Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 17 (2014) 428 – 433

www.elsevier.com/locate/procedia

Variety Management in Manufacturing. Proceedings of the 47th CIRP Conference on Manufacturing Systems

Robust Metaheuristics for Scheduling Cellular Flowshop with Family Sequence-Dependent Setup Times

Al-mehdi Ibrahem[a*], Tarek Elmekkawy[b] and Qingjin Peng[c]

[a,c] *Department of Mechanical and Manufacturing Engineering, University of Manitoba, Winnipeg, R3T 2N2, Canada*
[b] *Department of Mechanical and Industrial Engineering, Qatar University, Doha, Qatar*

* Corresponding author. Tel.: +1-204-4747474; fax: +1-204-275-7507. *E-mail address:* umalmehd@cc.umanitoba.ca

Abstract

In manufacturing systems, minimization of the total flow time has a great impact on the production time, the productivity and the profitability of a firm. This paper considers a cellular flowshop scheduling problem with family sequence setup time to minimize the total flow time. Two metaheuristic algorithms based on Genetic algorithm (GA) and particle swarm optimization (PSO) are proposed to solve the proposed problem. As it is customarily accepted, the performance of the proposed algorithms is evaluated using Design of Experiments (DOE) to study the robustness of the proposed metaheuristics based on the Relative Percentage Deviation (RPD) from the lower bounds. The results of the DOE evaluation of the proposed algorithms show that PSO-based metaheuristic is better than GA for solving scheduling problems in cellular flow shop, which aims to minimize the total flow time.

Keywords: Cellular manufacturing flowshop; Sequencing-dependent setup; Total flow time; Genetic Algorithm; Particle Swarm Optimization

1. Introduction

The challenges faced by manufacturing companies have forced them to place a significant value on efficiency, timing, and cost to remain competitive [1]. Cellular manufacturing system (CMS) aims to optimize the efficiency, timing, and cost effectiveness. In flowshop manufacturing, cells usually consist of a number of machines that are dedicated to produce a specific group of part families that have similar production requirements. In a manufacturing cell, parts (jobs) having similar tooling or required setup on machines are assigned to be processed as one family. This is done, in order to improve the efficiency of the production cell. The required setup time on a machines used for switching between jobs that belong to the same part family is usually considered as a part of processing time of these jobs. This is because the setup required for switching the process between part families is quite significant. Further, the setup time depends on the family to be processed and the preceding part family. Such a problem is called cellular flowshop scheduling problem with sequence dependent setup times. Readers may refer to Allahverdi et al [2] for a comprehensive study of the scheduling problems with setup times. Broadly, efficient job scheduling is a crucial aspect of any manufacturing environment [3]. Scheduling a flowshop cellular manufacturing with family setup times has been the subject of several studies reported in the literature, which was initiated by Schaller et al [4]. They developed several heuristic algorithms with minimization of the makespan as the criteria. Further, they developed a lower bounding method to evaluate

doi:10.1016/j.procir.2014.01.072

the solution quality of the proposed algorithms. Franca et al [5] developed the Genetic Algorithm (GA) and Memetic algorithm (MA) with local search for the makespan minimization performance measure. The authors concluded that the solution quality of the MA outperformed the available algorithms. Hendizadeh et al [6] developed a meta-heuristic algorithm based on Tabu Search (TS) for the proposed problem to minimize the makespan. They concluded that the solution quality is the same as that proposed by Franca et al[5]. Lin et al [7] studied the same problem and proposed a simulated annealing based meta-heuristic algorithm to minimize the makespan as the criterion. Salmasi et al [8] investigated the aforementioned problem for total flow time minimization for the first time; they then developed two metaheuristics based on Tabu Search and Ant Colony Optimization algorithm to minimize the total flow time as the criterion for the first time. They claimed that the algorithm has a superior performance in comparison the TS algorithm. Their algorithm is considered as the best available metaheuristic. Moreover, tight lower bounds for makespan minimization and total flow time minimization are developed by Salmasi et al [8]and [9]. Later, Naderi, and Salmasi [10] developed two different mixed integer linear programming models. These models were effective in solving even medium-sized problems and providing the optimum solution in a reasonable time.

