Simultaneous Considering of Machine Availability Constraint, Sequence Dependent Setup Time and Ready Time in a No-wait Hybrid Flow Shop Scheduling Problem to Minimize Mean Tardiness

Rahmanidoust Mohammad
Glorious Sun School of Business and Management
DongHua University
Shanghai, China

Zheng Jianguo
Glorious Sun School of Business and Management
DongHua University
Shanghai, China

Rabiee Meysam
Department of Industrial Engineering
Bu-Ali Sina University
Hamedan, Iran

ABSTRACT
This paper deal with the problem of no-wait hybrid flow shop which sequence dependent setup times, ready time and machine availability constraint. Minimizing the mean tardiness is considered as the objective to develop the optimal scheduling algorithm. To do so, an efficient hybrid imperialist competitive algorithm (HICA) is proposed to tackle this problem. Our proposed algorithm is compared to other solution approaches reported on the literature. For this purpose, we draw an analogy between the results obtained from algorithms applied to some random case study. To achieve reliable results, Taguchi approach is used to define robust parameters' values for our proposed algorithm. The superiority and effectiveness of our proposed algorithm is inferred from all the results obtained in various situations.

General Terms
Scheduling, Taguchi, Metaheuristic algorithms.

Keywords
No-wait; Scheduling; Hybrid flow shop; Meta-heuristic algorithms; Imperialist competitive algorithm.

1. INTRODUCTION
Due to importance of scheduling in manufacture environment and the fact that scheduling act as an important process for making decision, there is a need to be extensive researches on the scheduling problems. Recently, solving some new problems in scheduling area have been attracted by many researchers [1-18] The problem of how to sequence jobs in a no-wait hybrid flow shop has received vast attention in the academic literature, but the majority of literature focuses on how to minimize maximum completion time (i.e. makespan).

A no-wait hybrid flow shop scheduling problem can be widely considered in various applications. For example, in the plastic products industry, it is required for a series of processes to be performed, one immediately after another, in order to prevent degradation. In another situation, in the chemical industry; having a waiting time between each subsequent stage may lead to changes in the material properties (e.g. degrading the polymer) [2]. Similar situations also arise in the pharmaceutical industries [19].

In a no-wait flow shop, the jobs are processed from one machine to the next one without waiting time [20]. It is assumed that there are n jobs each one consists of m operations owning a predetermined processing order through machines. Each job is to be processed without preemption and interruption on or between m machines. That is, once a job is started on the first machine, it has to be continuously processed through subsequent machines without interruption. In addition, each machine can handle no more than one job at a time and each job has to visit each machine exactly once. Therefore, when needed, the start of a job on the first machine must be delayed in order to meet the no-wait requirement [21].

Review of the researches reveal that majority of the papers on no wait hybrid flow shop scheduling problem have concentrated on makespan (i.e. maximum completion time). For example, Liu et al. [22] presented a heuristic algorithm named Least Deviation (LD) algorithm for two-stage no-wait hybrid flow shop scheduling with a single machine in either stage. The performance measure used in this study is makespan. The results showed that LD algorithm outperforms the others in most practical cases. In addition, the proposed algorithms showed low computational complexity and easy to implement, thus it is favorable application value.

Xie et al. [23] proposed a new heuristic algorithm known as Minimum Deviation Algorithm (MDA) to minimize makespan in a two stage flexible flow shop with no waiting time. Experimental results of the study showed that MDA outperforms partition method, partition method with LPT, Johnson's and modified Johnson's algorithms. Huang et al. [20] considered a no-wait two stage flexible flow shop with setup times and with minimum total completion time performance measure. They proposed an integer programming model and Ant Colony Optimization heuristic approach. The results revealed that the efficiency of the proposed algorithm is superior to those solved by integer programming while having satisfactory solutions.

To reduce the makespan in a two stage flexible flow shop with no waiting time, Xie et al. [23], proposed a sequential algorithm called Minimum Deviation Algorithm (MAD). The experiences show that MDA performs drastically better than partition method, partition method with LPT, Johnson's and modified Johnson's algorithms. Huang et al. [20] considered a no-wait two stage flexible flow shop with setup times and with minimum total completion time performance measure. They suggest an integer programming model and Ant Colony Optimization heuristic method. This algorithm, as discovered by results, is more efficient than those solved by integer programing with acceptable solutions.

