$M_1 \to U[17,67]$	$M_2 \to U[17,67]$	$M_3 \to U[1,50]$			
	8	$M_1 \to U[45,95]$	$M_2 \to U[17,67]$	$M_3 \to U[45,95]$			
	9	$M_1 \to U[45,95]$	$M_2 \to U[17,67]$	$M_3 \to U[17,67]$			
Setup times of machine M_i	1	$M_1 \to U[1,50]$	$M_2 \to U[17,67]$	$M_3 \to U[45,95]$	$M_4 \to U[92,142]$	$M_5 \to U[170,220]$	$M_6 \to U[300,350]$
	2	$M_1 \to U[1,50]$	$M_2 \to U[1,50]$	$M_3 \to U[1,50]$	$M_4 \to U[1,50]$	$M_5 \to U[1,50]$	$M_6 \to U[1,50]$
	3	$M_1 \to U[300,350]$	$M_2 \to U[170,220]$	$M_3 \to U[92,142]$	$M_4 \to U[45,95]$	$M_5 \to U[17,67]$	$M_6 \to U[1,50]$

Table 3: Parameters of GHA.

Parameter	Notation	Value
Population size	N_{pop}	70
Crossover probability	p_{cr}	0.9
Mutation probability	p_m	0.1
Number of individuals subjected to local search procedure	N_{best}	$0.3 \cdot N_{pop}$,

(iv) N_{best} represents the number of times the algorithm is the best one among the three metaheuristics.

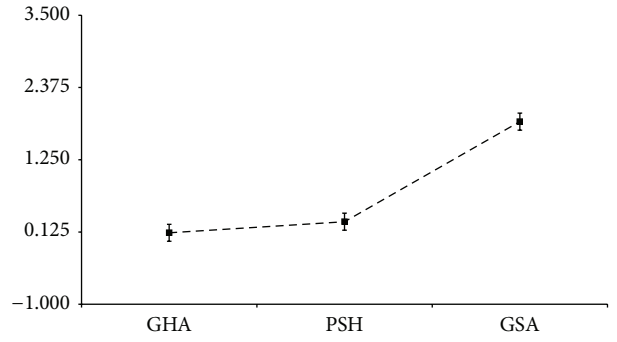

Figure 4: Means plot with 95% LSD intervals obtained for 2-machine problems.

The results on Table 4, which refer to the benchmark with 2 machines, show the effectiveness of both GHA and PSH in solving the problem at hand, with a superiority of the proposed genetic hybrid algorithm. In fact, when the learning effect is −0.152, GHA assures the lowest average $RPD_{average}$ equal to 0.12 and reaches the best solution in the 85% of the test cases. The 25% of the GHA solutions match the absolute minima of the problem and GHA is the best among the three algorithms in the 24% of times. The performances between GHA and PSH are rather comparable when the learning effect is −0.323, with exception of the $RPD_{average}$ indicator, as the PSH algorithm reaches 0.25 while GAH reaches 0.35.

Tables 5 and 6, which, respectively, hold the results from the benchmark with 3 machines and 6 machines, confirm the same findings discussed before: GHA and PSH perform significantly better than GSA, and GHA outperforms PSH under all the performance indicators viewpoints.

A slight difference among the three algorithms can be observed in terms of N_{exact} as all of them can easily reach the exact solution due to the small dimension of the solution space.

The learning effect strongly affects the $RPD_{average}$ indicator, particularly in the benchmark with 3 machines; the other measures of performance seem to be less influenced by that.

In order to infer statistical conclusion regarding the observed differences among the tested algorithms, an ANOVA analysis has been performed through Design Expert 7.0.0 version commercial tool, calculating LSD intervals at 95% confidence level for the $RPD_{average}$ performance measure connected to each optimization procedure. The analysis has been carried out for each scenario problem at varying machines, considering the mean of the $RPD_{average}$ values. The corresponding charts are reported in Figures 4, 5, and 6.

All the charts clearly show the superiority of GHA and PSH algorithms compared to GSA. In Figure 4, when 2 machines are taken into account, even though the average

Table 4: Results of 2 machines benchmarks.

