International Journal of Industrial Engineering Computations 8 (2017) 427–440

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Heterogeneous workers with learning ability assignment in a cellular manufacturing system

Sergio Fichera[a*], Antonio Costa[a] and Fulvio Antonio Cappadonna[a]

[a]*Department of Civil Engineering and Architecture, University of Catania, Viale Andrea Doria 6, 95125, Catania, Italy*

C H R O N I C L E

Article history:
Received October 27 2016
Received in Revised Format December 25 2016
Accepted March 2 2017
Available online March 3 2017

Keywords:
Flow-shop
Group scheduling
Workforce assignment
Learning effect
Skills
Evolutionary algorithm

A B S T R A C T

This paper deals with Flow-shop Sequence-Dependent Group Scheduling and worker assignment problem. Flow-shop allows the process of a set of families of products applying the group technology concept to reduce setup costs, lead times, and work-in-process inventory costs. The worker assignment problem deals with assigning workers to workstations considering their different abilities and learning effect. The proposed model in this paper considers different objectives. The decision problems in this cellular manufacturing system are the jobs scheduling within of own group, the group scheduling and the workers assignment to the machines. The aim of this paper is to consider a more realistic profile of heterogeneous workers introducing the learning effect in the joint group scheduling and workers assignment problem. A mathematical model and an evolutionary procedure has been developed to solve this problem. A benchmark of test cases having different numbers of machines, groups, jobs, worker skills and learning index, has been taken into account to compare the efficiency of the proposed algorithm with two well known procedures.

1. Introduction

Nowadays manufacturing systems have reached a high automation level in all phases of production. The human role evolved towards the control of the operation rather than the manual executions of the activity, but the high integration and automation level of the manufacturing plants is very costly and frequently not convenient for business. In this context, the workers play an important function and it is necessary to take into consideration the natural heterogeneity and the consequent different ability of each worker at shop level decisions. Typical examples of manufacturing environments where human resource is critical for the set-up activities are the cellular manufacturing system, where mechanical parts (hereinafter called jobs) are produced by CNC work centres: jobs to be processed are grouped into families due their similarity, (e.g. same/similar morphology, same/similar technological features), so that those jobs have to visit the working machines in the same order. Setup time of a single job is negligible, but setup time of a group may be significant and depends on the technological requirements of the previously processed

* Corresponding author
E-mail: sfichera@dii.unict.it (S. Fichera)

doi: 10.5267/j.ijiec.2017.3.005

group. The worker assigned to a machine manually performs the setup operations required by a given group of jobs (e.g. job fixing on the pallet, tool path and machining parameters programming). If the operations carried out by the workers are substantially repetitive then it is possible to take into account the learning effect to evaluate their durations. Furthermore, if the workers have different skills then they carry out the setup operations in diverse completion times.

The duration of the set-up group time varies with the order of the group in the sequence and the ability and experience of the worker assigned to the machine. The completion time of a given job within a given machine arises from the sum of a fixed processing time, because it is automatically processed on CNC and has a variable setup time. The decision problems in this cellular manufacturing system are the jobs scheduling within its own group, the group scheduling and the workers assignment to the machines. These kinds of problems are widely studied separately due to the high level of complexity characterizing the mixed problem. Recently, the authors (Costa et al., 2014) proposed an efficient hybrid genetic algorithm to solve the joint problems of a flow-shop group scheduling with sequence dependent set-up times and skilled workforce assignment.

The aim of this paper is to consider a more realistic profile of heterogeneous workers introducing the learning effect in the joint group scheduling and workers assignment problem.

The remainder of the paper is organized as follows: in Section 2 a review of current literature concerning the Group Scheduling, the Workforce Assignment and ability of the workers are reported. Section 3 presents the mixed integer programming mathematical model for the proposed problem. Section 4 shows the structure of the proposed Evolutionary algorithms to solve the joint problem. Section 5 addresses the evaluation of the performance of the worker assignment module and the comparisons between the proposed algorithms and two effective algorithms proposed in literature. Conclusions and future research complete the paper.

2. Literature Review

The description of the behavior of workers in the production contest is the first topic of the literature review. A pioneering research was carried out by Hunter (1986) who proposed a model to measure worker ability in learning and obtaining information from the process. This model allows a classification of the workers on the basis of individual differences, called skills, and evaluates the potential of cross-training and productivity. Fitzpatrick and Askin (2005) presented a mathematical model that exploits labor skill pools for arranging any team of workers; performance of such a team depends on individual behaviours and interpersonal interactions of workers as well as on their technical competences. Emmett et al. (2009) presented a survey on the human factor literature and constructed a framework for scheduling human tasks that accounted for physical and/or cognitive human characteristics and behaviours. The first studies about learning effect had been developed by Wright (1936). Biskup (1999) proposed that the production time of a job under learning effect decreases depending on the order the job is worked in. He introduced a learning effect model in which the processing time of job J_j when it is scheduled in the rth position in a processing sequence is defined as $p_{j[r]} = p_j r^a$ where p_j is the normal processing time of job J_j and $a = \log_2 LR < 0$ is the learning index, which is a function of the learning rate $LR<1$. 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. Biskup (2008) extensively reviewed the literature on scheduling problems that consider the learning effects.

The worker assignment problem has been mainly studied to solve the assignment of the workers to tasks to be worked or to production teams. McDonald et al. (2009) developed a model that assigns workers to tasks within a lean manufacturing cell, thus minimizing the net present cost. In determining how to assign workers to tasks, the model matches production requirements with customer demand, skill level required by tasks, quality levels based on skill levels and job rotation to retain skills for a cross-trained workforce.