Jolai et al. [24] introduced no-wait flexible flow line scheduling problem with time windows and job rejection to

maximizing profit. This is an extension of production and delivery scheduling problem with time windows. They also presented a mixed integer-linear programming model and genetic algorithm procedures to solve their model efficiently. Comparison of the results obtained by GA with LINGO solutions and Tabu search showed that the proposed GA obtains better solutions in a very low computational time in comparison with the solutions obtained from LINGO optimization software. Jolai et al. [25] proposed a novel hybrid meta-heuristic algorithm for a no-wait flexible flow shop scheduling problem with sequence dependent setup times. The computational evaluations of their research manifestly supported the high performance of their proposed novel hybrid algorithm.

In order to increase the profit, Jolai et al. [24] presented no-wait flexible flow line scheduling problem with time windows and job rejection. This is the same production and delivery preparation problem with time windows which is rather extended. To solve the model professionally They also offered a mixed integer-linear programming model and genetic algorithm processes. Comparing the two mentioned solutions and their gained outcomes, GA with LINGO solutions and Tabu search, released that if computational time is considered, the suggested GA gains smoothly better solutions than those of LINGO optimization software. Jolai et al. [25] proposed a novel hybrid meta-heuristic algorithm for a no-wait flexible flow shop scheduling problem with sequence dependent setup times. The assessment of their study clearly suggests the outperformance of their algorithm if computation is considered.

Ramezani et al. [26] dealt with a no-wait scheduling problem considering anticipatory sequence-dependent setup times on the flexible flow shop environment with uniform parallel machines to minimize maximum completion time of jobs. Since this problem was known to be NP-hard, they introduced a novel approach to tackle the problem. They proposed a hybrid meta-heuristic which involved invasive weed optimization, variable neighborhood search and simulated annealing to tackle of the problem. The experimental results of their research revealed the superiority of the performance of the hybrid meta-heuristic in comparison with original ones singularly. Rabiee et al. [8] addressed the problem of no-wait two stage flexible flow shop scheduling problem considering unrelated parallel machines, sequence dependent setup times, probable reworks and different ready times to minimize the maximum completion time.

One of the most common assumptions in the scheduling literature is that the machines or processors are always available for their use, when actually they may be stopped by several reasons, like failures and maintenances [27]. The other characteristic of this research is related to setup time. Machine setup time is an important factor for production scheduling in all flow-type manufacturing environments. Majority of scheduling researches on flow shops consider setup times to be negligible or as a part of the processing times [28]. But, we consider a sequence dependent setup time approach for our mentioned problem.

The remainder of this paper is organized as follows. In Section 2, a problem description is given in details. In Section 3, the frameworks of the proposed algorithm is explained. Numerical experiments developed to solve the problems are explained in section 4. This is followed by presenting the simulation results. Finally Section 5 presents the summary of the research with concluding remarks and recommendation for further researches.

2. PROBLEM DEFINITION

The no-wait hybrid flow shop scheduling problem (NWHFSSP) can be described as follows: given the processing time P (k, j) of Stage k on Job j and setup times S(i,j,k,l) between Job i and j at l th machines of kth stage. Each of n jobs will be sequentially processed in all stage respectively. At each stage there are m_k machines. The ready times of all jobs are different. It means the jobs cannot be processed at each time. Machines aren't always available. In other words, preventive maintenance is considered. Each of n jobs has unique due date to satisfy customer requirements. In addition, at any event of time, each machine can process at most one job. Similarly, each operation of a job can only be processed on one machine. Once the sequence of the jobs at the first stage is defined, the same sequence is applied for the second stage. To satisfy the no-wait restrictions, the completion time of a job on a given machine must be equal to the start time of the job on the next machine. In other words, there must be no waiting time between the processing of any consecutive operations of each job. The problem is to find a sequence that the total completion time is minimized.

3. PROPOSED ALGORITHM

3.1 Hybrid Imperialist Competitive Algorithm

Atashpaz-Gargari and Lucas [29] illustrated the imperialist competitive algorithm (ICA). ICA has been widely applied for many non-polynomial hard optimisation problems, such as flow shop and job shop scheduling [30].

