

Scheduling a Flexible Flow Shops Problem using DEA

Fatemeh Dadkhah, Hossein Ali Akbarpour

Abstract—This paper considers a scheduling problem in flexible flow shops environment with the aim of minimizing two important criteria including makespan and cumulative tardiness of jobs. Since the proposed problem is known as an Np-hard problem in literature, we have to develop a meta-heuristic to solve it. We considered general structure of Genetic Algorithm (GA) and developed a new version of that based on Data Envelopment Analysis (DEA). Two objective functions assumed as two different inputs for each Decision Making Unit (DMU). In this paper we focused on efficiency score of DMUs and efficient frontier concept in DEA technique. After introducing the method we defined two different scenarios with considering two types of mutation operator. Also we provided an experimental design with some computational results to show the performance of algorithm. The results show that the algorithm implements in a reasonable time.

Keywords—Data envelopment analysis, Efficiency, Flexible flow shops, Genetic algorithm

I. INTRODUCTION

TRADITIONAL manufacturing systems have taken many general forms. In increasingly complex manufacturing environments, more complex manufacturing systems have been created in order to address such factors as limited capacity and complicated process plans [1, 2]. The scheduling objective in such industries may vary, e.g., makespan, tardiness, earliness, etc. In flexible flow shops every job must be processed on at most one machine per stage. A flexible flow shop consists of several stages in series. A job may not revisit a stage that it has already visited. Each stage has at least one machine, and at least one stage must have more than one machine [1]. In the proposed scheduling problem we couldn't achieve the optimal solution by use of exact methods. The meta-heuristics are developed in literature as efficient methods to achieve the nearest solutions to optimal solutions. Many researchers worked on these methods and they succeeded to develop many different methods.

In 2002, Deb and his colleagues [3] developed a fast method based on genetic algorithm called Non-dominated Sorting Genetic Algorithm II (NSGAI). In this method, solutions are sorted in different sets based on their non domination, at first. Then in each set, solutions with minimum distance related to the others have more chance be selected and construct next generation. That is, distance criterion is so important to carry out crossover.

Ruiz Torres and Lopez considered problem of scheduling jobs on parallel machines in multi criteria environment [4]. They decided to minimize the makespan and the number of tardy jobs, simultaneously. To achieve this aim, they focused on simulated annealing algorithm and developed four different methods based on different initial solutions derived on benchmark. For evaluating the performance of proposed algorithms and identifying the most efficient algorithm, they used FDH formulation of DEA. Tavakkoli-Moghaddam and his colleague investigated a multi-objective model for a no-wait flow shop scheduling problem which minimizes both the weighted mean completion time and weighted mean tardiness [5]. They proposed a meta-heuristic based on immune system, hybrid multi-objective immune algorithm (HMOIA), to find optimal solutions. In order to evaluate performance of proposed algorithm, they compared HMOIA with five different methods from benchmark in large size problems.

There are many due date related important criteria which we considered cumulative tardiness penalty as the most important objective. The makespan criterion has been used by many researchers and has been selected for this research. Scheduling to minimize makespan and cumulative tardiness in flexible flow shops with multiple parallel machines and jobs that may skip stages is the focus of this paper. In this research we concentrated on GA and developed a new version of that based on DEA concept. We considered makespan and cumulative tardiness of jobs as two objective functions which we are going to minimize them by proposed algorithm, simultaneously.

The following of paper organized in four different sections. In section II a DEA background and its different features are presented. Also we provided a numerical example to better illustration of the proposed algorithm. General structure of GA and its developing is presented in section III. An experimental design including data generation, parameters setting and computational results is provided in section IV. The conclusion of paper and some future works are presented in section V.

II. DATA ENVELOPMENT ANALYSIS

General structure of DEA has been introduced by Farrell in 1954 for the first time. Based on his article some researchers worked on this new concept and developed two models; CCR by Charnes, Cooper and Rhodes in 1978 [6] and BCC by Banker, Charnes and Cooper in 1984 [7]. Now there are other models such as FDH, BCC-CCR and CCR-BCC. But the BCC and CCR models are the basic models in DEA. The DEA is a linear programming based method which evaluates relative efficiency of Decision Making Units (DMUs). It can include

F. Dadkhah is with the Department of Mathematic, Marvdasht Branch, Islamic Azad University, Marvdasht, Iran (e-mail: dadkhah_f@miau.ac.ir).