Wi et al. (2009) presented a framework for analyzing the knowledge of the candidates for managers and team members for the new team, and proposes a genetic algorithm and social network measurement for choosing a team manager and team members. Agustín-Blas et al. (2011) presented a new model for team formation based on group technology principles. Dorn et al. (2011) provide two heuristics based on Genetic Algorithms and Simulated Annealing for discovering efficient team configurations that yield the best trade-off between skill coverage and team connectivity. Savino et al. (2014) addressed the problem of real time workforce assignment and scheduling in assembly lines where the number of operators is less to the number of workstations, Moreira et al. (2015) proposed the Assembly Line Worker Integration and Balancing Problem (ALWIBP). The goal of this problem is to maintain high productivity levels by minimizing the number of workstations needed to reach a given output, while integrating in the assembly line a number of disabled workers. Finally, Karthikeyan et al. (2016) proposed a genetic algorithm to optimize the worker assignment in in a cellular manufacturing system.

In a labour intensive context like service centres where task scheduling and workforce assignment play a key role for increasing their efficiency, this kind of problem is denoted as the Skilled Workforce Project Scheduling (SWPS) problem and entails constraints on task times and variable task durations, depending on the worker efficiency. Valls et al. (2009) proposed a hybrid genetic algorithm to obtain a feasible scheduling plan and a balanced and efficient assignment of workforce to tasks for a SWPS problem. Corominas et al. (2010) considered the problem of assigning and scheduling a set of tasks to a set of workers, according to which worker's performance for a given task depends on the worker experience.

In this paper, the worker assignment problem is considered in a cellular manufacturing system where the jobs grouped in families are scheduled within a flow shop production system known as a Flow Shop Group Scheduling (FSGS) problem. For this kind of problems, setup time of a single job is considered to be negligible, whereas the setup time of each group of jobs is explicitly modelled and influenced by the order of two consecutive groups. This problem is denoted as Flow Shop Dependent Group Scheduling (FSDGS). Literature of the last decades extensively dealt with both FSGS and FSDGS problems as confirmed by the following three reviews performed by Allahverdi et al. (1999), Cheng et al. (2000) and Zhu and Wilhelm (2006), respectively.

Hendizadeh et al. (2008) and Salmasi and Logedran (2008) proposed a metaheuristic algorithm based on tabu search. Celano et al. (2010) proposed an evolutionary algorithm to solve a real group scheduling problem which pertains to the inspection department of a semiconductors company; the problem was modelled as a permutational flowshop group scheduling problem with sequence dependent setup times and limited inter-operational buffer capacity. Hajinejad et al. (2011) introduced a fast hybrid particle swarm optimization algorithm whose objective to be minimized was the total flow time. Recently, Naderi et al. (2012) studied the FSDGS problem with minimization of total completion time as the criterion, they proposed two different mixed integer linear programming (MILP) models, for medium size problems and a metaheuristic hybridising genetic and simulated annealing algorithm to solve the problems heuristically.

Solving problems that mix both scheduling and planning is a tricky challenge as described by Perron (2010) who tried to solve the following problem: assembling teams of skilled workers to perform jobs that require these skills, breaking up these teams and then assembling new ones to perform more jobs.

Wanga et al. (2014) investigated flowshop scheduling problems with a general exponential learning effect, i.e., the actual processing time of a job is defined by an exponent function of the total weighted normal processing time of the already processed jobs and its position in a sequence, where the weight is a position-dependent weight. Benavides et al. (2014) proposed a mathematical model that extends a flow shop model to admit a heterogeneous worker assignment, and propose a heuristic based on scatter search and path relinking to solve the problem.

As stated in the previous paragraph the authors consider the assignment of heterogeneous workers who improve their abilities during the job production due to the learning effect. However, this joint problem involving both job sequencing and assignment of a set of workers with different skills to machines and learning effect has not been discussed by the body of literature so far.

3. Problem formulation

In this section, a formulation of the flow shop group scheduling problem with heterogeneous workers is presented. Accordingly to the formalization proposed in Pinedo (2012), this problem can be denoted as: $F_m|fmls, Spkl, Spkl= Spkl^* r^a*SL\ prmu|C_{max}$, where F_m indicates a flow-shop with m machines, *fmls* indicates that the jobs are assigned to different groups, *Spkl* means that the set-up of each group is sequence dependent, $Spkl= Spkl^* r^a*SL$ means that the setup time depends on the position of the group in the sequence order of the groups and the skill level, *SL*, of the workers, *prmu* refers to a permutation type process: (that is, all the jobs and the groups are processed by respecting the same order on each machine) while $C_{\max}$, i.e. 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:

n_0	number of jobs
M	number of machines
G	number of groups
$j = 1,2,\ldots,n_0$	index of jobs
$i = 1,2,\ldots,m$	index of machines
$q = 1,2,\ldots,m$	index of workers with different
$k,t,r = 0,1,\ldots,g$	index of groups
$s = 0,1,\ldots,g\text{-}1$	index of group slots
G_k	set of jobs belonging to group k
$n_k = \|G_k\|$	number of jobs belonging to group k
$l = 1,2,\ldots n_k$	index of job slots in group k
p_{ji}	processing time of job j on machine i
a_{tkiq}	setup time of group k performed by worker q on machine i, if group k is processed immediately after group t
β	learning index
M	a big number

Decision variables:

X_{lj}	$\begin{cases}1 \text{ if job } j \text{ is assigned to slot } l \text{ of group } k\\ 0 \text{ otherwise}\end{cases}$	$k = 1,2,\ldots,g$ $l = 1,2,\ldots,n_k$ $j \in G_k$
U_{stk}	$\begin{cases}1 \text{ if group } k \text{ is processed immediately after group } t, \text{ and group } t\\ \text{ is assigned to slot } s\\ 0 \text{ otherwise}\end{cases}$	$s = 0,1,\ldots,g\text{-}1$ $t = 0,1,\ldots,g$ $k = 1,2,\ldots,g$ $t \neq k$
Z_{iq}	$\begin{cases}1 \text{ if worker } q \text{ is assigned to machine } i\\ 0 \text{ otherwise}\end{cases}$	$i, q =$ $1,2,\ldots,m$
C_{kli}	completion time of job processed in slot l of group k on machine i	
F_{ki}	finishing time of group k on machine i	

S_{ki} starting time of group k on machine i

$C_{\max}$ Makespan

Model
minimize $C_{\max}$

Subject to:

$$\sum_{l=1}^{n_k} X_{jl} = 1 \qquad k = 1,2,...,g \quad j \in G_k \tag{1}$$

$$\sum_{j \in G_k} X_{jl} = 1 \qquad k = 1,2,...,g \quad l = 1,2,...n_k \tag{2}$$

$$C_{kli} \geq C_{k(l-1)i} + \sum_{j \in G_k} X_{jl} \cdot p_{ji} \qquad k = 1,2,...,g \quad l = 2,3,...n_k \quad i = 1,2,...,m \tag{3}$$

$$C_{kli} \geq C_{kl(i-1)} + \sum_{j \in G_k} X_{jl} \cdot p_{ji} \qquad k = 1,2,...,g \quad l = 1,2,...n_k \quad i = 2,3,...,m \tag{4}$$

$$C_{k1i} \geq S_{ki} + \sum_{j \in G_k} X_{j1} \cdot p_{ji} \qquad k = 1,2,...,g \quad i = 1,2,...,m \tag{5}$$

$$F_{ki} \geq C_{kn_k i} \qquad k = 1,2,...,g \quad i = 1,2,...,m \tag{6}$$

$$S_{ki} \geq F_{ti} + \sum_{q}^{m} Z_{iq} \cdot a_{tkiq} \cdot (s+1)^{\beta} - (1 - U_{stk}) \cdot M \qquad \begin{matrix} s = 0,1,...g-1 \quad t = 0,1,...g \\ k = 1,2,...,g \quad k \neq t \quad i = 1,2,...,m \end{matrix} \tag{7}$$

$$\sum_{t=0}^{g} \sum_{k=1}^{g} U_{stk} = 1 \qquad s = 0,1,...,g-1 \tag{8}$$

$$\sum_{s=0}^{g-1} \sum_{t=0}^{g} U_{stk} = 1 \qquad k = 1,2,...,g \tag{9}$$

$$\sum_{s=0}^{g-1} \sum_{k=1}^{g} U_{stk} \leq 1 \qquad t = 1,2,...,g \tag{10}$$

$$\sum_{t=0}^{g} U_{(s-1)tk} = \sum_{r=1}^{g} U_{skr} \qquad s = 1,2,...,g-1 \quad k = 1,2,...,g \tag{11}$$

$$\sum_{i=1}^{m} Z_{iq} = 1 \qquad q = 1,2,...,m \tag{12}$$

$$\sum_{q=1}^{m} Z_{iq} = 1 \qquad i = 1,2,...,m \tag{13}$$

$$C_{\max} \geq F_{km} \qquad k = 1,2,...,g \tag{14}$$

$$C_{kli}, S_{ki}, F_{ki} \geq 0 \tag{15}$$

$$X_{lj}, U_{stk}, Z_{iq} \in \{0;1\} \tag{16}$$

Constraint (1) ensures that each job is assigned to exactly one slot within the group it belongs to. Constraint (2) states that each slot of any given group must be occupied by only one job. Through constraint (3) it is imposed that each job cannot start before the job assigned to the previous slot of the same group has been finished. Constraint (4) forces each job to start on a given machine after it has been completed on the previous one. Constraint (5) links the completion time of the job processed as first in each group to the starting time of the group itself, while constraint (6) links the completion time of the job processed as last in each group to the finishing time of the group itself. Constraint (7) states that each group can start to be processed on a given machine after the preceding group has been completed and the setup has been performed. Skills and learning effect are involved in setup time calculation. Constraints (8), (9), (10) define precedence relationships among groups: they ensure that each slot is occupied by one group preceding only one other group; that each group is preceded by one only group in a given slot; that a group precedes at most one other group. Furthermore, through constraint (11) it is imposed that, if group k is assigned to slot s, the group assigned to slot s-1 is the predecessor of k. Constraint (12) forces each machine to be assigned to only one worker; conversely, constraint (13) assigns each worker to only one machine. Through constraint (14), the makespan is set to be equal to the highest among finishing times of all groups. Constraints (15) establishes the non-negativity of continuous variables, while constraint (16) defines the binary variables.

4. Optimization algorithm

The proposed scheduling problem is *NP*-hard, (Costa, 2014) therefore a metaheuristic algorithm based on evolutionary principle and a random search has been proposed. Starting from a genetic algorithm properly adapted to a group scheduling problem, the proposed evolutionary algorithm (EA) aims to enhance the efficiency of the genetic procedure by embedding a random sampling algorithm.