Instances	2 machines					
	−0.152 learning effect			−0.323 learning effect		
	PSH	GSA	GHA	PSH	GSA	GHA
1	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
2	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
3	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
4	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
5	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
6	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
7	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
8	***0.00***	***0.00***	***0.00***	0.32	***0.00***	***0.00***
9	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
10	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
11	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
12	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
13	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
14	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
15	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
16	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
17	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
18	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
19	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
20	***0.00***	1.54	***0.00***	0.73	2.25	***0.00***
21	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
22	1.55	***0.00***	***0.00***	***0.00***	0.55	***0.00***
23	0.70	1.26	***0.00***	0.76	1.69	***0.00***
24	***0.00***	3.10	***0.00***	***0.00***	5.10	***0.00***
25	**0.42**	3.05	**0.42**	0.88	1.92	**0.35**
26	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
27	***0.00***	0.57	***0.00***	***0.00***	0.64	***0.00***
28	1.71	2.94	***0.00***	0.87	2.43	***0.00***
29	0.59	2.82	**0.34**	***0.00***	1.29	0.08
30	***0.00***	3.97	***0.00***	***0.00***	1.18	***0.00***
31	**0.00**	1.20	**0.00**	**0.00**	**0.00**	**0.00**
32	**0.00**	2.32	0.37	**0.00**	0.35	**0.00**
33	**0.00**	1.14	**0.00**	**0.00**	**0.00**	**0.00**
34	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
35	**0.00**	1.23	0.38	**0.00**	0.95	**0.00**
36	0.25	1.33	**0.00**	**0.00**	0.81	**0.00**
37	1.81	11.01	**0.00**	1.70	7.58	**0.00**
38	**0.00**	8.97	0.78	**0.00**	4.73	1.15
39	3.75	9.40	**0.00**	2.15	8.66	**0.00**
40	0.16	4.44	**0.00**	**0.00**	2.64	2.02
41	1.33	2.15	**0.00**	0.37	2.48	**0.00**
42	**0.00**	4.43	1.22	**0.00**	4.40	0.02
43	0.17	2.90	**0.00**	0.24	**0.00**	0.55
44	0.11	7.70	**0.00**	1.70	3.92	**0.00**
45	0.62	6.74	**0.00**	0.61	4.82	**0.00**
46	1.82	8.72	**0.00**	0.19	2.75	**0.00**
47	**0.00**	3.83	0.20	0.56	3.06	**0.00**
48	**0.00**	6.06	1.60	**0.00**	1.25	1.32
49	0.38	2.76	**0.00**	**0.00**	2.80	0.66
50	**0.00**	0.05	0.84	**0.00**	0.89	0.39
51	**0.00**	2.85	0.28	1.37	3.08	**0.00**
52	1.05	3.49	**0.00**	**0.00**	2.19	0.20
53	0.24	3.99	**0.00**	0.81	3.63	**0.00**
54	2.03	3.27	**0.00**	**0.00**	2.97	0.17
$RPD_{average}$	0.35	2.21	0.12	0.25	1.50	0.13
$N_{optimum}$	69%	43%	85%	72%	46%	81%
N_{exact}	46%	41%	52%	46%	39%	52%
N_{best}	13%	0%	24%	17%	2%	24%

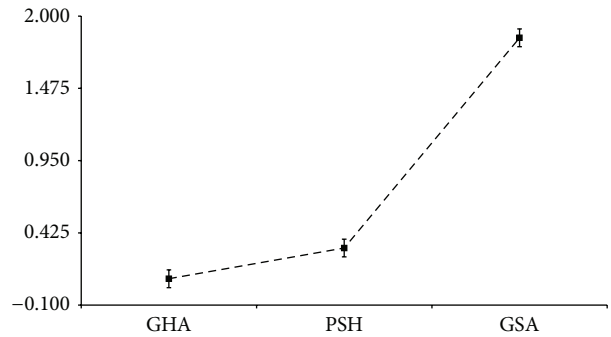

Figure 5: Means plot with 95% LSD intervals obtained for 3-machine problems.

$RPD_{average}$ related to GHA is lower than those obtained by PSH procedure, such a difference cannot be considered statistically significant, as LSD intervals of the two algorithms are overlapped. On the other hand, the narrow difference of performance between the two algorithms should depend on the poor computational complexity of the instances. As for the benchmarks with 3 and 6 machines, the superiority of the proposed GHA approach is clear from a statistical viewpoint as no overlap exists between its LSD interval and those of PSH and GSA.