As it highlighted in the previous paragraphs, most of the research effort has focused on developing approximation algorithms to minimize the makespan. However, the total flow time minimization is an important measure that needs to be considered. Optimizing total flow time reflects a stable utilization of resources, reducing the work-in-process inventory, and minimizing the setup time costs. In addition, the research reported in the literature evaluates the quality of the solution based on the Percentage Deviation Error (PDE) from the lower bounds. In our view, according to El-Ghazali (2009) [11], the analysis of metaheuristics should be done in three steps: experimental design, measurement (e.g., quality of solutions, computational effort, and robustness), and reporting (e.g., box plots, interaction plots) [11]. In our research, two metaheuristics based on Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) are developed to minimize the total flow time in a cellular flowshop scheduling problem with family sequence setup time. Moreover, the performance of the proposed algorithms is evaluated to find the most robust algorithm by means of the Design of Experiments (DOE) approach. This paper aims to present the paradigm to identify the most robust metaheuristic in a cellular flowshop scheduling problem.

2. Problem description

The focus of this research is to schedule a flowshop cellular manufacturing cell with sequence dependent family setup times with the aim of minimizing the total flow time which is notated as: $Fm \mid fmls, Splk, prum \mid \sum C_j$. The objective function of the proposed problem is:

$$Minimize \sum_{j=1}^{N_0} C_{j,m} \tag{1}$$

To describe the problem, we first define the following notations:

F:	Number of the families,
N_0:	Number of jobs;
N_f:	Number of the jobs in each family, $f=\{1,2,.....,F\}$
M:	Number of machines;
m:	Index for machines, $i=\{1,2,....,m\}$
j, k:	Index for jobs, $j,\ k\ \epsilon \{1,2,....,N_0\}$
f,l :	Index for families, $f,l\ \epsilon \{1,2,....,F\}$
$p_{j,i}$:	Processing time for job j on machine i
$Se_{l,f,m}$:	Setup time of family f if it processed immediately after the family l on machine i

The major assumptions are invoked in this study are:

- The number of parts (jobs), their processing times, the number of part families, and their setup times are known in advance.
- The sequence of parts in a part family, and part families are the same on all machines (permutation scheduling).
- Once a part starts to be processed on a machine, the process cannot be interrupted before completion (pre-emption is disallowed).
- Each machine can handle only one part (job)at a time.
- The jobs belonging to each family should be processed without any preemption by other jobs of other families (group technology assumption).
- The ready time of all parts is zero, which means that all parts in all part families are available for processing at the start time.
- Setup time depends on both the part to be processed and its preceding part. There is minor setup among parts within a family whereas there are major setup times among the part families.

In general, a solution of the cellular flowshop scheduling problem is performed in two phases: sequencing of the part families and sequencing parts (jobs) within each family. In other words, the solution representation consists of ***F* + *1*** segments. While the first segment ***F*** represents the sequence of part families on each machine, the other segments correspond to the sequence of parts (jobs) within each part family[12]. For a feasible schedule, the sequence of the families is:

$$\pi = \{\pi_{[1]}, \pi_{[2]} \ldots \pi_{[f-1]}, \pi_{[f]}, \pi_{[f+1]}, \ldots.., \pi_{[F]}\}$$

Where $\pi_{[f]} =\{ \delta_{[f][1]}\ , \delta_{[f][2]}\ , \delta_{[f][j]}\ , \ldots.., \delta_{[f][Nf]} \}$ is the sequence of the jobs within the family. Two metaheuristics which are based on Genetic algorithm (GA) and particle swarm optimization (PSO) are developed to solve the proposed problem heuristically.

3. Genetic Algorithm

Genetic Algorithm (GA) has successfully been applied to the scheduling problems. It consists of making a population of solutions. These individuals are evolved by mutation and reproduction processes. The best fitted solutions of the population survive while the worse fitted will be replaced [5].

