Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. VI (2011), No. 2 (June), pp. 286-296

A Hybrid Artificial Bee Colony Algorithm for Flexible Job Shop Scheduling Problems

J. Li, Q. Pan, S. Xie, S. Wang

Jun-qing Li, Sheng-xian Xie
School of Computer, Liaocheng University, Liaocheng
Liaocheng, 252059, PR China
E-mail: lijunqing.cn@gmail.com, xsx@lcu.edu.cn

Quan-ke Pan
1. School of Computer, Liaocheng University, Liaocheng
Liaocheng, 252059, PR China
2. State Key Lab. of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology
Wuhan, 430074, PR China
E-mail: qkpan@gmail.com

Song Wang
Department of Economic and Management
Shandong University of Science and Technology
Huangdao, 266510, PR China

Abstract: In this paper, we propose a hybrid Pareto-based artificial bee colony (HABC) algorithm for solving the multi-objective flexible job shop scheduling problem. In the hybrid algorithm, each food sources is represented by two vectors, i.e., the machine assignment vector and the operation scheduling vector. The artificial bee is divided into three groups, namely, employed bees, onlookers, and scouts bees. Furthermore, an external Pareto archive set is introduced to record non-dominated solutions found so far. To balance the exploration and exploitation capability of the algorithm, the scout bees in the hybrid algorithm are divided into two parts. The scout bees in one part perform randomly search in the predefined region while each scout bee in another part randomly select one non-dominated solution from the Pareto archive set. Experimental results on the well-known benchmark instances and comparisons with other recently published algorithms show the efficiency and effectiveness of the proposed algorithm.

Keywords: Flexbile job shop scheduling problem, artificial bee colony, multi-objective optimization, hybrid algorithm

1 Introduction

The flexible job shop scheduling problem (FJSP), as a branch of the classical job shop scheduling problem (JSP), has been studied in very recent years. Brandimarte (1993) [1] is among the first author to solve the FJSP instances with tabu search (TS) algorithm. In very recent years, some meta-heuristic algorithms, such as TS algorithm [2] [3], particle swarm optimization (PSO) [4] [5], ant colony optimization (ACO) [6], and genetic algorithm (GA) [7] [8], have been used in solving the single-objective FJSPs. Although the single-objective FJSP has been widely investigated, the research on the multi-objective FJSP is still considered relative limited. Kacem et al. (2002a, 2002b) [9] [10] proposed an effective evolutionary algorithm. Xia and Wu

(2005) [11] studied the problem with the hybrid algorithm of the PSO and the simulated annealing (SA). Zhang et al. (2009) [12] introduced a hybrid algorithm combining PSO algorithm with TS algorithm. Ho et al. (2008) [13]studied a hybrid evolution algorithm combined with a guided local search and an external Pareto archive set.

In this paper, we propose a hybrid algorithm combining an external Pareto archive set and the artificial bee colony (ABC) optimizer to solve the multi-objective FJSP. The rest of this paper is organized as follows: In Section 2, we briefly describe the problem formulation. Then, the artificial bee colony (ABC) algorithm is introduced in Section 3. The elements and framework of the hybrid algorithm are presented in Section 4 while Section 5 shows the experimental results and comparisons with other algorithms in the literature to demonstrate the superiority of the HABC performance. Finally, the last section presents conclusion of our work.

2 Problem Formulation

The FJSP considers n jobs to be processed on m machines. There are some assumptions and constrains in the FJSP considered in this study as follows: 1) each job has predefined number of operations and a known determined sequence among these operations; 2) each machine and each operation is ready at zero time; 3) each machine can only process one operation at a time, and each job must be processed on one machine at a given time; 4) each machine can process a new operation only after completing the predecessor operation; 5) each operation can be operated on a given candidate machine set instead of only one machine like in JSP; 6) given an operation O_{ij} and the selected machine M_k, the processing time p_{ijk} is also fixed.