5. Conclusions

In this paper the scheduling of a group of jobs in a flow shop environment managed by workforce having different levels of learning abilities is considered (FSDGSLE). The set-up times of the scheduled groups is influenced by the reciprocal position of the groups in the sequence and by the workforce ability. The jobs are automatically worked on every machine of the flow shop; therefore the processing time is not influenced by the learning effect of the operators. The objective of the scheduling is the minimization of the completion time. The operators are not a critical resource for the scheduling issue under investigation. A new

Table 5: Results of 3 machines benchmarks.

Instances	3 machines					
	−0.152 learning effect			−0.323 learning effect		
	PSH	GSA	GHA	PSH	GSA	GHA
1	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
2	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
3	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
4	***0.00***	***0.00***	***0.00***	0.78	***0.00***	***0.00***
5	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
6	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
7	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
8	0.03	***0.00***	***0.00***	0.16	***0.00***	***0.00***
9	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
10	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
11	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
12	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
13	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
14	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
15	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
16	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
17	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
18	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
19	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
20	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
21	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
22	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
23	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
24	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
25	0.22	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
26	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
27	0.27	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
28	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
29	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
30	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
31	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
32	***0.00***	1.24	***0.00***	***0.00***	1.47	***0.00***
33	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
34	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
35	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
36	***0.00***	***0.00***	***0.00***	0.61	0.19	***0.00***
37	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
38	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
39	***0.00***	***0.00***	***0.00***	0.38	***0.00***	***0.00***
40	0.43	***0.00***	0.35	0.45	***0.00***	***0.00***
41	***0.00***	0.15	***0.00***	***0.00***	***0.00***	***0.00***
42	0.17	1.52	***0.00***	**0.11**	0.16	**0.11**
43	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
44	***0.00***	***0.00***	***0.00***	**0.08**	0.13	**0.08**
45	0.93	0.08	***0.00***	***0.00***	***0.00***	***0.00***
46	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
47	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
48	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***

Table 5: Continued.

Instances	3 machines					
	−0.152 learning effect			−0.323 learning effect		
	PSH	GSA	GHA	PSH	GSA	GHA
49	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
50	0.23	***0.00***	***0.00***	0.24	***0.00***	***0.00***
51	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
52	0.26	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
53	1.79	0.65	***0.00***	0.15	***0.00***	***0.00***
54	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
55	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
56	**0.62**	5.76	2.07	0.41	3.93	***0.00***
57	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
58	0.46	3.03	***0.00***	***0.00***	0.57	***0.00***
59	**0.44**	1.91	0.57	0.70	2.62	**0.21**
60	***0.00***	4.01	***0.00***	***0.00***	2.21	***0.00***
61	0.18	1.20	***0.00***	***0.00***	5.62	0.20
62	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
63	***0.00***	1.08	***0.00***	1.12	0.30	***0.00***
64	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
65	***0.00***	1.43	***0.00***	***0.00***	***0.00***	***0.00***
66	***0.00***	0.07	***0.00***	***0.00***	***0.00***	***0.00***
67	***0.00***	1.23	***0.00***	***0.00***	***0.00***	***0.00***
68	***0.00***	2.00	***0.00***	***0.00***	***0.00***	***0.00***
69	***0.00***	1.12	***0.00***	***0.00***	1.94	***0.00***
70	**0.27**	5.67	**0.27**	0.28	2.95	***0.00***
71	***0.00***	***0.00***	***0.00***	***0.00***	0.02	***0.00***
72	***0.00***	***0.00***	***0.00***	***0.00***	0.00	***0.00***
73	***0.00***	3.24	***0.00***	**0.72**	2.39	1.48
74	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
75	0.49	1.30	***0.00***	***0.00***	0.94	***0.00***
76	0.28	0.57	***0.00***	0.44	0.71	**0.29**
77	***0.00***	3.59	***0.00***	***0.00***	1.51	0.24
78	1.54	0.44	***0.00***	***0.00***	***0.00***	***0.00***
79	0.70	0.38	***0.00***	**0.13**	1.51	**0.13**
80	***0.00***	0.21	0.21	***0.00***	1.49	0.43
81	***0.00***	2.94	1.06	***0.00***	0.41	***0.00***
82	***0.00***	0.70	***0.00***	**0.05**	**0.05**	0.44
83	**0.32**	0.54	**0.32**	***0.00***	0.27	***0.00***
84	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
85	***0.00***	1.74	***0.00***	***0.00***	1.08	***0.00***
86	**0.53**	1.01	**0.53**	0.99	**0.32**	0.38
87	1.81	0.88	**0.78**	**0.52**	0.98	0.98
88	**0.21**	1.65	0.82	1.24	1.30	**0.37**
89	**0.15**	1.10	**0.15**	***0.00***	2.67	***0.00***
90	1.24	0.15	***0.00***	***0.00***	0.57	***0.00***
91	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
92	0.69	2.50	**0.00**	0.86	1.64	**0.00**
93	**0.00**	3.68	**0.00**	***0.00***	1.00	***0.00***
94	**0.00**	0.90	0.13	**0.33**	1.39	**0.33**
95	**0.00**	**0.00**	0.24	***0.00***	***0.00***	***0.00***
96	0.13	0.51	**0.00**	0.12	***0.00***	0.12
97	**0.00**	1.28	0.54	0.34	1.10	**0.31**