H. A. Akbarpour is with the Department of Engineering, Marvdasht Branch, Islamic Azad University, Marvdasht, Iran (e-mail: akbarpour.ha@gmail.com).

multiple outputs and inputs without a priori weights and without requiring explicit specification of functional forms between inputs and outputs. It computes a scalar measure of efficiency and determines efficient levels of inputs and outputs for each DMU under evaluation which has a range of zero to one [8]. In fact, the DEA solves a linear programming to evaluate efficiency score of different decision making units relatively. Each DMU can have some inputs and outputs with different weights. In case of one input and one output one can divide value of output by value of input for evaluating the efficiency score of an especial DMU. But, in real management problems usually there are many different parameters either with or without specific weights which effect the determination of efficiency score of a DMU. In this case one should challenge with a decision making problem. In this research, in order to triumph on formed decision making problem, we used BCC input oriented model of DEA technique. We considered each individual of population in proposed GA (each chromosome) as a DMU. In DEA each DMU can have input and output one or more. In proposed GA we supposed that each DMU has two inputs which are makespan and cumulative tardiness. Also we supposed that all DMUs have identical outputs, all of them give us processed jobs. In order to employ the DEA technique in GA, we provided a numerical example and illustrated efficiency score, efficient frontier and ranking of DMUs.

TABLE I
INPUTS AND OUTPUT OF DMUS

DMU	A	B	C	D	E
Input1	2	5	3	4	6
Input2	5	2	2	4	1
Output	1	1	1	1	1

TABLE II
EFFICIENCY SCORE OF DMUS

DMU	A	B	C	D	E
Efficiency	1	0.82	1	0.73	1

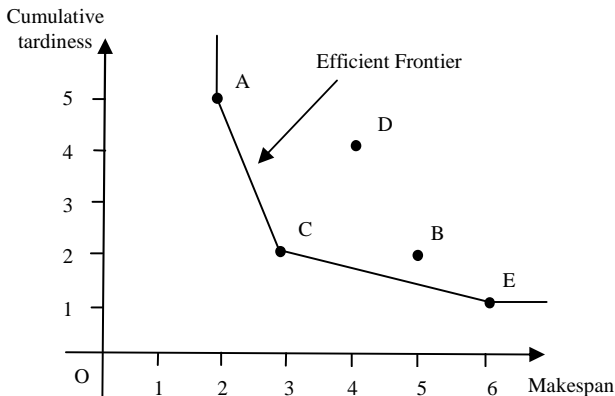


Fig. 1 BCC input oriented efficient frontier generated from observed data

Example: Suppose that five DMUs A, B, C, D and E with identical outputs and two different inputs achieved from five various chromosomes, are as Table I. Table II shows the efficiency score of each DMU according to BCC model. Also we provided the efficient frontier in Fig. 1.

In order for ranking the DMUs, first, we divided the makespan of each DMU by the tardiness of it and then, sort them based on the minimum distance to one, e.g., rank of the C is better than rank of the A.

III. DEVELOPED GENETIC ALGORITHM

GA is the first meta-heuristic method developed based on natural genetic science of body. The GA is a guided local search based algorithm by genetic operators which tries to find an optimal solution in limited iterations. Genetic operators are the most important factors to improve the method and determine diversity and intensity. Those are reproduction, mutation and crossover. The reproduction operates based on elitism strategy and usually transfers 25 percent of populations with high quality to the next generation with no change. The mutation always operates on less than 5 percent of populations for diversifying and prevents from stick in a local optimality. About 70 percent of populations are operated by crossover in order to produce better populations. Usually the last operator carries out its operation based on a selection strategy.

In this research we developed a new genetic algorithm based on data envelopment analysis. The chromosome scheme, which we used in this paper, adopted from random key genetic algorithm (RKGGA) proposed by [1, 2]. We used efficiency concept to sort the individuals in each population. Furthermore, we used roulette wheel selection strategy for crossover. We provided the general structure of considered algorithm as following.

- Step 1: Initial population creation
- Step 2: fitness function evaluation and chromosome ordering based on DEA efficiency
- Step 3: While (termination condition is not met) do the following:
 - a) Next generation construction by genetic operators
 - b) Updating the chromosomes ordering based on DEA efficiency
- Step 4: Returning the best solutions found on efficient frontier