4.1 Genetic algorithm

The genetic procedure starts with a set of feasible solutions (the *initial population*) and iteratively replaces the current population by a new population generation based on probabilistic criteria that preserves the most promising solutions. The population evolution proceeds by means of a *reproduction mechanism* called *crossover*, which selects one or more couples of chromosomes (the *parents*) and recombines them to generate the *children* (two new chromosomes). Children having higher fitness performance, the makespan, than parents replace them into the new population. Search towards unexplored areas of the solutions domain is assured by means of a mutation operator, which can randomly alter one or more chromosomes within a population. The algorithm proceeds by evolving the population through successive *generations,* until a given stopping criterion is reached. In order to apply the crossover and the mutation a proper *representation* of the solution domain (*chromosome encoding*) is considered. A numerical example describes the genetic operators. The problem consists of three groups, $G_1(J_1,J_{2,..}\ J_9)$, G_2 $(J_1,J_2,..J_6)$ and $G_3(J_1,J_{2...}\ J_5)$, worked on 5 machines. There are five workers. A feasible solution can be represented by the following chromosome [*C*1]:

$$C1= \begin{array}{|l} 2\ 3\ 6\ 1\ 5\ 4\ 9\ 7\ 8 \\ 1\ 4\ 5\ 6\ 3\ 2 \\ 2\ 3\ 4\ 1\ 5 \\ \hline 1\ 2\ 3 \\ \hline 3\ 1\ 5\ 2\ 4 \\ \hline \end{array} \qquad (17)$$

The rows of the chromosome are the sub-chromosomes. They represent from 1 to 3 the schedules of jobs within each group. Sub-chromosome *4* fixes the sequence of groups Ω=1-2 -3, while sub-chromosome corresponding to the sequence Θ=3−1−5−2−4 assigns worker 3 to machine1, worker 1 to machine 2 and

so on. If C2 is another individual of the population it is possible to apply the two methods of crossover operators denoted as *Position Based Crossover* (*PBC*) and *Two Point Crossover* (*TPC*), respectively.

$$C2= \begin{array}{|l|}\hline 4\ 2\ 1\ 3\ 7\ 5\ 8\ 6\ 9 \\ 2\ 4\ 6\ 5\ 3\ 1 \\ 4\ 2\ 5\ 1\ 3 \\ \hline 2\ 1\ 3 \\ \hline 2\ 1\ 4\ 3\ 5 \\ \hline \end{array} \tag{18}$$

PBC generates offspring by considering the relative order in which some alleles are positioned within the parents. Indeed, it works on a couple of sub-chromosomes G_{11} and G_{12} (the second index regards the chromosomes) as follows: 1) one or more alleles are randomly selected; 2) the alleles genetic information of *parent 1* (G_{11}) are reordered in the *offspring 1* (O_{11}) in the same order as they appear within the second *parent 2* (G_{12}); 3) remaining elements are positioned in the sequence by copying directly from the *parent 1* the unselected alleles. The same procedure is followed in the second parent, i.e. *parent 2*, to obtain offspring (O_{12}). The offspring substitute his parent if he improves the makespan. If only O_{12} improve the solution C2 and the selected alleles are those in positions {2}, {4}, {5} and {7}, the new solutions C2 is (19).

$$C2= \begin{array}{|l|}\hline 4\ 8\ 1\ 3\ 2\ 5\ 7\ 6\ 9 \\ 2\ 4\ 6\ 5\ 3\ 1 \\ 4\ 2\ 5\ 1\ 3 \\ \hline 2\ 1\ 3 \\ \hline 2\ 1\ 4\ 3\ 5 \\ \hline \end{array} \tag{19}$$

As far as is concerned with the *TPC* method, two positions are randomly selected and each sub-chromosome 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 sub-chromosome. Probability of selecting either *PBC* or *TPC* crossover is denoted as p_{cr}. If only O_{12} improve the solution C2 and the selected alleles are those in positions {3}, {7}, the new solutions C2 is Eq. (20),

$$C2= \begin{array}{|l|}\hline 4\ 2\ 3\ 1\ 5\ 7\ 8\ 6\ 9 \\ 2\ 4\ 6\ 5\ 3\ 1 \\ 4\ 2\ 5\ 1\ 3 \\ \hline 2\ 1\ 3 \\ \hline 2\ 1\ 4\ 3\ 5 \\ \hline \end{array} \tag{20}$$

The Mutation operator may be applied with probability p_m to a chromosome belonging to the current population; such a chromosome is randomly selected on the basis of its fitness and it is substituted by a new chromosome generated by means of the mutation operator. The criterion according to which mutation can modify one or more sub-chromosomes of a single solution consists of a "fair coin toss" 50% probability and two kind of operators (each one having the same chances to be selected) have been adopted in the present research: an *Allele Swapping Operator* (*ASO*), which performs an exchange on a random number of alleles; and a B*lock Swapping Operator* (*BSO*), which performs a block exchange. To avoid any loss of the current best sequence, the survival of the current fittest individual within the population is ensured by an elitist strategy. Probability of selecting either ASO or BSO mutation operator has been denoted as p_{cm}. A population diversity control technique has been embedded within the proposed optimization technique, in order to mutate those identical chromosomes exceeding a pre-selected value D_{MAX} in the current population. The exit criterion of the proposed evolutionary is based on the number

of iterations *IT* reached by the algorithm. Nevertheless, whenever a new best solution is found, the algorithm iteration counter is suddenly updated with the aim of allowing *ΔIT* further iterations.

4.2 Biased Random Sampling (BRS)

A Biased Random Sampling (BRS) search scheme (Baker and Trietsch, 2009) has been considered. Such procedure operates only on the N_{best} best individuals obtained after each generation. For each selected chromosome C_s (s = 1,2,…, N_{best}), a sample of N_{BRS} neighbour solutions is generated by modifying the sequence of groups and the sequences of the workers. For example, if there are five groups in the sequence, the probability that each group has to be extracted in the new sequence is:

$$p_{1,k} = \alpha^k \cdot \frac{1}{\sum_k \alpha^k} \qquad k = 1,2,...,5 \tag{21}$$

where $p_{1,k}$ is the probability to select the k-th group of the Ω as first element of the new sequence as reported in Table 1 and α is the biased parameter.