Let C_i be the completion date of job J_i. W_k is the workload of machine M_k, which is the total processing time of operations that are operated on machine M_k. p_{ijk} be the processing time of $O_{i,j}$ on machine M_k. Three objectives are considered in this study, namely [13]:

1) minimization of maximum completion time (makespan):

$$F_1 = max\{C_i \mid i = 1, \ldots, n\} \tag{1}$$

2) minimization of total workload

$$F_2 = \sum p_{i,j,k} \tag{2}$$

3) minimization of critical machine workload:

$$F_3 = max\{W_k \mid k = 1, \ldots, m\} \tag{3}$$

3 Artificial Bee Colony Algorithm

Very recently, by simulating the behavior of honey bee swarm intelligence, an efficient bee colony (ABC) algorithm is proposed by Karaboga ([14] - [17]). Due to its simplicity and ease of implementation, the ABC algorithm has gained more and more attention and has been used to solve many practical engineering problems. In the basic ABC algorithm ([14] - [17]), there are two components: the foraging artificial bees and the food source. The position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality or fitness of the associated solution. The artificial bee is divided into three groups, namely, employed bees, onlookers, and scouts bees. The employed bee is the one who is currently performing exploitation on a food source. A bee that is waiting in the hive for making decision to choose a food source is called an onlooker. The scout bee is the one who perform exploration procedure and random exploitation search to find a new food source. The main steps of the algorithm are given as follows ([14] - [18]).

Step1. Produce initial population;
Step2. While stop criteria is not satisfied, perform steps 3 to steps 6.
Step3. Send the employed bees onto their food sources.
Step4. Send the onlooker bees onto the food sources depending on their nectar amounts.
Step5. Send the scout bees to search possible new food sources.
Step6. Memorize the best food source found so far.

4 The hybrid algorithm HABC

The basic ABC algorithm was originally designed for continuous function optimization. In order to make it applicable for solving the problem considered, a novel hybrid version of the ABC algorithm, named HABC, is proposed in this section.

4.1 Solution representation

The solution of the problem is represented with two vectors [19]: the machine assignment vector and the operation scheduling vector. The first part places the assigned machine number for each operation at the corresponding position, while the second part puts the same number symbol for each operation of a job and interpret them according to the occurrence in the operation scheduling vector.

4.2 Employed bee phase

The employed bee is to perform the local search around a given food source. Therefore, the employed bee takes the exploitation search of the algorithm. In order to generate good quality and diversity neighboring solutions, two types of local search operators are applied for the employed bees in this study, which are shown as follows.

(1)Local search operator in machine assignment component

The local search operator in machine assignment component is very simple and easy to be implemented. The perturbation is obtained by following steps.

Step1. Select a position in the machine assignment component, randomly or using some priority rules.

Step2. Assign a suitable machine different with the old one for the operation in the corresponding position.

Step3. Replace the machine number in the selected position and produce the new machine assignment component for the solution.

(2)Local search operator in operation scheduling component

The local search in the operation scheduling component is just like the perturbation in solving the JSP, where insert and swap operations are commonly used in the literature [20-22]. The insert operator is to remove a number symbol for an operation in the permutation π from its original position j and insert it into another position k such that $(k \neq j)$. The swap operator is to interchange two job symbols of π in the different positions.

After performing the above two local search approaches, the employed bee obtains a new neighboring food source around the old one. Then the new food source will be evaluated and compared with the old one. The better food source will be kept in the population as in the basic ABC algorithm which performs a greedy selection procedure.

4.3 Onlooker bee phase

In the classical ABC algorithm, each onlooker bee selects a food source based on the percent of the nectar amount of each food source among the total nectar amounts. However, the above approach consumes large computational time to compute the nectar amount of each food source. For this reason, we propose a tournament selection with the size of 3 in the HABC algorithm. In the tournament selection, three food sources are picked randomly from the population, and then the food source with highest nectar amount will be selected by the onlooker bee. After selecting the food source, each onlooker bee performs local search for the selected food source and produce a new neighboring food source. The better food source between the old one and the new neighboring one will be memorized in the population.