In step 1 we produce an initial generation random completely. Each chromosome acts as a DMU with two inputs, makespan and cumulative tardiness, and just one output, 1. Then in step 2 we calculate the value of efficiency of each DMU in BCC input oriented model as the fitness function of each chromosome. After ordering the chromosomes, we carry out iterative step 3. At first by using the genetic operators, i.e., crossover, mutation and reproduction, we construct the next generation. We use roulette wheel strategy to select candidates. Then we refresh the chromosomes ordering and prepare them for constructing the next generation. We apply

these two stages until condition criterion is met. The condition criterion can be based on max generation or the CPU time which we consider the max generation for this algorithm. Eventually, proposed algorithm returns the best solutions which located on efficient frontier with efficiency score equals to one. Of course, we defined two scenarios based on two different types of mutation operator. We considered inverse mutation as the first scenario and pair-wise exchange mutation with cyclic exchange for the second. As shown in the Fig. 2, the inverse mutation operator inverses sequence of jobs in each chromosome. Also, the pair-wise exchange mutation operates on a chromosome as the Fig. 3.

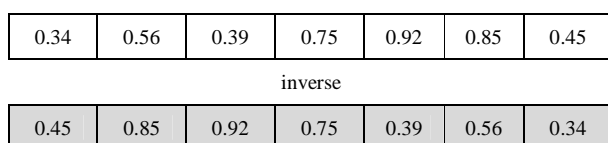


Fig. 2 Inverse mutation

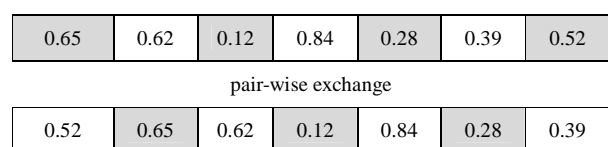


Fig. 3 Pair-wise Exchange with Cyclic Exchange

IV. DESIGN OF EXPERIMENT

A. Data Generation

In this paper we generated experimental data to evaluate and compare the performance of considered meta-heuristic. Required data to define proposed problem are problem size, jobs processing times, skipping probability and flexibility level. In flexible flow shops environment problem size is defined based on two important parameters: number of jobs and stages. However, in flexible flow shops the representative parameter for number of machines is number of stages, so some stages can have parallel machines. Three problem sizes are named by small, medium and large. The ranges of each parameter to determine the problem size are as follows. Number of stages involves 3, 4 and 5 for small, 7, 8 and 9 for medium and 15, 18 and 20 for large problems. Moreover, this value involves 5, 7 and 9, 15, 18 and 20 and 40, 45 and 50 for the number of jobs. Flexibility level divided into three levels: low flexibility as that being represented by 1/3 of stages

TABLE III
FACTOR LEVELS

Factor	Levels
Number of Stages	3, 4, 5(s), 7, 8, 9(m), 15, 18, 20(l) *
Number of Jobs	5, 7, 9(s), 15, 18, 20(m), 40, 45, 50(l)
Processing times	U (25, 50) : Pr(0.8) U (5, 75) : Pr(0.2)
Skipping probability	0.1
Flexibility	1/3, 2/3, 3/3

* s: small problem, m: medium problem, l: large problem

having parallel machines, medium flexibility by 2/3 of stages having parallel machines and high flexibility by all stages being parallel [9, 10]. To determine the number of flexible stages in flexibility levels 1 and 2 we round up to greater integer number. If a stage is recognized parallel, then we would determine its machine numbers by integer numbers 2 or 3, completely random and identical probability. We provided two different ranges for processing time of jobs. In order for selection of range, we produce a random number from uniform distribution U(0, 1), if that greater than or equal to 0.2 we would select the range 1, otherwise range 2. There is skipping characteristic in flexible flow shop problems. Meaning that every job processing time is may be equal to zero by specific possibility. In this paper we used this concept by probability 0.1. For each job, we generated r from U(0, 1); if $r < 0.1$ then processing time of this job is equal to zero. All important factors and their levels are represented in Table III.

According to these explanations we could produce 27 test problems for each size of small, medium and large problem and in general there are 81 test problems that each of them iterates ten independent replicates.

TABLE IV
PROPOSED ALGORITHM FACTOR LEVEL

Factor	Levels
Percent of crossover (pc)	0.66, 0.70, 0.74
Percent of mutation (pm)	0.01, 0.03, 0.05
Number of generation ($ngen$)	25, 50, 75 (s)
	50, 100, 150 (m)
	150, 200, 300 (l)
Number of initial population ($popsiz$ e)	100

B. Parameters Setting

Performance of each algorithm is affected by some various parameters, significantly. If these values aren't selected correctly, appropriate results won't obtain. In order to select the parameters that result in solution with high quality, we considered problems in three different sizes that described before and selected some problems as a sample in each size. Sample sizes are 6 for small, medium and large problems. In this paper we considered some of the important factors with different levels for proposed algorithm. These factors and their levels are shown in Table IV.