The basic steps of the proposed GA are as follows:

- **Step 1:** Initialize the population by generating initial solutions.
- **Step 2:** Assess each solution in the population based on the total flow time and assigning a fitness value.
- **Step 3:** Select individuals for recombination and go to step 4.
- **Step 4:** Recombine individuals generating new ones and enter step 5.
- **Step 5**: Mutate the new individuals and go to step 6.
- **Step 6**: If the stopping criterion is met, STOP, otherwise, replace old individuals with the new ones, and return to step 3.

3.1. Selection

The proposed algorithm starts with an initial solution that is generated randomly from the population (Pop. size). After that, the fitness function is estimated based on the objective function which is used to minimize the total completion time or total flow time (TFT). The fitness function is equal to the inverse of TFT. Hence each solution, namely solution π, the desirable values of the fitness function is the higher values, and the normal probability can be determined by the following formula [10]:

$$\frac{\frac{1}{TFT(\pi)}}{\sum_{X=1}^{PopSize}\frac{1}{TFT(X)}} \qquad (2)$$

Based on the fitness values, two solutions are randomly picked for the reproduction process using the crossover and mutation operators.

3.2. Crossover Operator

The crossover operator is used to find better solutions by recombining good genes from different parent chromosomes. The crossover operator combines the selected solutions to build two new ones. For instance, a random solution of the aforementioned problem can be represented in two segments: the sequence of the families, and the sequence of jobs in each family. In this paper, crossover has been performed similar to the crossover presented by Franca et al [5] and Hendizadeh et al [13]. A new offspring is constructed from two parents as follows: A segment of parent one is randomly selected and copied in the same position in the new offspring. The rest of the first part of the offspring is completed or filled from the second parent [13].

3.3. Mutation Operator

The mutation operator generates offspring from a single parent. First, two random positions in each part of a parent are chosen and then their positions are swapped. The offspring is compared with its corresponding parent. This different from the traditional GA, where either computation time or the number of generations is selected as termination criterion [14]. In this paper, intensive the parameters are tuned using Design of Experiments (DOE) to find the best combination of the algorithm parameters. We concluded that the population size is 30, the maximum CPU 60 sec, mutation rate is two, and crossover rate is 0.9.

4. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a novel iterative computational evolution model that was developed by Kennedy and Eberhart [15]. They simulated birds' swarm behaviour in PSO, and made every particle in the swarm move according to its experience and the best particle's experience to find a new better position. After the evolution, the best particle in the swarm was seen as the best solution for the input problem. The population of PSO is called swarm, and each individual or particle which is a potential solution is known with its current position and current velocity. The new position of each individual particle is obtained by assigning a new position as well as a new velocity to the particle. Each particle gains a different position, and the value of each position is evaluated based on the objective function value of the position. The main advantage of this approach is that every particle always remembers its best position in the experience. When a particle moves to another position, it must refer to its best experience and the best experience of all particles in the swarm. The best position of each particle that has been gained so far during the previous steps is named the best particle (p-best). The best position gained by all particle so far is named global best (g-best) position of the search. The new position as well as the new velocity of each particle is obtained based on the previous positions, the p-best, and the g-best. Considering an *n*-dimension search space; there are *S* particles (swarm size) cooperating to find the global optimum in the search space.