4.4 Scout bee phase

A scout bee performs randomly search in the basic ABC algorithm. This will increase the population diversity and avoid local minima, whereas this will also decrease the search efficacy. Since the food sources memorized in the Pareto archive set often carry better information than others, and the search space around these non-dominated solutions could be the most promising region. Therefore, in the HABC algorithm, the scout bees are first divided into two parts. One half of the scout bees randomly select a solution from the external Pareto archive set and perform several insert and swap operators to the selected solution, while the other half scout bees perform randomly search in the predefined search scope. In the hybrid algorithm, at least $5\% - 10\%$ of the population is scout bees.

4.5 Multi-objective optimizer

The Pareto archive set AS

To provide a set of solutions with good diversity, a Pareto archive set (AS) was introduced in this study, which is used to maintain a limited number of non-dominated solutions found so far. During the optimization process, the archive set is iteratively updated with adding some non-dominated solutions and removing some dominated solutions to get closer to the Pareto-optimal front. Once a new non-dominated solution is found, it will be added to AS and any solution which is dominated by the added one will be removed from AS. In case AS becomes overfull, its member which is in the crowded domain is eliminated to maintain the diversity of the Pareto archive set.

The storage structure of AS

To reduce the computational time complexity consumed on the update process of the archive set, the members of the AS firstly sequence in an ascending order according to their first objective function value (Pan, 2009) [21].

Non-dominated sorting algorithm

For the population, we should sequence each solution according to a certain criteria. For multi-objective optimization problems, we can not use one objective function value to determine the solution quality. In this study, a non-dominated sort algorithm (Deb et al., 2002) [23]was introduced to divide the population solutions into several levels according to their dominated solutions number.

4.6 The framework of HABC

The details steps of the proposed HABC algorithm are as follows:

Step1 Initialization phase;

- **Step 1.1** Set the system parameters;
- **Step 1.2** Produce the initial population.

Step2 Apply the Pareto non-dominated sorting function on the population, and then update the external Pareto archive set by using the solutions in the first Pareto level front.

Step3 If the stopping criterion is satisfied, output the non-dominates solutions in the external Pareto archive set; otherwise, perform steps 4-7.

Step4 Employed bee phase.

- **Step 4.1** Put each employed bee on each solution in the population.
- **Step 4.2** For each employed bee, perform local search on the assigned solution and generate a new neighboring solution.
- **Step 4.3** Evaluate the new neighboring solution and record the better solution among the new solution and the old one as the current solution and put it into the population. If the two solutions are non-dominated with each other, randomly select one as the current solution.
- **Step 4.4** If a solution has not been improved through limit cycles, then the corresponding employed bee becomes a scout bee and perform step 6.
- **Step 4.5** Evaluate each solution corresponding to each employed bee, apply the Pareto non-dominated sorting algorithm on the new population and update the external Pareto archive set using the solutions in the first Pareto level.

Step5 Onlooker bee phase.

- **Step 5.1** For each onlooker bee, randomly selects three solutions from the population and selects the best one as the food source. If the three solutions cannot dominate each other, then randomly select a non-dominate solution.
- **Step 5.2** For each onlooker bee, performs local search for the selected food source and carries out greedy selection procedure to record the better solution in the population.
- **Step 5.3** Evaluate each solution corresponding to each onlooker bee, apply the Pareto non-dominated sorting algorithm on the new population and update the external Pareto archive set using the solutions in the first Pareto level.

Step6 Scout bee phase.

- **Step 6.1** Divide the scout bees into two parts with the same number of bees.
- **Step 6.2** The scout bees in the first part randomly select a food source and perform local search operator in the predefined region. After generating a new solution, performs greedy selection procedure.
- **Step 6.3** Each scout bee in the second part randomly select a non-dominate solution in the external Pareto archive set and perform several local search for the selected solution.. After generating a new solution, performs greedy selection procedure.

Step 6.4 Evaluate each solution corresponding to each scout bee, apply the Pareto non-dominated sorting algorithm on the new population and update the external Pareto archive set using the solutions in the first Pareto level.

Step7 go to step 3.

5 Experiment Results

This section describes the computational experiments to evaluate the performance of the proposed algorithm. The test samples come from Kacem instances set [9]. The current instantiation was implemented in C++ on a Pentium IV 1.8GHz with 512M memory.