Table 1
Probability distribution

k	1	2	3	4	5
p_{1k}	0.297	0.238	0.19	0.152	0.122

Once the first group is chosen, the probability is recalculated according to the following distribution of probabilities:

$$p_{2,k} = \alpha^k \cdot \frac{1}{\sum_k \alpha^k} \qquad k = 1,2,...,5+1-2 \tag{22}$$

After a total of N_{BRS} solutions are originated from chromosome C_s, the best one is used for replacing C_s in the current population, whether it leads to a better makespan value. Such procedure is executed for all the N_{best} individuals originally selected. Then, the newly obtained population undergoes the next generation cycle.

4.3 Pseudocode of the proposed EA

The pseudocode of the proposed Evolutionary algorithm (EA) is reported:

Step 1: Initialization of parameters N_{pop}, p_{cr}, p_m, D_{max}, N_{best}, N_{BRS}, α;
Step 2: Generation of initial random population of chromosomes;
Step 3: Application of crossover operator, chosen between Position Based and Two Point Crossover, to a couple of chromosomes chosen through roulette-wheel selection;
Step 4: Generation of the new population after crossover operator through the insertion of the two best chromosomes individuated between parents and offspring: the two individuals with best values of fitness are introduced in the population;
Step 5: Evaluation of p_m. If it is verified go to Step 6 else go to Step 7;
Step 6: Application of mutation operator, chosen among two different operators: Allele Swapping and Block Swapping. The operator is applied randomly to a chromosome of the population;
Step 7: Population control: a mutation operator is applied on duplicates exceeding D_{max}.;
Step 8: Application of BRS procedure to the group substring of N_{best} best individuals of the population;
Step 9: Application of BRS procedure to the worker substring N_{best} best individuals of the population;
Step 10: Updating of the current population, then return to Step 3.

5. Experimental computations

In order to evaluate the efficiency of the evolutionary algorithm (EA) two procedures are considered. Firstly the evolutionary algorithm with workers assignment module is compared with the same algorithm with random assignment of the workers to the machines. This comparison permits to evaluate the effectiveness of the proposed model and the relevance of the best assignment of the workers on the value of objective function, the makespan. The second step evaluates the performance of the evolutionary algorithm in comparison with the best known algorithms applied to this scheduling problem or, if it is available, with the exact solution obtained solving the mathematical model.

The comparisons in the two procedures are developed on the basis of a proper benchmark of problems using the scheme provided by Salmasi et al. (2011). In reference to the parameters of the mathematical model described in the paragraph 3, the test cases are generated as shown in Tables 2. The processing times of each job on each machine has been randomly drawn from the range [1, 20], where symbol U[*a*, *b*] denotes a value extracted by a uniform distribution between *a* and *b*.

Table 2
Benchmark

Model parameters	Level 1		Level 2		Level 3	
Number of groups	U [1,5]		U [6,10]		U [11,16]	
Number of jobs in a group	U [2,4]		U [5,7]		U [8,10]	
Number of machines	2		3		6	
Set up time	M_1	U [1,50]	M_1	U [1,50]	M_1	U [17,67]
	M_2	U [17,67]	M_2	U [1,50]	M_2	U [1,50]
	M_1	U [1,50]	M_1	U [17,67]	M_1	U [45,95]
	M_2	U [17,67]	M_2	U [17,67]	M_2	U [17,67]
	M_3	U [45,95]	M_3	U [17,67]	M_3	U [1,50]
	M_1	U [1,50]	M_1	U [1,50]	M_1	U [300,350]
	M_2	U [17,67]	M_2	U [1,50]	M_2	U [170,220]
	M_3	U [45,95]	M_3	U [1,50]	M_3	U [92,142]
	M_4	U [92,142]	M_4	U [1,50]	M_4	U [45,95]
	M_5	U [170,220]	M_5	U [1,50]	M_5	U [17,67]
	M_6	U [300,350]	M_6	U [1,50]	M_6	U [1,50]

With regards to the workers characteristics, 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 (1999) model, respectively. The different skill levels, *SL*, of the heterogeneous workers are randomly chosen in the discrete interval [0.5 0.75 1 1.25 1.5]. Two distinct replicates have been randomly generated for each test case. Therefore, a total of 3 (levels of groups) × 3 (levels of machines) × 3 (levels of setup) × 2 (levels of learning) = 162 separate instances have been created.

5.1 Worker assignment

In this section, the EA with optimized workers assignment procedure and the EA with a random assignment of the workers are compared. The parameters of the EA algorithm are $N_{pop} = 70$, $p_{cr} = 0.9$, $p_m = 0.1$, $D_{max} = 2$, $N_{best} = 20$, $N_{BRS} = 4$, $\alpha = 0.8$. In table 3 the results are reported for the different levels of learning index for different number of machines, in total six classes of problems everyone constituted by 27 instances. The performance indicator used to compare the algorithms is the Relative Percentage Deviation (*RPD*), calculated as follows:

$$RPD = 100 \cdot \frac{ALG_{sol} - BEST_{sol}}{BEST_{sol}}, \tag{23}$$

where ALG_{sol} is the makespan solution provided by a given algorithm with reference to a certain instance and $BEST_{sol}$ is the lowest makespan value among EA with optimized assignment and EA with random assignment. It is possible that the algorithms obtain the same solution, if the algorithms do not reach the same value, the bold values indicates the best solution. In the bottom of each table are reported three performance indicators to evaluate the effectiveness of the algorithms:

- $RPD_{average}$ is the average value of all RPDs;
- $N_{minimum}$ denotes the number of times each optimization procedure reaches the best solution;
- N_{best} represents the number of times the algorithm is the best one among the two algorithms;

The EA with optimized assignment outperform the EA with random assignment in all classes of problems, in particular the efficiency of EA significantly increase with the number of machines. The RPD values of the algorithm EA with worker assignment reach a significant improvement, even up to 46% on the instances with 6 machines. Conversely, the EA with random assignment reach a very small improvement in all classes. Moreover, the results show how the learning effect influences significantly the performance of the EA with the optimized assignment and confirm the role of the heterogeneous workers and the importance of an intelligent assignment to the machines.