A warm of *S* particles, the i^{th} particle is associated with the position vector $x_i = (x_{i1}, x_{i2}, \ldots\ldots, x_{in})$, and the velocity $v_i = (v_{i1}, v_{i2}, \ldots\ldots, v_{in})$ the best solution is obtained by the i^{th} particle (p-best), and the best so-far solution is obtained by whole swarm (global best) are updated each iteration. Each particle uses its own search experience and the global experience by the swarm to update the velocity and files to a new position based on the following equations:

$$v_{ij}(t+1) = wv_{ij}(t) + C_1 r_1 \left(y_{ij}(t) - x_{ij}(t)\right) + C_2 r_2 \left(Y_j^*(t) - x_{ij}(t)\right) \qquad (3)$$

$$x_{ij}(t+1) = v_{ij}\left((t+1) + x_{ij}(t)\right) \qquad (4)$$

Where *w* is the inertia weight that controls the influence of the previous velocity of particles, C_1 , C_2 are called acceleration coefficient to weight the social influence. The parameters r_1 , and r_2 are uniformly distributed random variables in (0, 1), and. Particles fly in the search space based on Eqs. (3) and (4). Every particle always remembers its best position in the experience, where $i = \{1,2,\ldots,S\}$, and $j = \{1,2,\ldots,n\}$. When a particle moves to another position, new velocity is calculated according to the previous velocity and the distance of its position both p-best and g-best. However, the new velocity is limited to the range $[V_{\min}, V_{\max}]$ to control the extreme traveling of particles outside the search space. Particles gain their new positions according to the new velocity and the previous position (Eq.4). These continuous positions of the particles will be converted into discrete values using the Ranked Order Value (ROV). It converts the continuous position value of the particles to job and family sequences. For more details, the reader may refer to Liu et al [16].

The steps of the PSO algorithm are executed as follows:

```
Step1: Initialize parameters.
Step2: Iteration =0;
    2.1 Generate initial position, initial velocity: $\{X_i^0, i=1,2,\ldots,n\}$ and $\{V_i^0, i=1,2,\ldots,n\}$
    2.2 Apply the (ROV) to find the permutation;
    2.3 Evaluate each particle based on the objective function $\{f_i^0, i=1,2,\ldots,n\}$;
    2.4 For each particle in the swarm using the objective function $f_i^0$ for $i=1,2,...,n$ for each particle set:
        $P_i^0 = X_i^0, where P_i^0 = \left[ p_{i1}^0 = x_{i1}^0,\ p_{i2}^0 = x_{i2}^0,\ldots,p_{in}^0 = x_{in}^0 \right]$
        together with its best fitness value $f_i^{pb}$ for $i=1,2,...n$
    2.5 Find the best fitness value among the whole swarm such as $f_l = min\ \{f_i^0\}\ for\, i=1,2,\ldots,n$
    2.6 Set the global best to $G^0 = X_l^0$ such that $G^0 = \left[ g_1 = x_{l,1},\ g_2 = x_{l,2},\ldots,g_n = x_{l,n} \right]$ with its fitness value $f^{gb} = f_l$
Step3: Repeat until Iteration =Maximum Iteration
Step 4: Update velocity
    $v_{ij}(t+1) = wv_{ij}(t) + C_1 r_1 \left( y_{ij}(t) - x_{ij}(t) \right) + C_2 r_2 \left( Y_j^*(t) - x_{ij}(t) \right)$
Step 5: Update position $x_{ij}(t+1) = v_{ij}\left((t+1) + x_{ij}(t)\right)$
Step6: Apply the (ROV) to find the permutation
Step7: Update personal best
Step8: Update global best to $G^t = X_l^t$
Step9: If the stopping criterion is satisfied, STOP; otherwise, return to step 3.
```

The algorithm starts with generating the initial velocity and position according to the following equations:

$$X_{ki} = X_{\min} + rand\left(X_{\max} - X_{\min}\right) \qquad (5)$$

$$V_{kj} = V_{\min} + rand\left(V_{\max} - V_{\min}\right) \qquad (6)$$

Where X_{min} and $X_{\max}$ represent the minimum and maximum position values. V_{min} And $V_{\max}$ represent the minimum and maximum velocities. Also *Rand* is a random variable between (0, 1). These values are presented in Table 1.

Table 1: The Min and Max. values of the PSO parameters.