ICA is originated Similar to other evolutionary algorithms using an initial population and any individual of the population is named a country. Countries are divided in two groups: imperialists and colonies. Some of the best countries (countries with the least cost) are chosen to be the imperialist countries and the rest are divided among the mentioned imperialists based on their power. The power of each country is calculated based on the objective function. A set of one imperialist and their colonies forms one empire. The total power of an empire is equal to the power of the imperialist country plus a percentage of mean power of its colonies. After forming initial empires, the competition starts, the colonies in each of empires start moving toward their imperialist country, and the imperialists attempt to achieve more colonies. Hence during the competition, the weak imperialist will be collapsed. At the end just one imperialist will remain. The framework of the proposed hybrid imperialist competitive algorithm (HICA) is described as follow:

Alike every evolutionary algorithm, ICA is invented by means of a primarily population and its people which is named a country. Imperialists and colonies are two major groups of a country. The imperialists are selected to be the heads of some colonies which are less economic than them. The distribution is based on the imperialists' power. The country authority is computed based on the goal and its function. An empire consists of one imperialist and the subordinate colonies. The power of an empire is equivalent the power of its imperialist and the average of its colonies' power. The rivalry begins as soon as the first empire generates. Every single colony keep moving toward its imperialist country and on the other hand, the imperialist itself tries to reach to more colonies. This is apparent that the weaker the colony, sooner to disappear. Finally, only the strongest imperialist remains. The outline of the suggested hybrid imperialist competitive algorithm (HICA) is described as follow:

3.1.1 Generating initial empires

Each solution in HICA is in a form of an array. Each array consists of variable values to be optimized. In GA terminology, this array is called "chromosome," but here, we use the term "country". In an N dimensional optimization problem, a country is a 1×N array. This array is defined by:

$$country = [p_1, p_2, p_3, ..., p_N] \quad \textbf{(1)}$$

Where Pi is the variable to be optimized. Each variable in a country denotes a socio-political characteristic of a country. From this point of view, all the algorithm does is to search for the best country that is the country with the best combination of socio-political characteristics such as culture, language, and economical policy [28].

In the HICA each solution (country) is a 1×N array of integer variables that N represents the number of jobs. The array of the country represents a sequence of jobs to be assigned to earliest available machines in both stages. The structure of one solution for a seven-job problem is shown in Figure 1:

Job 1	Job 2	Job 3	Job 4	Job 5	Job 6	Job 7
6	5	7	2	1	3	4

Fig 1: The structure of one solution for a seven-job problem in HICA

Figure II show that job5 is in the first sequence followed by jobs 4-6-7-2-1-3 in the other subsequent sequences respectively.

The cost of a country is calculated using a cost function f at the variables $(P_1, P_2, ..., P_N)$ as follow:

$$c_i = f(country_i) = f(p_{i1}, p_{i2}, ..., p_{iN}) \quad \textbf{(2)}$$

The algorithm starts with initial countries that are generated randomly by a number of population size (PopSize) and the most powerful countries (countries with minimum cost) are selected as the imperialists by a number of N-imp. The remaining countries are colonies each of which belongs to an empire. The colonies are distributed among imperialists based on imperialist's power. For calculating the imperialists power, the normalized cost of an imperialist are applied based on follow definition:

$$C_n = \max_i c_i - c_n \quad \textbf{(3)}$$

Where, c_n is the cost of nth imperialist and C_n is its normalized cost which is equal to the deviation of the maximum total completion time from the nth imperialist cost. The power of each imperialist is calculated according to Equation 4.

$$p_n = \left| \frac{C_n}{\sum_{i=1}^{N_{imp}} C_i} \right| \quad \textbf{(4)}$$

Having obtained the imperialist power, the colonies are distributed among the imperialist accordingly. In addition, the initial number of colonies of an imperialist is calculated as follow:

$$NC_n = round\{P_n . N_{col}\} \quad \textbf{(5)}$$

Where, NC_n is the initial number of colonies of nth imperialist and N_{col} is the number of all colonies. We randomly select NC_n of colonies and designate them for each imperialist. Imperialist with the bigger power has a greater number of colonies while imperialist with weaker power has less.