We run proposed algorithm ten independent replicates, by combination of different factors represented in Table 4 and selection the best combination according to results of them. Minimizing both of the two considered objective functions simultaneously is our measurement. However, the CPU time is an important criterion to realize the best factor values. After tuning all parameters except $ngen$, we fixed the best obtained parameter values and found the best value of $ngen$. The obtained values for every factor in all three different sizes are shown in Table V.

TABLE V
BEST VALUES FOR PROPOSED ALGORITHMS PARAMETERS

Size	<i>pc</i>	<i>pm</i>	<i>ngen</i>
Small	0.70	0.05	50
Medium	0.70	0.05	150
Large	0.70	0.05	300

C. Computational Results

The results of proposed algorithm performance to solve considered problem is presented in this section. Both of two scenarios are coded in *MATLAB 7.1* and are carried out 10 independent runs. Every run records all the non repeated Pareto optimal solutions. Scenarios run on a PC with a Pentium IV 3.0 GHz processor with 512 MB of RAM and Windows Xp professional operating system.

In order to measure the performance of presented algorithm, we considered three criteria as MID, RAS and CPU time. The CPU time is a known criterion; therefore, we explained the two others as (1) and (2).

$$MID = \frac{\sum_{i=1}^n c_i}{n} \quad \ni \quad c_i = \sqrt{f_{1i}^2 + f_{2i}^2} \quad (1)$$

$$RAS = \frac{\sum_{i=1}^n \left(\frac{f_{i1} - F_i}{F_i} \right) + \left(\frac{f_{i2} - F_i}{F_i} \right)}{n}, \quad F_i = \min \{f_{1i}, f_{2i}\} \quad (2)$$

TABLE VI
RAS CRITERION COMPARISON

Size	Scenario1		Scenario2	
	Mean	St. Dev	Mean	St. Dev
Small	3.546	6.161	2.017	5.049
Medium	5.204	2.132	4.069	1.534
Large	15.905	5.701	14.140	3.177

TABLE VII
MID CRITERION COMPARISON

Size	Scenario1		Scenario2	
	Mean	St. Dev	Mean	St. Dev
Small	6025	5802	4170	6107
Medium	30184	13150	28609	15306
Large	207685	76708	191176	74079

TABLE VIII
CPU TIME CRITERION COMPARISON

Size	Scenario1		Scenario2	
	Mean	St. Dev	Mean	St. Dev
Small	1.2628	0.2477	1.2640	0.2555
Medium	11.332	1.538	11.326	1.513
Large	105.97	16.83	107.10	17.79

According to (1) and (2), the smallest value of MID or RAS criterion shows the best performance. In order to evaluate the performance of proposed algorithm we selected nondominated solutions which positioned on efficient frontier with efficiency score of 1. Then we calculated the MID and the RAS of all these solutions using (1) and (2).