Parameter	Max. Values	Min. Value
W	*2*	*0.4*
C₁	*2*	*0.4*
C₂	*2*	*0.4*
Position value	*0*	*4*
Velocity	*-4*	*4*

To evaluate the performance of the proposed algorithms, test problems developed by Salmasi [9] have been used in this research. . Three different parameters define a problem structure. These include: number of machines, number of families, and maximum job in family. The test problems are classified into small, medium, and large size problems with the number of families is a random integer from DU [1, 5], DU [6, 10], and DU [11, 16] respectively. The number of jobs in a family is also a random integer from DU [2, 4], DU [5, 7], and DU [8, 10] for small, medium, and large problems respectively. Also, the test problems are classified into two and six machine problems.

5. Robustness

A schedule is considered robust if its quality is only slightly sensitive to data uncertainties and to unexpected events [17]. Furthermore, robustness means the evaluation process that is able to determine the best or the most efficient algorithm that gives a high quality solution. Samarghandi et al [18] reported that the most important feature of Design of Experiments (DOE) is its ability to study the interaction effects between the considered factors.. The levels corresponding to each parameter are presented in Table 2.

Table 2 Factor levels.

Factor	Level1	Level2	Level 3
Max. Job. in Family	*2-4*	*5-7*	*8-10*
N. Family	*2-5*	*6-10*	*11-16*
Machine	*2*	*-*	*6*

6. Results and discussion

In this section, we will compare the performance of two metaheuristics based on GA and PSO. The comparison is conducted to answer the following questions: 1) which algorithm is more robust? 2) What are the factors affecting the response? To answer the questions, a statistical analysis based on (DOE) is conducted. The response is based on the Relative Percentage Deviation (RPD) that measures the deviation of the total flow time from the corresponding lower bound developed by Salmasi [8]. The RPD is calculated using the following formula:

$$RPD = 100 \times \frac{TFT - LB}{LB} \qquad (7)$$

This study showed that the number of families has a more significant impact than the other factors. Nevertheless, maximum number of jobs in a family significantly affected the RPD values when the numbers of families is large as shown in Figures 1, and 2.

Table 3: ANOVA results

Source	Degrees of freedom	Seq. Sums of Squares	Adj. Sum of Squares	Adj. Mean Square	F-value	P-value
Max. Job. in Family	*2*	*0.04149*	*0.04149*	*0.02075*	*0.64*	*0.535*
N. Family	*2*	*0.22447*	*0.25389*	*0.12695*	*3.92*	*0.033*
Machine	*1*	*0.00598*	*0.00598*	*0.00598*	*0.18*	*0.671*
Algorithm	*1*	*0.00684*	*0.00684*	*0.00684*	*0.21*	*0.650*
Error	*25*	*0.80866*	*0.80866*	*0.03235*		
Total	*31*	1.08744				

Furthermore, based on the P-values, analysis of variance (ANOVA) showed that the most significant factor is the number of families since its P-value is close to zero. Yet, as shown in Table 3, most of other factors are insignificant since their values are greater than 0.005.

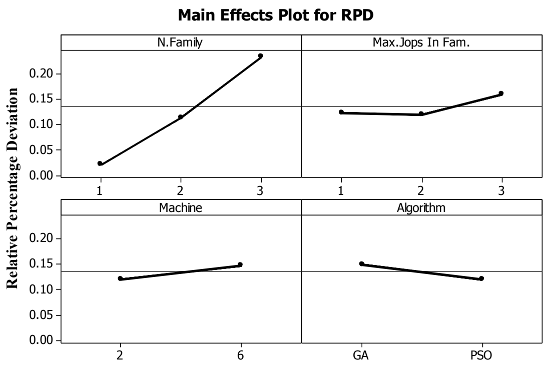

Figure 1: The main effect plot

Moreover, studying the interaction among the factors gives a better understanding about the effect of several simultaneous factors. As shown in Figure 3, GA is better than PSO if the number of families is small and medium. Yet, large families are more practical in manufacturing. Even though the maximum jobs in families have greater effect on the RPD in both metaheuristics as shown in Figure 4, PSO shows better performance than GA.