3.1.2 Moving the colonies of an empire toward the imperialist (assimilating)

Colonies start improving their power by capturing more Imperialist countries. Some part of a colony's structure will be similar to the empire's structure as it is created from the nature of this movement. Figure 2 illustrates the imperialist's and colony's arrays. In the HICA, the percent of job numbers from colony's array are chosen to be same the imperialist's array. For this purpose, a new array with the cells value equal to one and zero is randomly generated (Figure 3). Noted this, the number of the ones are equal to percent of jobs that have the positions equal to that of the imperialist array and named as Prct-Assimilate. Then the subsequent job positions are determined base on the orders defined in the colonies (i.e. 4th sequence for job number 4, 3th sequence for job number 5 and so forth). The resulting job sequence is shown in Figure 4.

	Job 1	Job 2	Job 3	Job 4	Job 5	Job 6	Job 7
Imperialist	6	5	7	2	1	3	4
Colony	4	3	1	6	7	5	2

Fig 2: Imperialist's and colony's arrays

Job 1	Job 2	Job 3	Job 4	Job 5	Job 6	Job 7
0	1	1	0	0	0	0

Fig 3: New array

Job 1	Job 2	Job 3	Job 4	Job 5	Job 6	Job 7
2	5	7	4	3	1	6

Fig 4: Assimilated colony

3.1.3 Exchanging positions of the imperialist and a colony

A colony might reach to a position with lower cost than the imperialist when colony moved toward the imperialist. In these situations, the position of the imperialist and the colony is swapped as showed in Figure 5 a. Afterward the algorithm will continue and the colonies will be moved toward imperialist in its new position. The resulting empire, after swapping the position of the imperialist and the colony, is depicted in Figure 5 b.

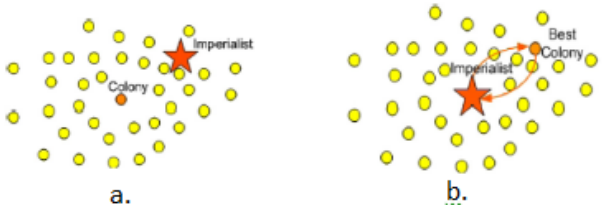

Fig 5: Exchanging positions of the imperialist and a colony

3.1.4 Total power of an empire

Total power of an empire is mainly affected by the power of the imperialist country, but the power of the colonies of an

empire has an indigent effect on the total power of that empire. Therefore, the equation of the total cost is defined as follow:

$$TC_n = \cos t(imperialist_n) + \xi mean\{\cos t(colonies\ of\ empire_n)\} \quad \textbf{(6)}$$

Where TC_n is the total cost of the nth empire and zeta (ξ) is a positive number which is considered to be less than 1. The total power of the empire to be determined by just the imperialist when the value of ξ is small. The role of the colonies, which determines the total power of an empire, becomes more important as the value of ξ increases.

3.1.5 Imperialistic competition

All empires attempt to take the possess and control of colonies of other empires. In the imperialistic competition the power of weaker emprises will gradually reduce and the power of more powerful ones will rise. In other words, picking some (usually one) of the weakest colonies of the weakest empire and making a competition among all empires to possess these colonies are the imperialistic competition. In this competition, the most powerful empires will not definitely possess these colonies, but these empires will be more likely to possess them. This competition is modelled by picking one of the weakest colonies from the weakest empire. Then in order to calculate the possession probability of each empire, first the normalized total cost is calculated as follows:

Every single empire is in a competition with others to authorize, own and govern colonies in other empires. As the competition goes forward, the strength of the weakers decrease incrementally. On the other hand, that of the strongers increases steadily. This is the nature of imperialistic competition among all empires that the authority of the weakest colony in the weakest empire is taken by other stronger empire. Although the most strong empire is not the only option to get the authority but this is the most likely to happen. When the weakest colony from the weakest empire is picked up, the competition begin to demonstrate. Here, the normalized total cost is calculated as follows to find out the authority probability of every single empire:

$$NTC_n = TC_n - \max\{TC_i\} \quad \textbf{(7)}$$

Where, NTC_n is the normalized total cost of nth empire and TC_n is the total cost of nth empire. Having normalized the total cost, the possession probability of each empire is calculated as below:

$$p_{Pn} = \left| \frac{NTC_n}{\sum_{i=1}^{N_{imp}} NTC_i} \right| \quad \textbf{(8)}$$

We use Roulette wheel method for assigning the mentioned colony to the empires.