Table 3

Efficiency of workers assignment optimization module

	2 machines				3 machines			
	-0.152 learning effect		-0.323 learning effect		-0.152 learning effect		-0.323 learning effect	
	Optimized Assignment	Random Assignment	Optimized Assignment	Random Assignment	Optimized Assignment	Random Assignment	Optimized Assignment	Random Assignment
1	0.000	0.000	0.000	0.000	**0.000**	0.062	**0.000**	0.009
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	**0.000**	0.143	0.000	0.000	0.000	0.000	**0.000**	0.132
4	**0.000**	0.109	0.000	0.000	**0.000**	0.164	0.000	0.000
5	0.000	0.000	0.000	0.000	**0.000**	0.080	**0.000**	0.224
6	0.000	0.000	**0.000**	0.136	**0.000**	0.063	**0.000**	0.031
7	**0.000**	0.010	**0.000**	0.068	**0.000**	0.091	**0.000**	0.223
8	0.000	0.000	**0.000**	0.044	**0.000**	0.124	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
10	**0.000**	0.386	**0.000**	0.356	**0.000**	0.311	**0.000**	0.204
11	0.000	0.000	**0.000**	0.047	0.003	**0.000**	**0.000**	0.028
12	0.000	0.000	**0.000**	0.274	**0.000**	0.004	**0.000**	0.279
13	0.003	**0.000**	0.000	0.000	**0.000**	0.499	0.000	0.000
14	0.000	0.000	0.000	0.000	**0.000**	0.028	**0.000**	0.055
15	**0.000**	0.007	**0.000**	0.066	**0.000**	0.115	**0.000**	0.015
16	0.000	0.000	**0.000**	0.113	**0.000**	0.177	0.000	0.000
17	0.000	0.000	0.013	**0.000**	0.008	**0.000**	**0.000**	0.010
18	0.000	0.000	0.000	0.000	**0.000**	0.106	**0.000**	0.051
19	**0.000**	0.235	**0.000**	0.007	0.007	**0.000**	0.025	**0.000**
20	0.040	**0.000**	0.015	**0.000**	**0.000**	0.008	**0.000**	0.011
21	0.009	0.000	0.026	**0.000**	**0.000**	0.148	**0.000**	0.231
22	**0.000**	0.006	**0.000**	0.005	**0.000**	0.006	**0.000**	0.420
23	0.014	**0.000**	**0.000**	0.001	0.006	**0.000**	0.002	**0.000**
24	**0.000**	0.086	**0.000**	0.044	**0.000**	0.011	**0.000**	0.136
25	**0.000**	0.003	**0.000**	0.025	0.006	**0.000**	**0.000**	0.002
26	0.000	0.000	**0.000**	0.011	**0.000**	0.003	**0.000**	0.010
27	**0.000**	0.010	0.009	**0.000**	0.002	**0.000**	**0.000**	0.140
$RPD_{average}$	0.002	0.037	0.002	0.044	0.001	0.074	0.001	0.082
$N_{minimum}$	22	17	23	13	21	9	25	8
N_{best}	10	5	14	4	18	6	19	2

Table 3
Efficiency of workers assignment optimization module (Continued)

	6 machines			
	-0.152 learning effect		-0.323 learning effect	
	Optimized Assignment	Random Assignment	Optimized Assignment	Random Assignment
1	**0.000**	0.165	**0.000**	1.638
2	**0.000**	0.127	**0.000**	0.040
3	**0.000**	0.801	**0.000**	0.387
4	**0.000**	0.741	**0.000**	0.057
5	**0.000**	0.049	**0.000**	0.131
6	0.000	0.000	**0.000**	1.382
7	**0.000**	0.132	**0.000**	0.483
8	**0.000**	0.064	**0.000**	0.019
9	**0.000**	0.120	**0.000**	0.434
10	0.000	0.000	**0.000**	1.598
11	0.000	0.000	**0.000**	0.007
12	**0.000**	0.791	**0.000**	0.046
13	**0.000**	0.113	**0.000**	1.405
14	**0.000**	0.085	**0.000**	0.065
15	**0.000**	1.435	**0.000**	1.297
16	**0.000**	0.991	**0.000**	1.239
17	**0.000**	0.010	**0.000**	0.051
18	**0.000**	0.517	**0.000**	0.114
19	**0.000**	1.288	**0.000**	0.169
20	**0.000**	0.052	**0.000**	0.019
21	0.003	**0.000**	**0.000**	1.576
22	**0.000**	0.692	**0.000**	0.022
23	**0.000**	0.048	**0.000**	0.036
24	**0.000**	1.407	**0.000**	0.123
25	**0.000**	0.308	**0.000**	0.279
26	**0.000**	0.007	**0.000**	0.009
27	0.001	**0.000**	**0.000**	0.154
$RPD_{average}$	0.000	0.368	0.000	0.473
$N_{minimum}$	25	4	27	0
N_{best}	23	2	27	0