It should also be noted that metaheuristic algorithm based on PSO is more efficient than that based on GA. For instant, for the two-machine problems, the variation in RPD for GA is more than for PSO as shown in Figure 5. Yet, GA gives some solutions that are better than the ones obtained by PSO. However, in terms of the variation, PSO shows less variation than GA.

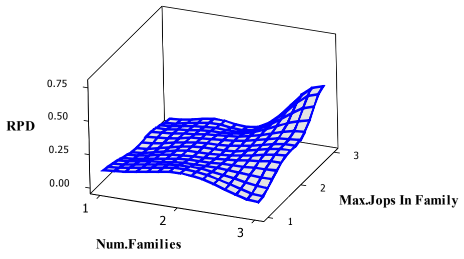

Figure 2: Surface Plot of RPD vs Max.Jops In Family, and No. of Families

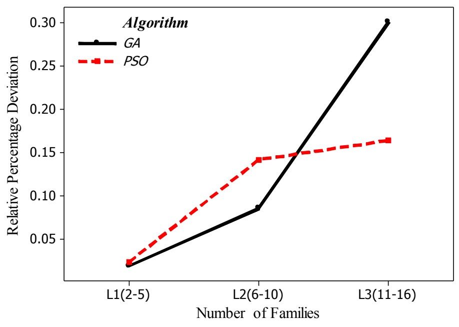

Figure 3: Interaction plots of maximum jobs in family vs. the proposed algorithms

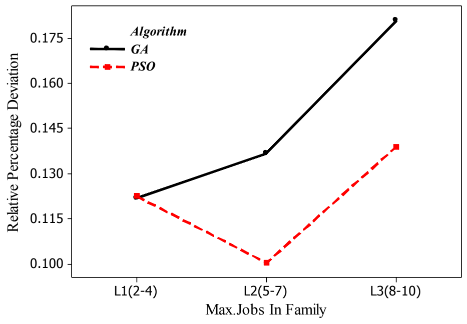

Figure 4: Interaction plots of maximum jobs in family vs. the proposed algorithms.

In addition, there is a considerable difference in the variation in RPD between the two metaheuristics for the six-machine problems as shown in Figure 6, indicating that PSO is performed better than GA. Therefore, PSO is more efficient than GA in solving the cellular flowshop to minimize the total flow time.

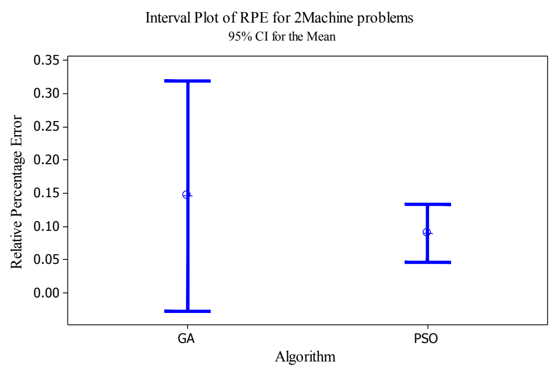

Figure 5:Interval plot for two-machine problems.

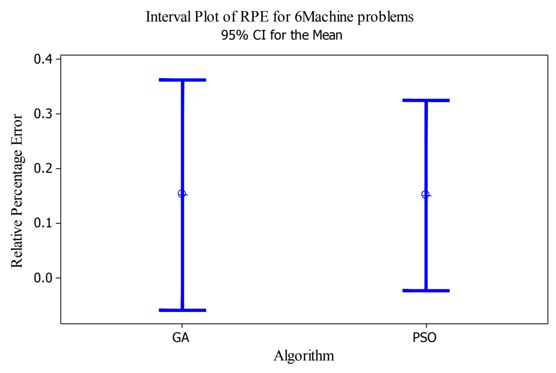

Figure 6: Interval plot for six-machine problems

7. Conclusion

In this paper, we examined two metaheuristics to find robust schedules that minimizing total flow time in cellular flowshop scheduling problems with sequence dependent setup times. Metaheuristic algorithms based on PSO and GA is selected due to their successful implementation in scheduling. The Robustness of these metaheuristics are compared Design of Experiments (DOE) technique based on the Relative Percentage Deviation from the corresponding lower bounds proposed by Salmasi et al [8] . It was found the proposed PSO algorithm is more robust than GA even though high quality solutions can be obtained by GA based on the problems examined in this study. Future research on this study is to study the robustness of any other metaheuristic algorithms or other performance measures like makespan. Additionally, the proposed approach can be implemented in multi-objective optimization problems.