3.1.6 Population based simulated annealing as a local search

Numbers of PBSA initial solutions are equal to the number of imperialists and the number of outputs is the same as well. Note that solution representation in this algorithm is same as HICA and PBSA and also machine assignments are performed based on first available machine at each stage.

3.1.7 Revolution

In each iteration, for every imperialist two positions of imperialist's array are chosen and these positions are exchanged together and new imperialist is replaced with the weakest imperialist colony's. These processes are repeated by a percentage of jobs for each imperialist named as Prct- Imp - R. Furthermore, some of the colonies are selected and then two positions of the colony's array are chosen and these positions are exchanged. These processes are repeated for a percentage of jobs for each colony named as Prct- Col -R. The replacement ratio is identified as the revolution rate and named as P-R.

3.1.8 Eliminating the powerless empires

Powerless empires will collapse and their colonies will be distributed among other empires in the imperialistic competition. In this paper, when an empire loses all of its colonies, we consider it as a collapsed empire.

3.1.9 Global war

After preceding a number of iterations, a Global War occurs, named as I-Global-War. Then new countries equal to the number of population are produced. This is followed by merging new countries with the old population and sorting the countries based on their accenting cost functions. Finally from the sorted countries, a number of countries equal to the old population is selected. This process is repeated a number of times known as N-Global-War.

Global War happens as some repetitions go forward that is named I-Global-War. As the competition continues, new countries and population are shaped. Countries merge and population mix together. These countries sort according to the relevant economic function. Then some countries are chosen which is equal to the previous population from the sorted countries. Iteration of this process is known as N-Global-War.

3.1.10 Stopping criteria

In this paper stopping criteria or end of imperialistic competition is considered when there is only one empire for all of the countries. A novel algorithm which is hybridization of imperialist competitive algorithm and population based simulated annealing (AICA+PBSA) is described in this section, it has similar base to AICA though applies a local search (PBSA) to improve imperialists in addition. In this algorithm, the numbers of PBSA initial solutions are equal to number of imperialists and number of outputs is same as well. Noted that, machine assignments are performed based on first available machine at each stage. Pseudo code for this proposed algorithm is depicted in Figure 6.

3.1.11 HICA notations

$MaxDc$: Maximum Decades

PopSize: Number of initial countries

P_i : A socio-political specific of a country

c_n : The cost function for nth country

N_{imp} : Number of imperialists

K: Boltzman constant

C_n : Normalized cost of nth country

P_n : Power of nth imperialist

NC_n : Difference in fitness function between new and previous solution

P_{as} : Percentage of assimilation

ξ : A positive constant for consideration of average power of colonies in each empire

TC_n : The total cost of the nth empire

NTC_n : The normalized total cost of nth empire

PP_n : Possession probability of each empire

P_{ir} : Percentage of imperialists revolution

P_{cr} : Percentage of colonies revolution

P_r : Probability of revolution

I_{gw} : Number of iterations for occurrence of global war

N_{gw} : Number of global war

```
Begin
Initialisation %
        – Set Parameters ((MaxDc, PopSize), N_imp, P_as, I_gw, N_gw, P_ir, P_cr, P_r, n_Pop, T_0, T_f, a, Max_ipt)
        – Generating initial Countries (Randomly)
        – Evaluate fitness of each country
Main loop of algorithm %
 For it = 1 : MaxDc
 Formation of empires %
        – Choice most powerful countries as the imperialists
        – Use PBSA as a local search for reach to better imperialists
        – Assign other countries to imperialists based on imperialist power
 Assimilation %
       – Move the colonies of an empire toward the imperialist
      – Revolution among colonies and imperialist
       If cost of colony < own imperialist
        exchanging positions of the imperialist and related colony
       end
 Imperialistic competition %
       – Calculate Total power of the empires
      – Select the weakest colony of the weakest empire and assign this to
        one of the strange empires based on their powers
      – Eliminate the powerless empires (the imperialist with no colony)
 Global war %
   if global war condition is met
        – Form new countries as the same number as PopSize
        – Evaluate fitness of each country
        – Merge new and old countries
     end
 Selection for next iteration %
        – Sort merged countries and choice the first countries with the size of the
         PopSize as new population
 end
end
```

Fig 6: The pseudo code of HICA1

4. COMPUTATIONAL EXPERIMENTS

4.1 Problem design

In this study we examined the effectiveness of the proposed approaches for a 15 test problems. The problem data can be characterized by three factors in terms of the number of jobs, number of machines, processing time, sequence dependent setup times, Ready time and machine availability time. Table 1 shows the random generated problems in detail.