Table 5: Continued.

Instances	3 machines					
	−0.152 learning effect			−0.323 learning effect		
	PSH	GSA	GHA	PSH	GSA	GHA
98	**0.00**	1.30	**0.00**	0.10	0.14	***0.00***
99	**0.00**	1.53	0.45	1.32	**0.56**	0.72
100	**0.00**	1.10	0.24	**0.50**	1.05	**0.50**
101	**0.00**	2.76	**0.00**	***0.00***	***0.00***	***0.00***
102	**0.00**	0.74	**0.00**	***0.00***	1.30	***0.00***
103	**0.00**	**0.00**	0.01	0.10	1.42	***0.00***
104	0.14	**0.00**	0.01	1.21	0.92	**0.77**
105	**0.00**	**0.00**	**0.00**	***0.00***	***0.00***	***0.00***
106	0.21	0.21	**0.00**	0.22	1.94	***0.00***
107	**0.00**	**0.00**	**0.00**	***0.00***	***0.00***	***0.00***
108	0.54	0.46	**0.00**	0.31	***0.00***	***0.00***
109	2.04	7.57	**0.00**	2.68	6.55	**0.00**
110	0.77	5.87	**0.00**	**0.00**	2.82	0.20
111	**0.00**	3.93	2.95	1.78	4.89	**0.00**
112	4.63	10.01	**0.00**	**0.00**	4.29	0.53
113	1.12	6.28	**0.00**	**0.00**	4.49	0.26
114	0.23	4.12	**0.00**	0.12	4.60	**0.00**
115	3.12	10.68	**0.00**	1.09	6.88	**0.00**
116	1.80	8.16	**0.00**	0.05	5.25	**0.00**
117	0.50	9.07	**0.00**	1.07	8.33	**0.00**
118	1.71	10.93	**0.00**	2.57	7.65	**0.00**
119	2.11	9.54	**0.00**	2.78	5.97	**0.00**
120	1.87	10.69	**0.00**	0.86	5.11	**0.00**
121	**0.00**	7.78	0.67	0.31	4.48	**0.00**
122	**0.00**	7.92	0.40	**0.00**	2.67	**0.00**
123	2.01	5.85	**0.00**	0.40	4.82	**0.00**
124	2.12	5.37	**0.00**	**0.00**	3.29	0.48
125	**0.00**	1.97	1.02	1.18	6.31	**0.00**
126	1.17	3.41	**0.00**	**0.00**	3.36	0.57
127	1.42	4.41	**0.00**	0.35	4.54	**0.00**
128	2.65	5.27	**0.00**	**0.00**	5.18	1.48
129	1.80	4.84	**0.00**	0.62	3.82	**0.00**
130	0.42	3.36	**0.00**	**0.00**	2.22	1.58
131	1.88	4.58	**0.00**	1.29	3.65	**0.00**
132	**0.00**	4.83	1.17	**0.00**	3.75	0.21
133	**0.00**	5.01	0.01	0.67	5.48	**0.00**
134	1.76	4.26	**0.00**	1.55	3.29	**0.00**
135	**0.00**	6.06	0.15	1.88	3.98	**0.00**
136	**0.00**	5.63	**0.00**	**0.00**	1.03	0.45
137	1.87	6.39	**0.00**	0.07	6.18	**0.00**
138	2.06	5.01	**0.00**	0.28	2.82	**0.00**
139	0.28	3.31	**0.00**	0.43	3.63	**0.00**
140	**0.00**	7.06	2.01	0.48	4.85	**0.00**
141	1.47	5.36	**0.00**	0.50	3.69	**0.00**
142	1.31	6.99	**0.00**	1.51	4.18	**0.00**
143	0.48	5.96	**0.00**	0.62	4.04	**0.00**
144	**0.00**	2.02	1.10	**0.00**	0.94	**0.00**
145	**0.00**	3.12	0.54	**0.00**	1.79	0.33
146	**0.00**	1.91	0.75	0.35	2.23	**0.00**