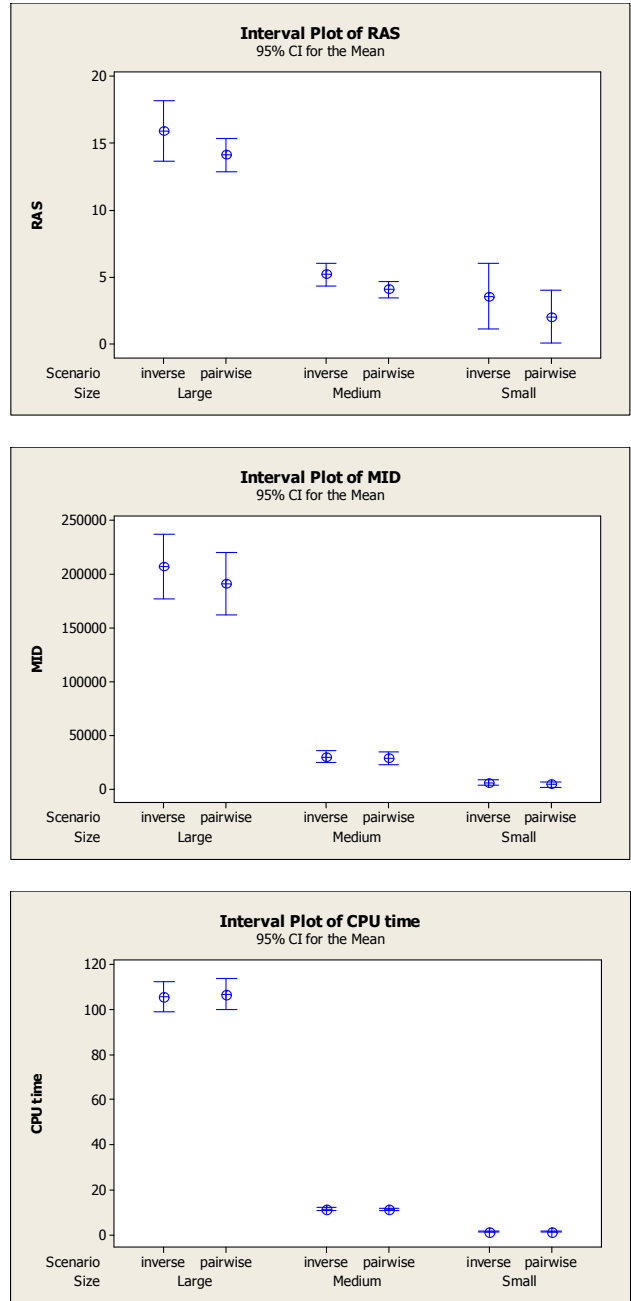


Fig. 3 Interval plot of RAS, MID and CPU time criteria in three sizes

Since the performance of meta-heuristics depends on used parameters, intensely, the elimination of Pareto archive set number is an important advantage which is performed by DEA in proposed algorithm. This feature causes not to need to determine number of optimal solutions for evaluating of algorithm performance.

We compared two different scenarios performance by two criteria introduced in (1) and (2) and provided the results in Tables VI and VII. The results show that there are no significantly differences between two scenarios. Also the Table VIII presents the performance of two scenarios in CPU time for considered different size which show proposed algorithm run in a reasonable time. In Fig. 3, presented the interval plot of RAS, MID and CPU time acquired from Minitab 16 statistical software in all different sizes.

V. CONCLUSION

In presented research we considered a multi criteria scheduling problem in flexible flow shops to minimize makespan and cumulative tardiness of jobs. We proposed a genetic algorithm as an efficient meta-heuristic and developed a new method based on DEA. For each DMU we considered two different inputs and an identical output. Efficient DMUs with efficiency score of one, which located on efficient frontier, have more chance to construct the next generation. We evaluated performance of proposed algorithm by producing empirical experimental data in three different sizes: small, medium and large. Then we presented obtained computational results. The results show that the algorithm implements in a reasonable time and it can compete with other meta-heuristics. For future work DEA can apply on immune algorithm, ant colony optimization and compare with proposed algorithm.

REFERENCES

- [1] M. E. Kurz, and R.G. Askin, "Comparing scheduling rules for flexible flow lines," *International Journal of Production Economics*, 2003, 85, 371–388.
- [2] M. E. Kurz, and R.G. Askin, "Scheduling flexible flow lines with sequence-dependent setup times," *European Journal of Operational Research*, 2004, 159(1), 66–82.
- [3] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, 2002, 6(2), 182–197.
- [4] A. J. Ruiz-Torres, and F. J. López, "Using the FDH formulation of DEA to evaluate a multi-criteria problem in parallel machine scheduling," *Computers & Industrial Engineering*, 2004, 47, 107–121.
- [5] R. Tavakkoli-Moghaddam, A. Rahimi-Vahed, A. H. Mirzaei, "A hybrid multi-objective immune algorithm for a flow shop scheduling problem with bi-objectives: Weighted mean completion time and weighted mean tardiness," *Information Science*, 2007, 177, 5072–5090.
- [6] A. Charnes, W.W. Cooper, and E. Rhodes, "Measuring the efficiency of decision making units," *European Journal of Operation Research*, 1978, 429–444.
- [7] R. D. Banker, A. Charnes, and W.W. Cooper, "Some models for estimating technical and scale inefficiencies in data envelopment analysis," *Management Science*, 1984, 30, 1078–1092.
- [8] M. R. Alirezaee and M. Afsharian, "A complete ranking of DMUs using restrictions in DEA models," *Applied Mathematics and Computation*, 2007, 189, 1550–1559.
- [9] R. Logendran, S. Carson, and E. Hanson, "Group scheduling in flexible flow shops," *International Journal of Production Economics*, 2005, 96(2), 143–155.
- [10] R. Logendran, S. Carson, and E. Hanson, P. deSzoek, and F. Barnard "Sequence-dependent group scheduling problems in flexible flow shops," *International Journal of Production Economics*, 2006, 102, 66–86.