5.1 Setting parameters

Each instance can be characterized by the following parameters: number of jobs (n), number of machines (m), and the number of operations (op_num). Followings are the detail parameters value:

The size of the population is equal to the number of employed bee and the number of onlooker, which is set to $5n$; the maximum cycle of the algorithm is set to $10 \times n \times m$; the limit number of cycles through which no improvement occurs on the food source, then the employed bee becomes a scout bee; the limit number is set to $n \times \frac{m}{2}$; the percent of scout bee is set to a random number between 0.05 and 0.1.

5.2 Results comparisons

The five test instances come from Kacem [9] [10], which range from 4 jobs×5 machines to 15 jobs ×10 machines. Two tests are performed for comparison, i.e. the instances with single objective and the problems with three objectives. Several recently published algorithms are compared with the proposed HABC algorithm, such as the AL+CGA proposed by Kacem et al. (2002b) [10] ,the GENACE approach employed by Ho (2004) [24], the PSO+SA developed by Xia and Wu (2005) [11], the ant systems & local search optimization method (hereafter called ACO+LS) presented by Liouane et al. (2006) [6], and the PSO+TS introduced by Zhang (2009) [12].

The signle objective five Kacem instances

For solving the five instances with single objective to minimize the makespan criterion, the experimental results and comparisons are given in Table 1. It can bee seen from Table 1 that the HABC algorithm can obtain the best results for all the Kacem instances. The proposed algorithm outperforms the AL+CGA in 4 out of 5 instances, while outperforms the GENACE method in 2 out of 4 cases. For comparison with the very recently published algorithms, the HABC algorithm obtained a superior result in solving the largest problem than the ACO+LS proposed by Liouane (2006). Particle swarm optimization (PSO) is an efficient swarm intelligent algorithm and the experimental results obtained by PSO+SA and PSO+TS are considered as the competitive results for the FJSP. Table 1 shows that the HABC algorithm outperforms the PSO+SA algorithm in 2 out of 3 problems. The HABC algorithm can obtain the same experimental results with the PSO+TS in very short computational times. For example, in solving the largest problem 15×10, our algorithm consumes just about 50 seconds to reach the best result so far.

Table 1: Comparison of the five instances with single objective (makespan)

Problem Set	AL+CGA	GENACE	ACO+LS	PSO+SA	PSO+TS	HABC
4×5	16	11	11	-	-	11
8×8	15	-	-	15	14	14
10×7	15	12	11	-	-	11
10×10	7	7	7	7	7	7
15×10	23	12	12	12	11	11

"-" denotes not given by the author

The multi objective five instances

Table 2 shows the comparison of the results on the five multi-objective FJSP instances. The three objectives are considered simultaneously, i.e. minimization of the makespan (denoted by f_1),the total workload (denoted by f_2), and the maximal workload (denoted by f_3). It can bee seen from Table 2 that the HABC algorithm is competitive to other algorithms. The experimental results of the proposed algorithm dominate the results of the AL+CGA for solving the four instances. For comparison with the very recently published algorithms, the HABC can either obtain more non-dominated solutions or obtain superior result than PSO+TS and PSO+SA algorithms. For example, our algorithm obtain three non-dominated solutions in solving the 8×8 instance,while the PSO+SA and PSO+TS can only obtain two results. In addition, our algorithm obtains all these results in a run while the other algorithms can obtain only one result in a run. In other words, the other algorithm should run several times to obtain different results for an instance. Therefore, the external Pareto archive can enhance the population diversity of our algorithm. Fig. 1 shows the Gantt chart of the resulted solution of 15×10 instance, due to our proposed algorithm.

To make a further comparison with the ACO+LS proposed by Liouane (2006), we also test the problem listed in the paper [6]. The problem is given in Table 3. The comparison of the experimental results from ACO+LS and our algorithm are given in Table 4 and the two solutions obtained by the HABC algorithm are given in Table 5 and Table 6, respectively. It can be seen from Table 4 that the HABC algorithm obtained two non-dominated solutions for the example benchmark while the TS method in literature [6]can obtained only one solution. Furthermore, the resulted solutions obtained by our algorithm dominated all the results by the ACO method. In addition, our algorithm obtained the two non-dominated solutions consuming just only 0.01 seconds for the example benchmark. Therefore, Table 4 concludes that the proposed algorithm is efficient in solving the example problem especially when compared with the ACO method.