5.2 Comparison of algorithms

The instances of the benchmark have been solved by the mathematical model through MILP solver into ILOG CPLEX® commercial tool, by the EA algorithm and by another two algorithms proposed in literature. In particular they are the Hybrid Particle Swarm Optimization (hereinafter coded as PSO) devised by Salmasi et al. (2011) and the metaheuristic procedure hybridizing Genetic and Simulated Annealing algorithms (hereinafter coded as GSA) proposed by Naderi et al. (2012). The optimization procedures are executed on a 2GB 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. The performance indicator is the RPD but in this case, the best solution, $BEST_{sol,}$ is the lowest makespan value among those obtained or the exact solution obtained by the MILP solution. Firstly, the effectiveness of the three algorithms has been evaluated with respect to the exact solutions obtained resolving the mathematical model. In table 4 the number of optimal solutions are reported for every algorithm which is calculated as percentage of the available optimal solutions. The results, reported for the different levels of learning index and for different number of machines, show that EA algorithm outperforms in all classes the PSO and the GSA algorithms. In the case of two machines the EA algorithm achieves the exact solution in almost all the problems (97%).

Table 4
Exact solutions

learning index	machines	PSO	GSA	EA
-0.152	2	76%	72%	97%
	3	67%	67%	83%
	6	41%	48%	63%
-0.323	2	83%	77%	97%
	3	57%	53%	83%
	6	43%	39%	61%

The second comparison considers the minimum value makespan reached in every problem by the three algorithms. This value can coincide with the exact solution if it corresponds to the solution obtained solving the mathematical model. The numbers of time that every algorithm reaches the minimum value is reported in table 5. The results are showed in the same manner as in table 4, the percentage is calculated with respect to the overall instance, that is 27 for each class.

Table 5
Best solutions

learning index	machine	PSO	GSA	EA
-0,152	2	70%	39%	78%
	3	52%	41%	83%
	6	50%	35%	83%
-0,323	2	67%	48%	80%
	3	52%	31%	80%
	6	44%	30%	87%

The EA has the highest percentage of success among the three algorithm in all classes, around ~~the~~ 80% that is the double of GSA algorithm. Instead the difference with PSO is more significant in the instance with 2 and 3 machines and learning index of -0.323.

Table 6
RPD

learning index	machine	PSO	GSA	EA
-0,152	2	0,33%	2,31%	0,27%
	3	0,48%	2,13%	0,17%
	6	0,62%	1,52%	0,16%
-0,323	2	0,22%	1,97%	0,27%
	3	0,35%	2,03%	0,19%
	6	0,59%	1,31%	0,04%

Table 7
Anova analysis

Source	Sum of Squares	df	Mean Square	F Value	p-value Prob > F
Block	9,75E+04	1	9,75E+04		
Model	1,61E+08	97	1,66E+06	48,11	< 0.0001
A-M (number of machines)	3,68E+07	2	1,84E+07	531,52	< 0.0001
B-G (number of groups)	6,35E+07	2	3,18E+07	918,59	< 0.0001
C-J (number of jobs)	1,31E+07	2	6,54E+06	189,08	< 0.0001
D-S (setup times)	1,12E+07	2	5,61E+06	162,23	< 0.0001
E-a (learning index)	1,07E+06	1	1,07E+06	31,03	< 0.0001
AB	5,57E+06	4	1,39E+06	40,30	< 0.0001
AC	2,63E+05	4	6,58E+04	1,90	0,111
AD	1,65E+07	4	4,12E+06	119,20	< 0.0001
AE	4,02E+05	2	2,01E+05	5,81	0,0035
BC	4,07E+06	4	1,02E+06	29,45	< 0.0001
BD	1,95E+06	4	4,87E+05	14,09	< 0.0001
BE	3,98E+05	2	1,99E+05	5,76	0,0036
CD	7,03E+04	4	1,76E+04	0,51	0,7298
CE	1,67E+04	2	8,36E+03	0,24	0,7855
DE	3,36E+05	2	1,68E+05	4,85	0,0087
Residual	7,78E+06	225	3,46E+04		

The results of RPD value on Table 6 shows that EA show the effectiveness of both PSO and EA in solving the problem at hand, with a superiority of the proposed evolutionary algorithm, whereas GSA has poor performance. The analysis of EA algorithm is completed evaluating the effects of the model parameters (machines, group, jobs, setup and learning) on the makespan value. A proper Anova analysis has been performed through Design Expert® 7.0.0 version commercial tool to evaluate the effects of the first and the second order. Table 7 shows the results of the ANOVA and it is possible to notice that all first order effect of the parameters are significant. Only the second order effect of machines*group, machines*setup, group*jobs and group*setup is significant.

6. Conclusions

In this paper the scheduling of a group of jobs in a flow shop and the assignment to the machines of the workforce having different skills levels and learning ability is considered. 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 objective of the scheduling is the minimization of the completion time. A new mathematical model is considered to optimally solve the scheduling problem. Due to the large computational time required to cope with large-sized instances, an evolutionary algorithm (EA) is proposed. A comparison campaign based on a wide benchmarks arisen from literature involving two-, three- and six-machine problems has been fulfilled in order to test the performance of EA with respect to the two latest metaheuristic procedures presented by literature in the field of FSDGS scheduling problems. The obtained numerical results highlight the effectiveness of EA in approaching 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 are a critical resource or machines are affected by deterioration effects or breakdown.