Table 5: Continued.

Instances	3 machines					
	−0.152 learning effect			−0.323 learning effect		
	PSH	GSA	GHA	PSH	GSA	GHA
147	0.06	5.13	**0.00**	0.09	3.43	**0.00**
148	1.95	2.89	**0.00**	0.38	1.59	**0.00**
149	0.66	4.62	**0.00**	0.67	4.33	**0.00**
150	1.00	5.69	**0.00**	0.26	3.67	**0.00**
151	0.31	2.84	**0.00**	0.49	1.90	**0.00**
152	**0.00**	4.31	0.03	0.99	4.62	**0.00**
153	**0.00**	2.83	0.00	**0.00**	2.73	0.90
154	0.83	5.23	**0.00**	0.89	3.24	**0.00**
155	1.18	1.00	**0.00**	0.10	0.66	**0.00**
156	0.36	0.44	**0.00**	**0.00**	0.75	0.59
157	**0.00**	2.20	0.08	**0.00**	2.46	0.06
158	**0.00**	2.07	0.00	**0.00**	2.03	0.05
159	**0.00**	2.76	0.62	0.17	2.39	**0.00**
160	0.32	3.73	**0.00**	0.46	2.80	**0.00**
161	**0.00**	4.21	0.16	0.82	3.63	**0.00**
162	**0.00**	4.54	2.18	0.33	2.86	**0.00**
$RPD_{average}$	0.40	2.15	0.14	0.28	1.59	0.10
$N_{optimum}$	65%	39%	81%	61%	44%	85%
N_{exact}	41%	35%	49%	48%	43%	55%
N_{best}	14%	1%	30%	12%	2%	29%

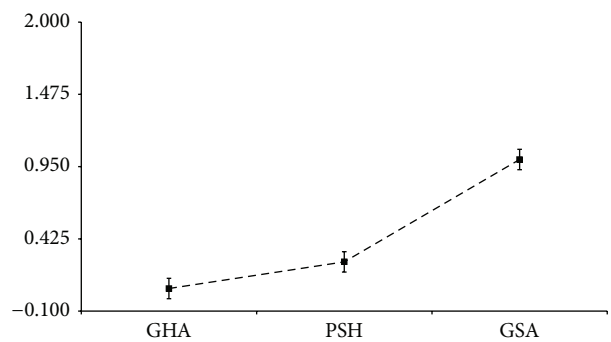

Figure 6: Means plot with 95% LSD intervals obtained for 6-machine problems.

mathematical model is considered to optimally solve small- and medium-sized instances of this FSDGSLE scheduling problem. Due to the large computational time required to cope with large-sized instances a genetic hybrid algorithm (GHA) is proposed. A comparison campaign based on three separate benchmarks risen from literature involving two-, three- and six-machine problems has been fulfilled in order to test the performance of GHA with respect to the two latest metaheuristic procedures presented by literature in the field of FSDGS scheduling problems. An ANOVA analysis focusing on a statistical validation of the obtained outcomes has been performed. The obtained numerical results highlight the effectiveness of GHA in approaching

Table 6: Results of 6 machines benchmarks.