6 Conclusion

In this paper, we have proposed an efficient algorithm for solving multi-objective FJSPs. Instead of applying the basic ABC algorithm, we developed a hybrid ABC method. To memory the non-dominated solutions found so far and increase the population diversity, we presented an external Pareto archive set. A fast Pareto update function is also introduced in the algorithm to enhance the computational capability. In the hybrid algorithm, the balance of the capability of exploration and exploitation is considered. Experimental results on several well-known benchmarks show that our algorithm is competitive to other recently published algorithms for solving the FJSPs. The future work is to improve the neighborhood structure of the problem considered

Table 2: Comparison of the five instances with three objectives

		AL+CGA		PSO+SA		PSO+TS		HABC		
4×5	f_1	16				11		11	12	13
	f_2	34				32		32	32	33
	f_3	10				10		10	8	7
8×8	f_1	15	16	15	16	14	15	14	15	16
	f_2	79	75	75	73	77	75	77	75	73
	f_3	13	13	12	13	12	12	12	12	13
10×7	f_1							11	12	
	f_2							61	60	
	f_3							11	12	
10×10	f_1	7		7		7		8	7	8
	f_2	45		44		43		41	42	43
	f_3	5		6		6		7	6	5
15×10	f_1	23	24	12		11		12	11	
	f_2	95	91	91		93		91	93	
	f_3	11	11	11		11		11	11	