References

Agustín-Blas, L. E., Salcedo-Sanz, S., Ortiz-García, E. G., Portilla-Figueras, A., Pérez-Bellido, Á. M., & Jiménez-Fernández, S. (2011). Team formation based on group technology: A hybrid grouping genetic algorithm approach. *Computers & Operations Research*, *38*(2), 484-495.

Allahverdi, A., Gupta, J. N., & Aldowaisan, T. (1999). A review of scheduling research involving setup considerations. *Omega*, *27*(2), 219-239.

Baker, K.R., & Trietsch, D. (2009). *Principles of Sequencing and Scheduling*. John Wiley & Sons: Hoboken, NJ, USA.

Benavides, A. J., Ritt, M., & Miralles, C. (2014). Flow shop scheduling with heterogeneous workers. *European Journal of Operational Research*, *237*(2), 713-720.

Biskup, D. (1999). Single-machine scheduling with learning considerations. *European Journal of Operational Research*, *115*(1), 173-178.

Biskup, D. (2008). A state-of-the-art review on scheduling with learning effects. *European Journal of Operational Research*, *188*(2), 315-329.

Celano, G., Costa, A., & Fichera, S. (2010). 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*, *46*(5-8), 695-705.

Cheng, T. E., Gupta, J. N., & Wang, G. (2000). A review of flowshop scheduling research with setup times. *Production and Operations Management*, *9*(3), 262-282.

Corominas, A., Olivella, J., & Pastor, R. (2010). A model for the assignment of a set of tasks when work performance depends on experience of all tasks involved. *International Journal of Production Economics*, *126*(2), 335-340.

Costa, A., Cappadonna, F. A., & Fichera, S. (2014). Joint optimization of a flow-shop group scheduling with sequence dependent set-up times and skilled workforce assignment. *International Journal of Production Research*, *52*(9), 2696-2728.

Dorn, C., Skopik, F., Schall, D., & Dustdar, S. (2011). Interaction mining and skill-dependent recommendations for multi-objective team composition. *Data & Knowledge Engineering*, *70*(10), 866-891.

Lodree, E. J., Geiger, C. D., & Jiang, X. (2009). Taxonomy for integrating scheduling theory and human factors: Review and research opportunities. *International Journal of Industrial Ergonomics*, *39*(1), 39-51.

Fitzpatrick, E. L., & Askin, R. G. (2005). Forming effective worker teams with multi-functional skill requirements. *Computers & Industrial Engineering*, *48*(3), 593-608.

Hajinejad, D., Salmasi, N., & Mokhtari, R. (2011). A fast hybrid particle swarm optimization algorithm for flow shop sequence dependent group scheduling problem. *Scientia Iranica*, *18*(3), 759-764.

Hendizadeh, S. H., Faramarzi, H., Mansouri, S. A., Gupta, J. N., & ElMekkawy, T. Y. (2008). Meta-heuristics for scheduling a flowline manufacturing cell with sequence dependent family setup times. *International Journal of Production Economics*, *111*(2), 593-605.

Hunter, J. E. (1986). Cognitive ability, cognitive aptitudes, job knowledge, and job performance. *Journal of vocational behavior*, *29*(3), 340-362.

Karthikeyan, S., Saravanan, M., & Rajkumar, M. (2016). Optimization of Worker Assignment in Dynamic Cellular Manufacturing System Using Genetic Algorithm. *Journal of Advanced Manufacturing Systems*, *15*(01), 35-42.

McDonald, T., Ellis, K. P., Van Aken, E. M., & Patrick Koelling, C. (2009). Development and application of a worker assignment model to evaluate a lean manufacturing cell. *International Journal of Production Research*, *47*(9), 2427-2447.

Moreira, M. C. O., Miralles, C., & Costa, A. M. (2015). Model and heuristics for the assembly line worker integration and balancing problem. *Computers & Operations Research*, *54*, 64-73.

Naderi, B., & Salmasi, N. (2012). Permutation flowshops in group scheduling with sequence-dependent setup times. *European Journal of Industrial Engineering*, *6*(2), 177-198.

Perron, L. (2010). Planning and scheduling teams of skilled workers. *Journal of Intelligent Manufacturing*, *21*(1), 155-164.

Pinedo, M. (2012). *Scheduling*. 4th ed., Springer-Verlag, New York.

Salmasi, N., & Logendran, R. (2008). A heuristic approach for multi-stage sequence-dependent group scheduling problems. *Journal of Industrial Engineering International*, *4*(7), 48-58.

Salmasi, N., Logendran, R., & Skandari, M. R. (2011). Makespan minimization of a flowshop sequence-dependent group scheduling problem. *The International Journal of Advanced Manufacturing Technology*, *56*(5-8), 699-710.

Savino, M. M., Brun, A., & Mazza, A. (2014). Dynamic workforce allocation in a constrained flow shop with multi-agent system. *Computers in Industry*, *65*(6), 967-975.

Valls, V., Pérez, Á., & Quintanilla, S. (2009). Skilled workforce scheduling in service centres. *European Journal of Operational Research*, *193*(3), 791-804.

Wang, J. B., & Wang, J. J. (2014). Flowshop scheduling with a general exponential learning effect. *Computers & Operations Research*, *43*, 292-308.

Wi, H., Oh, S., Mun, J., & Jung, M. (2009). A team formation model based on knowledge and collaboration. *Expert Systems with Applications*, *36*(5), 9121-9134.

Wright, T. P. (1936). Factors affecting the cost of airplanes. *Journal of the aeronautical sciences*, *3*(4), 122-128.

Zhu, X., & Wilhelm, W. E. (2006). Scheduling and lot sizing with sequence-dependent setup: A literature review. *IIE transactions*, *38*(11), 987-1007.