Instances	6 machines					
	−0.152 learning effect			−0.323 learning effect		
	PSH	GSA	GHA	PSH	GSA	GHA
1	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
2	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
3	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
4	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
5	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
6	***0.00***	***0.00***	***0.00***	**0.00**	**0.00**	**0.00**
7	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
8	***0.00***	***0.00***	***0.00***	**0.00**	**0.00**	**0.00**
9	***0.00***	***0.00***	***0.00***	0.38	***0.00***	***0.00***
10	***0.00***	***0.00***	***0.00***	0.26	**0.01**	0.26
11	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
12	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
13	***0.00***	***0.00***	***0.00***	**0.00**	**0.00**	**0.00**
14	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
15	0.89	1.92	**0.08**	1.85	2.11	**0.24**
16	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
17	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
18	0.13	0.13	***0.00***	0.07	0.07	***0.00***
19	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
20	***0.00***	***0.00***	***0.00***	***0.00***	0.87	***0.00***
21	***0.00***	***0.00***	***0.00***	***0.00***	0.31	***0.00***
22	**0.23**	0.46	0.46	0.99	***0.00***	***0.00***
23	***0.00***	0.21	***0.00***	**0.09**	0.30	**0.09**
24	0.37	1.70	***0.00***	0.18	1.21	**0.00**
25	***0.00***	0.53	***0.00***	0.03	0.25	**0.00**
26	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***	***0.00***
27	0.13	3.35	***0.00***	0.41	2.49	**0.05**
28	2.01	3.75	**1.36**	0.80	3.59	***0.00***
29	0.48	0.63	**0.19**	0.37	**0.20**	**0.20**
30	0.04	0.19	***0.00***	***0.00***	0.42	***0.00***
31	**0.00**	***0.00***	**0.00**	***0.00***	0.42	***0.00***
32	**0.00**	0.31	**0.00**	**0.00**	0.73	**0.00**
33	**0.00**	0.14	**0.00**	1.75	1.38	**1.31**
34	0.87	2.35	**0.00**	0.48	2.56	***0.00***
35	0.24	0.24	**0.00**	0.41	0.18	**0.14**
36	0.53	0.50	**0.00**	**0.00**	0.47	**0.00**
37	0.90	1.62	**0.00**	0.41	2.09	**0.00**
38	**0.00**	0.72	0.04	0.06	1.32	**0.00**
39	**0.00**	5.68	2.09	0.06	1.04	**0.00**
40	0.78	5.44	**0.00**	**0.00**	5.12	2.85
41	0.17	0.95	**0.00**	**0.00**	0.54	0.03
42	0.31	1.76	**0.00**	**0.00**	1.36	0.36
43	0.10	0.85	**0.00**	**0.00**	1.05	0.36
44	0.74	1.33	**0.00**	0.28	2.12	**0.00**
45	1.92	4.38	**0.00**	1.29	4.36	**0.00**
46	**0.00**	3.58	0.48	0.95	3.03	**0.00**
47	0.07	0.15	**0.00**	0.16	1.15	**0.00**
48	0.24	1.38	**0.00**	**0.00**	1.28	**0.00**
49	0.18	0.55	**0.00**	**0.00**	1.32	0.08
50	0.29	1.07	**0.00**	**0.00**	1.07	0.08
51	0.20	2.84	**0.00**	0.34	4.01	**0.00**
52	2.57	4.69	**0.00**	1.48	3.96	**0.00**
53	1.29	1.76	**0.00**	0.39	1.87	**0.00**
54	0.23	1.61	**0.00**	0.49	1.12	**0.00**
$RPD_{average}$	0.29	1.05	0.09	0.26	1.03	0.11
$N_{optimum}$	54%	39%	93%	56%	37%	87%
N_{exact}	41%	37%	46%	41%	33%	54%
N_{best}	7%	0%	44%	11%	2%	37%

the scheduling problem at hand, regardless of the specific class of problem to be analyzed. Further research should involve other variants of the flow-shop group scheduling issue, especially inspired to real-world applications. For instance, it would be interesting to investigate manufacturing system wherein workers have different learning abilities or machines are affected by deterioration effects.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] T. P. Wright, "Factors affecting the cost of airplanes," *Journal of the Aeronautical Sciences*, vol. 3, no. 4, pp. 122–128, 1936.

[2] D. Biskup, "Single-machine scheduling with learning considerations," *European Journal of Operational Research*, vol. 115, no. 1, pp. 173–178, 1999.

[3] W.-H. Kuo and D.-L. Yang, "Minimizing the total completion time in a single-machine scheduling problem with a time-dependent learning effect," *European Journal of Operational Research*, vol. 174, no. 2, pp. 1184–1190, 2006.