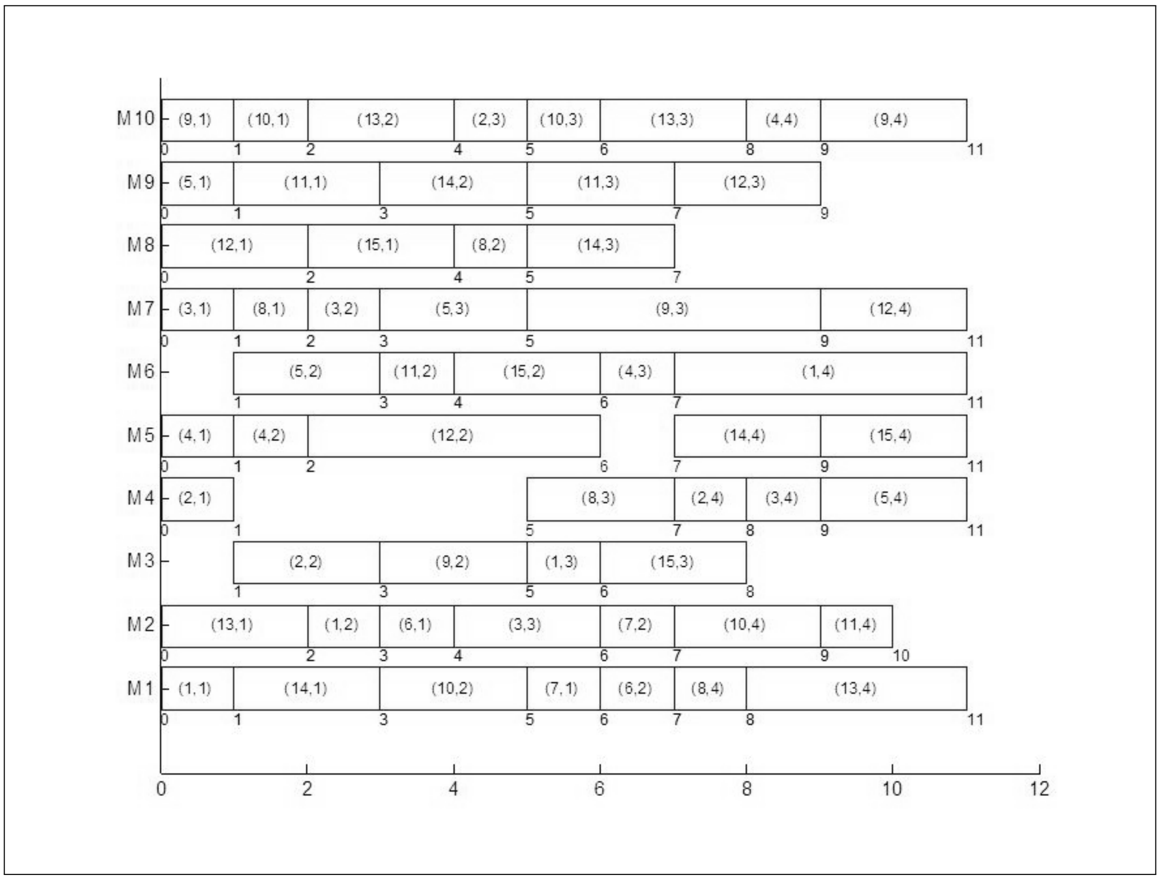

Figure 1: The Gantt chart of the solution 2 (f_1=11, f_2=93, f_3=11) for the instance 15×10

Table 3: Example benchmark 3 jobs-6 machines from [6]

		M_1	M_2	M_3	M_4	M_5	M_6
J_1	O_{11}	10	7	6	13	5	1
	O_{12}	4	5	8	12	7	11
	O_{13}	9	5	6	12	6	17
	O_{14}	7	8	4	10	15	3
J_2	O_{21}	15	12	8	6	10	19
	O_{22}	9	5	7	13	14	7
	O_{23}	14	13	14	20	8	17
J_2	O_{31}	7	16	5	11	17	9
	O_{32}	9	16	8	11	6	3
	O_{33}	6	14	8	18	21	14

Table 4: Comparison of the example benchmark

	f_1	f_2	f_3	
Lower Bound Value	18	45	8	Avgtime(s)
ACO+LS	19	51	13	
	19	50	13	
	19	48	14	
	19	47	14	
TS	18	45	12	-
HABC	18	45	12	0.01
	19	46	10	

"-" denotes not given by the author

Table 5: Solution 1 for the example benchmark (f_1=18, f_2=45, f_3=12)

	O_1	O_2	O_3	O_4
J_1	M_6:[0,1]	M_1:[1,5]	M_2:[5,10]	M_6:[10,13]
J_2	M_4:[0,6]	M_5:[6,10]	M_5:[10,18]	***
J_3	M_3:[0,5]	M_6:[5,8]	M_1:[8,14]	***

Table 6: Solution 2 for the example benchmark (f_1=19, f_2=46, f_3=10)

	O_1	O_2	O_3	O_4
J_1	M_6:[0,1]	M_1:[1,5]	M_2:[11,16]	M_6:[16,19]
J_2	M_4:[0,6]	M_5:[6,11]	M_5:[11,19]	***
J_3	M_3:[0,5]	M_6:[5,8]	M_1:[8,14]	***

and enhance the convergence ability of the algorithm.

7 Acknowledgments

This work supported by National Science Foundation of China under Grants 60874075, Science Research and Development of Provincial Department of Public Education of Shandong under Grant J08LJ20, J09LG29, J08LJ59, Soft Science Foundation of Shandong under Grant 2009RKB125.

Bibliography

[1] P. Brandimarte. Routing and scheduling in a flexible job shop by tabu search.*Annals of Operations Research* , Vol 22, pp. 158-183, 1993.

[2] M. Mastrolilli and L. M. Gambardella. Effective neighborhood functions for the flexible job shop problem.*Journal of Scheduling* , Vol. 3(1), pp. 3-20, 2000.

[3] J. Q. Li, Q. K. Pan, P. N. Suganthan and T. J. Chua. A hybrid tabu search algorithm with an efficient neighborhood structure for the flexible job shop scheduling problem. *International Journal of Advanced Manufacturing Technology.* doi: 10.1007/s00170-010-2743-y.

[4] L. Gao, C. Y. Peng, C. Zhou and P. G. Li. Solving flexible job shop scheduling problem using general particle swarm optimization. *In Proceedings of the 36th CIE Conference on Computers & Industrial Engineering*, pp. 3018-3027, 2006.