Table 1. Problem parameters and their levels

Factors	Levels
Number of job	8,16,20,24,30
Number of stages	2,3,4
Machine distribution function	U(1,4)
Processing times	U(1,100)
Sequence dependent setup times	U(5,20)
Ready time	U(1,100)
Machine availability time	U(500,1000)

Calculate mean processing time of each job on all s stages.

$$p_j = round(\sum_{i=1}^{s} \frac{\sum_{u=1}^{no.eligible\,machine} p_{i,u}^{j}}{no.eligible\,machine_i}) \;, \forall i \in N \qquad (9)$$

Calculate average setup times for all possible subsequent jobs and sum it for all s stages.

$$s_j = round(\sum_{i=1}^{s} \frac{\sum_{u=1}^{no.eligible\,machine} s_{k,j,u}^{i}}{(n-1) \times no.eligible\,machine_i}) \;, \forall i \in N \qquad (10)$$

Determine a due date for each job with following formula.

$$d_j = p_j + s_j + round(\alpha \times U[0, \frac{\sum_{j=1}^{n}(p_j + s_j)}{\sum_{i=1}^{s} m^i}]) \qquad (11)$$

4.2 Parameter tuning

There are two different strategies for parameter tuning: Offline parameter initialization (or meta optimization), and online parameter tuning strategy. In the off-line parameter initialization the values of different parameters are fixed before the execution of the metaheuristic, whereas in the online approach the parameters are controlled and updated dynamically or adaptively during the execution of the metaheuristic [31]. In this research we used a try and error method to tune the parameters.

4.3 Experimental results

In this section the results of tested experiments for all algorithms are presented and the performance of the proposed algorithms is compared to each other in terms of the performance metrics. All algorithms were coded using MATLAB 2011a and run on personal computer with a 2.66 GHz CPU and 4 GB main memory.

Regarding the performance measures, Relative Percentage Deviation (RPD) over the best solutions is used. It is calculated as follows:

This section includes the outputs of the tested inspections for all algorithms. The performances of suggested ones are compared in terms of metrics as well. These algorithms were encrypted using MATLAB 2011a and run on personal computer with a 2.66 GHz CPU and 4 GB main memory.

Concerning the performance measures, Relative Percentage Deviation (RPD) over the best solutions is used. It is calculated as follows:

$$RPD = \frac{|Method_{sol} - Best_{sol}|}{Best_{sol}} \times 100 \qquad (12)$$

Where $Method_{sol}$ is value of method and $Best_{sol}$ is the best value between the algorithms. ARPD is the average of RPDs.

The effectiveness of the algorithms was testified by solving 15 different problems. Table 2 show the comparative results of the three algorithms with respect to RPD performance measures. As you can see in Table 2, HICA outperforms the other algorithms in terms of ARPD. In order to comparison of algorithms we provided an interval plot with 95% confidence interval. Figure 7 represents that HICA outperforms both of algorithms and also there isn't any meaning statistical difference between ICA and SA.

Table 2. Average relative percentage deviation (ARPD) for proposed algorithms

No. jobs	No. Stages	ARPD		
		SA	ICA	HICA
8	2	1/76	2/36	0/38
	3	0/85	1/12	0/58
	4	0/55	0/97	0/39
16	2	5/98	5/65	0/80
	3	5/03	5/75	1/22
	4	6/34	8/56	0/43
20	2	4/34	9/47	1/95
	3	3/83	5/42	0/86
	4	7/87	6/10	1/26
24	2	7/38	9/17	1/85
	3	5/55	11/62	1/20
	4	9/01	12/02	1/72
30	2	9/17	14/88	1/55
	3	12/87	16/29	2/48
	4	8/25	11/08	2/67
average		**5/92**	**8/03**	**1/29**