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

[5] Y. Yin, D. Xu, K. Sun, and H. Li, "Some scheduling problems with general position-dependent and time-dependent learning effects," *Information Sciences*, vol. 179, no. 14, pp. 2416–2425, 2009.

[6] H. M. Soroush, "Solving the single machine scheduling problem with general job-dependent past-sequence-dependent setup times and learning effects," *European Journal of Industrial Engineering*, vol. 6, no. 5, pp. 596–628, 2012.

[7] D. Biskup, "A state-of-the-art review on scheduling with learning effects," *European Journal of Operational Research*, vol. 188, no. 2, pp. 315–329, 2008.

[8] W.-C. Lee, "Single-machine scheduling with past-sequence-dependent setup times and general effects of deterioration and learning," *Optimization Letters*, vol. 8, no. 1, pp. 135–144, 2014.

[9] A. Costa, F. A. Cappadonna, and S. Fichera, "Heuristics for single machine scheduling problem with release dates, deteriorating effect and skilled workforce with learning ability," *Journal of Advanced Manufacturing Systems*, vol. 13, no. 2, pp. 73–88, 2014.

[10] X. Zhang, "Single-machine group scheduling problems with the sum-of-processing-time based on learning effect," *Operations Research Transactions*, vol. 17, no. 1, pp. 98–105, 2013.

[11] B. Vahedi-Nouri, P. Fattahi, R. Tavakkoli-Moghaddam, and R. Ramezanian, "A general flow shop scheduling problem with consideration of position-based learning effect and multiple availability constraints," *The International Journal of Advanced Manufacturing Technology*, vol. 73, no. 5–8, pp. 601–611, 2014.

[12] M. Liu, "Parallel-machine scheduling with past-sequence-dependent delivery times and learning effect," *Applied Mathematical Modelling. Simulation and Computation for Engineering and Environmental Systems*, vol. 37, no. 23, pp. 9630–9633, 2013.

[13] L.-H. Sun, K. Cui, J.-H. Chen, J. Wang, and X.-C. He, "Research on permutation flow shop scheduling problems with general position-dependent learning effects," *Annals of Operations Research*, vol. 211, no. 1, pp. 473–480, 2013.

[14] J. E. Schaller, J. N. D. Gupta, and A. J. Vakharia, "Scheduling a flowline manufacturing cell with sequence dependent family setup times," *European Journal of Operational Research*, vol. 125, no. 2, pp. 324–339, 2000.

[15] M. L. Pinedo, *Scheduling: Theory, Algorithms and Systems*, Springer, New York, NY, USA, 4th edition, 2012.

[16] N. Salmasi, R. Logendran, and M. R. Skandari, "Total flow time minimization in a flowshop sequence-dependent group scheduling problem," *Computers & Operations Research*, vol. 37, no. 1, pp. 199–212, 2010.

[17] X. Zhu and W. E. Wilhelm, "Scheduling and lot sizing with sequence-dependent setup: a literature review," *IIE Transactions*, vol. 38, no. 11, pp. 987–1007, 2006.

[18] N. Salmasi, R. Logendran, and M. R. Skandari, "Makespan minimization of a flowshop sequence-dependent group scheduling problem," *The International Journal of Advanced Manufacturing Technology*, vol. 56, no. 5–8, pp. 699–710, 2011.

[19] D. Hajinejad, N. Salmasi, and R. Mokhtari, "A fast hybrid particle swarm optimization algorithm for flow shop sequence dependent group scheduling problem," *Scientia Iranica*, vol. 18, no. 3, pp. 759–764, 2011.

[20] B. Naderi and N. Salmasi, "Permutation flowshops in group scheduling with sequence-dependent setup times," *European Journal of Industrial Engineering*, vol. 6, no. 2, pp. 177–198, 2012.

[21] A. Costa, G. Celano, S. Fichera, and E. Trovato, "A new efficient encoding/decoding procedure for the design of a supply chain network with genetic algorithms," *Computers & Industrial Engineering*, vol. 59, no. 4, pp. 986–999, 2010.

[22] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Berlin, Germany, 2nd edition, 1994.

[23] G. Celano, A. Costa, and S. Fichera, "Constrained scheduling of the inspection activities on semiconductor wafers grouped in families with sequence-dependent set-up times," *The International Journal of Advanced Manufacturing Technology*, vol. 46, no. 5–8, pp. 695–705, 2010.