[5] J. Q. Li, Q. K. Pan and S. X. Xie. A hybrid variable neighborhood search algorithm for solving multi-objective flexible job shop problems. *ComSIS Computer Science of Software Information and System.* doi: 10.2298/CSIS090608017L.

[6] N. Liouane, I. Saad, S. Hammadi and P. Borne. Ant Systems & Local Search Optimization for Flexible Job-Shop Scheduling Production. *International Journal of Computers, Communications & Control*, Vol. 2, pp. 174-184, 2007.

[7] J. Gao, L. Sun and M. Gen. A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems.*Computers & Operations Research*, Vol. 35(9), pp. 2892-2907, 2008.

[8] F. Pezzella, G. Morganti and G. Ciaschetti. A genetic algorithm for the Flexible Job-shop Scheduling Problem.*Computers & Operations Research* , Vol. 35, pp. 3202-3212, 2008.

[9] I. Kacem, S. Hammadi and P. Borne. Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic. *Mathematics and Computers in Simulation*, Vol. 60, pp. 245-276, 2002a.

[10] I. Kacem, S. Hammadi and P. Borne. Approach by localization and multi-objective evolutionary optimization for flexible job-shop scheduling problems,*IEEE Transactions on Systems* , Man and Cybernetics, Part C, Vol. 32(1), pp. 408-419, 2002b.

[11] W. J. Xia and Z. M. Wu. An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems.*Computers & Industrial Engineering* , Vol. 48(2), pp. 409-425, 2005.

[12] G. H. Zhang, X. Y. Shao, P. G. Li and L. Gao. An effective hybrid swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. *Computers & Industrial Engineering*, Vol. 56(4), pp. 1309-1318, 2009.

[13] N. B. Ho and J. C. Tay. Solving Multiple-Objective Flexible Job Shop Problems by Evolution and Local Search. IEEE Transactions on Systems, Man, and Cybernetics, Part C:*Applications and Reviews* , Vol. 38(5), pp. 674-685, 2008.

[14] D. Karaboga. An idea based on honey bee swarm for numerical optimization. Technical Report TR06. *Computer Engineering Department. Erciyes University.* Turkey. 2005.

[15] D. Karaboga and B. Basturk. A powerful and efficient algorithm for numerical function optimization: Artificial Bee Colony (ABC) algorithm.*Journal of Global Optimization* , Vol. 39(3), pp. 459-171, 2007.

[16] D. Karaboga and B. Basturk. On The performance of Artificial Bee Colony (ABC) algorithm.*Applied Soft Computing* , Vol. 8(1), pp. 687-697, 2008.

[17] D. Karaboga and B. Akay. A comparative study of Artificial Bee Colony Algorithm. *Applied Mathematics and Computation*,Vol. 214, pp. 108-132, 2009.

[18] Q. K. Pan, M. F. Tasgetiren, P. N. Suganthan and T. J. Chua. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. Information Sciences. doi: 10.1016/j.ins.2009.12.025, 2010.

[19] J. Q. Li, Q. K. Pan and Y.C. Liang. An effective hybrid tabu search algorithm for multi-objective flexible job shop scheduling problems. *Computers & Industrial Engineering*, Vol. 59, pp. 647-662, 2010b.

[20] F. Pezzella, G. Morganti and G. Ciaschetti. A genetic algorithm for the Flexible Job-shop Scheduling Problem. *Computers & Operations Research*, Vol. 35, pp. 3202-3212, 2008.

[21] Q. K. Pan, L. Wang, B. Qian. A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems.*Computers & Operations Research*, Vol. 36(8), pp. 2498 -2511, 2009.

[22] L. Wang. Shop scheduling with genetic algorithms. *Tsinghua university press*, Beijing, China, 2003.

[23] K. Deb, A. Paratap, S. Agarwal and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutinary Computation*, Vol. 6(2), pp. 182-197, 2002.

[24] N. B. Ho and J. C. Tay. GENACE: An Efficient Cultural Algorithm for solving the Flexible Job-Shop Problem. *In Proceedings of the IEEE Congress on Evolutionary Computation (CEC2004) Piscataway*, pp.1759-1766, 2004.