References

[1] Dennie JS. Efficient job scheduling for a cellular manufacturing environment. Rochester Institute of Technology, 2006.
[2] Allahverdi A, Ng C, Cheng T, Kovalyov M. A survey of scheduling problems with setup times or costs. Eur J Oper Res 2008;187:985–1032.
[3] Shiyas CR, Madhusudanan Pillai V. Cellular manufacturing system design using grouping efficacy-based genetic algorithm. Int J Prod Res 2014:1–14.
[4] Schaller J, Gupta JND, Vakharia AJ. Scheduling a flowline manufacturing cell with sequence dependent family setup times. Eur J Oper Res 2000;125:324–39.
[5] Franca P, Gupta J, Mendes A, Moscato P, Veltink K. Evolutionary algorithms for scheduling a flowshop manufacturing cell with sequence dependent family setups. Comput Ind Eng 2005;48:491–506.
[6] Hamed Hendizadeh S, Faramarzi H, Mansouri SA, Gupta JND and, Elmakkawy T. Meta-heuristics for scheduling a flowline manufacturing cell with sequence dependent family setup times. Int J Prod Econ 2008;111:593–605.
[7] Lin S-W, Gupta JND, Ying K-C, Lee Z-J. Using simulated annealing to schedule a flowshop manufacturing cell with sequence-dependent family setup times. Int J Prod Res 2009;47:3205–17.
[8] Salmasi N, Logendran R, Skandari MR. Total flow time minimization in a flowshop sequence-dependent group scheduling problem. Comput Oper Res 2010;37:199–212.
[9] Salmasi N, Logendran R, Skandari MR. Makespan minimization of a flowshop sequence-dependent group scheduling problem. Int J Adv Manuf Technol 2011.
[10] Naderi B, Salmasi N. Permutation flowshops in group scheduling with sequence- dependent setup times. Eur J Ind Eng 2012;6:177–98.
[11] El-Ghazali T. Metaheuristics: from design to implementation. Jonh Wiley Sons Inc, Chichester 2009.
[12] Bouabda R, Jarboui B, Rebai A. A nested iterated local search algorithm for scheduling a flowline manufacturing cell with sequence dependent family setup times. (LOGISTIQUA), 2011 4th 2011:526–31.
[13] Hendizadeh H, Elmekkawy T, Wang G. Bi-criteria scheduling of a flowshop manufacturing cell with sequence dependent setup times. Eur J Ind Eng 2007;1:1751–5254.
[14] Zhang Y, Li X, Wang Q. Hybrid genetic algorithm for permutation flowshop scheduling problems with total flowtime minimization. Eur J Oper Res 2009;196:869–76.
[15] Kennedy J, Eberhart R. Particle swarm optimization. Proc ICNN'95 - Int Conf Neural Networks 1995;4:1942–8.
[16] Liu B, Wang L, Jin Y-H. An effective hybrid PSO-based algorithm for flowshopscheduling with limited buffers. Comput Oper Res 2008;35:2791–806.
[17] Wang L, Ng AHC, Deb K, editors. Multi-objective Evolutionary Optimisation for Product Design and Manufacturing. London: Springer London; 2011.
[18] Samarghandi H, Elmekkawy TY, Ibrahem AM. Studying the effect of different combinations of timetabling with sequencing algorithms to solve the no-wait job shop scheduling problem. Int J Prod Res 2013;51-16:4942–65.