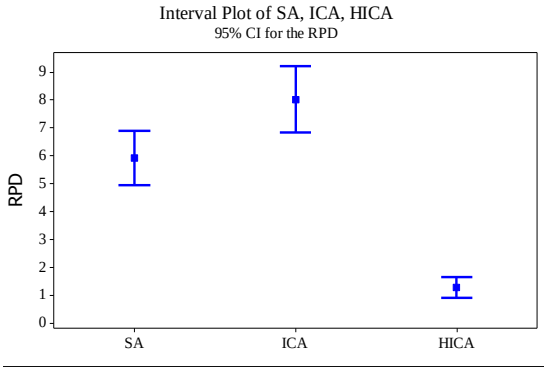

Fig 7. Intervals of algorithms in terms of RPD

5. CONCLUSION

In this paper, a no-wait hybrid flow shop scheduling problem is considered. The objectives of minimizing makespan is taken into account. The aim is finding the best approximate of job sequences for each problem. Three meta-heuristic algorithms, called SA, ICA and HICA, were proposed. In order to evaluate the performance of these algorithms, 15 problems in different sizes were solved. The performance of the proposed algorithms were studied in terms of RPD. The results of the numerical experiments revealed that HICA outperforms the other algorithms. As a direction for further research in this area, it is recommended to apply other efficient meta-heuristic algorithms to the mentioned problem.

6. ACKNOWLEDGMENTS

This research was supported by the National Natural Science Foundation of China (no. 70971020)

7. REFERENCES

[1] Ruiz, R. and Vazquez-Rodriguez, J. A., 2010. The hybrid flow shop scheduling problem. European Journal of Operational Research, 205, 1–18.

[2] Shafaei, R., Rabiee, M. and Mirzaeyan, M. An adaptive neuro fuzzy inference system for makespan estimation in multiprocessor no-wait two stage flow shop, Int. J. Comput. Integr. Manuf., 24(10), pp. 888?99 (2011).

[3] Shafaei, R., Moradinasab, N., & Rabiee, M. (2011b). Efficient meta heuristic algorithms to minimize mean flow time in no-wait two stage flow shops with parallel and identical machines. International Journal of Management Science and Engineering Management, 6(6), 421-430.

[4] Shafaei, R., Rabiee, M., & Mazinani, M. (2012). Minimization of maximum tardiness in a no-wait two stage flexible flow shop. International Journal of Artificial Intelligence? 8(S12), 166-181.

[5] Rabiee, M., Zandieh, M., & Jafarian, A. (2012a). Scheduling of a no-wait two-machine flow shop with sequence-dependent setup times and probable rework using robust meta-heuristics. International Journal of Production Research, 50(24), 7428-7446.

[6] Rabiee, M., Zandieh, M., & Ramezani, P. (2012b). Bi-objective partial flexible job shop scheduling problem: NSGA-II, NRGA, MOGA and PAES approaches. International Journal of Production Research, 50(24), 7327-7342.

[7] Moradinasab, N., Shafaei, R., Rabiee, M., & Ramezani, P. (2013). No-wait two stage hybrid flow shop scheduling with genetic and adaptive imperialist competitive algorithms. Journal of Experimental & Theoretical Artificial Intelligence, 25(2), 207-225.

[8] Rabiee, M., Rad, R. S., Mazinani, M., & Shafaei, R. (2014). An intelligent hybrid meta-heuristic for solving a case of no-wait two-stage flexible flow shop scheduling problem with unrelated parallel machines. The International Journal of Advanced Manufacturing Technology, 71(5-8), 1229-1245.

[9] Asefi, H., Jolai, F., Rabiee, M., & Araghi, M. T. (2014). A hybrid NSGA-II and VNS for solving a bi-objective no-wait flexible flowshop scheduling problem. The International Journal of Advanced Manufacturing Technology, 75(5-8), 1017-1033.

[10] Jolai, F., Tavakkoli-Moghaddam, R., Rabiee, M., & Gheisariha, E. (2014). An enhanced invasive weed optimization for makespan minimization in a flexible flowshop scheduling problem. Scientia Iranica. Transaction E, Industrial Engineering, 21(3), 1007.

[11] Samarghandi, H., & ElMekkawy, T. Y. (2014). Solving the no-wait flow-shop problem with sequence-dependent set-up times. International Journal of Computer Integrated Manufacturing, 27(3), 213-228.

[12] Fattahi, P., Azizi, V., & Jabbari, M. (2015). Lot Streaming in No-wait Multi Product Flowshop Considering Sequence Dependent Setup Times and Position Based Learning Factors. International Journal of Engineering TRANSACTIONS A: Basics, 28, 1031-1039.

[13] Azizi, V., Jabbari, M., & Kheirkhah, A. (2016). M-machine, no-wait flowshop scheduling with sequence dependent setup times and truncated learning function to minimize the makespan. International Journal of Industrial Engineering Computations, 7(2), 309-322.

[14] Joo, C. M., & Kim, B. S. (2015). Hybrid genetic algorithms with dispatching rules for unrelated parallel machine scheduling with setup time and production availability. Computers & Industrial Engineering, 85, 102-109.

[15] Rabiee, M., Jolai, F., Asefi, H., Fattahi, P., & Lim, S. (2016). A biogeography-based optimisation algorithm for a realistic no-wait hybrid flow shop with unrelated parallel machines to minimise mean tardiness. International Journal of Computer Integrated Manufacturing, 29(9), 1007-1024.

[16] Marichelvam, M. K., Tosun, ?, & Geetha, M. (2017). Hybrid monkey search algorithm for flow shop scheduling problem under makespan and total flow time. Applied Soft Computing, 55, 82-92.

[17] Guo, Z., Shi, L., Chen, L., & Liang, Y. (2017). A harmony search-based memetic optimization model for integrated production and transportation scheduling in MTO manufacturing. Omega, 66, 327-343.

[18] Fanjul-Peyro, L., Perea, F., & Ruiz, R. (2017). MIP models and mataheuristics for the unrelated parallel machine scheduling problem with additional resources. European Journal of Operational Research.

[19] Ruiz, R. and Allahverdi, A. ‘‘No-wait flowshop with separate setup times to minimize maximum lateness’’, Int. J. Adv. Manuf. Technol., 35(5), pp. 551–565 (2007).

[20] Huang R. H., Yang C. L. and Huang Y. C., 2009. No-wait two-stage multiprocessor flow shop scheduling with unit setup. International Journal of Advanced Manufacturing Technology, 44:921–927.

[21] Tasgetiren M. F., Pan Q. K., Suganthan P.N. and Liang Y. C., 2007. A Discrete Differential Evolution Algorithm for the No-Wait Flow shop Scheduling Problem with Total Flow time Criterion. Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling.

[22] Liu Z., Xie J., Li J. and Dong J., 2003. A heuristic for two-stage no-wait hybrid flowshop scheduling with a single machine in either stage, Tsinghua Science and Technology, 8, 43-48.

[23] Xie J., Xing W., Liu Z. and Dong J., 2004. Minimum Deviation Algorithm for Two-Stage No-Wait Flowshops with Parallel Machines. Computer and Mathematics with Application, 47, 1857-1863.

[24] Jolai F., Sheikh S., Rabbani M. and Karimi B., 2009. A genetic algorithm for solving no-wait flexible flow lines with due window and job rejection. International Journal of Advanced Manufacturing Technology, 42:523–532.

[25] Jolai, F., Rabiee, M., & Asefi, H. (2012). A novel hybrid meta-heuristic algorithm for a no-wait flexible flow shop scheduling problem with sequence dependent setup times. International Journal of Production Research, 50(24), 7447-7466.

[26] Ramezani P, Rabiee M, Jolai F (2013) No-wait flexible flowshop with uniform parallel machines and sequence-dependent setup time: a hybrid meta-heuristic approach. J Int Manuf 1–14.

[27] Ángel-Bello, F., Álvarez, A., Pacheco, J., & Martínez, I. (2011). A single machine scheduling problem with availability constraints and sequence-dependent setup costs. Applied Mathematical Modelling, 35(4), 2041-2050.

[28] Aldowaisan T (2001) A new heuristic and dominance relations for no-wait flowshops with setups. Comput Oper Res 28(6):563–584.

[29] Atashpaz-Gargari and Lucas, E.C., 2007. Imperialist Competitive Algorithm: An algorithm for optimization inspired by imperialist competitive. IEEE Congress on Evolutionary computation, Singapore.

[30] Shokrollahpour, E.; Zandieh, M.; Dorri, B. A novel imperialist competitive algorithm for bi-criteria scheduling of the assembly flowshop problem. Int. J. Prod. Res. 2011, 49, 3087–3103, doi:10.1080/00207540903536155.

[31] Talbi, E.G., Metaheuristics: from Design to Implementation, Wiley, 2009.