Scheduling of Unrelated Parallel Machines using Two Multi Objective Genetic Algorithms with Sequence-Dependent Setup Times and Precedent Constraints

S. Safaei
Department of Industrial Management,
Shahid Beheshti University, Iran
E-mail: sahar.safaie@gmail.com

R. Naderi*
Department of Industrial Management,
Semnan University, Iran
E-mail: rh.naderi@gmail.com
*Corresponding author

A. Sohrabi
Department of Molecular Biology, Research Center of Health Reference Laboratory, Ministry of Health and Medical Education, Iran
E-mail: sohrabi58@gmail.com

A. Hatami
Department of Business Management,
Farabi College, University of Tehran, Iran
E-mail: hatami.am75@gmail.com

Received: 8 May 2015, Revised: 22 August 2015, Accepted: 27 September 2015

Abstract: This paper considers the problem of scheduling N jobs on M unrelated parallel machines with sequence-dependent setup times. To better comply with industrial situations, jobs have varying due dates and ready times and there are some precedence relations between them. Furthermore sequence-dependent setup times and anticipatory setups are included in the proposed model. Our objective is to determine a schedule that minimizes makespan and number of tardy jobs. The problem is NP-hard, so for obtaining an optimal solution, in reasonable computational time, we propose two multi objective genetic algorithms (MOGA). To evaluate the proposed algorithms, random test problems are produced in medium and large sizes with tight due dates. After setting the parameters, the performances of these algorithms are evaluated using the concept of data envelopment analysis (DEA), distance method, and a number of non-dominated solutions.

Keywords: Data Envelopment Analysis, Genetic Algorithm, Makespan, Multi-Objective, Number of Tardy Job, Parallel Machine Scheduling, Precedence Constraints, Sequence-Dependent Setup Times, Topsis Method

Reference: Safaei, S., Naderi, R., Sohrabi, A. and Hatami, A. "scheduling of unrelated parallel machines using two multi objective genetic algorithm with sequence-dependent setup times and precedent constraints", Int J of Advanced Design and Manufacturing Technology, Vol. 8/ No. 4, 2015, pp. 63–74.

Biographical notes: S. Safaei received her BSc in Industrial Management from Allameh Tabatabaie University, Iran in 2006, and her MSc in Industrial Management from Shahid Beheshti University, Iran in 2009. **R. Naderi** received her MSc in Industrial Management from Shahid Beheshti University, Iran in 2009 and she is currently PhD student at the department of Industrial Management in Semnan University, Iran. **A. Sohrabi** is an Assistant Professor in Molecular Medicine in Research Center of Health Reference Laboratory, Ministry of Health and Medical Education, Iran. **A. Hatami** is MSc student in Business Management department of Farabi College, University of Tehran, Iran.

1 INTRODUCTION

The problem of scheduling N jobs on M unrelated parallel machines is considered with sequence-dependent setup times to minimize makespan and number of tardy jobs. Parallel machine scheduling is important from both theoretical and practical points of view. From a theoretical viewpoint, it is important because many algorithms can be reduced to solve single machine problems and from practical viewpoint, it is important because in a real world manufacturing environment most workshops have more than one machine [1].

The classical parallel machine problem is defined as: A set of the independent jobs to be processed on a number of available identical parallel machines. Each machine can process only one job at a specific time, and each job can be processed on one machine. Each job is ready at the beginning of the scheduling horizon and has a distinct processing time and due date. It is assumed that machines are identical, all jobs are available at the beginning of scheduling and have common due dates [2]. According to reference [3], when machines are not identical and cannot be completely correlated by simple rate adjustments, they are named unrelated parallel machines. In this work, we have considered unrelated parallel machine with varying ready time and due dates. Many scheduling researches assume that setup times are negligible or part of the processing time. While, this assumption simplifies the analysis and/or reflects certain applications, it adversely affects the solution quality for many applications which require explicit treatment of setup times, and cause the model not to be effective in real environments.

In a manufacturing environment, setup times consist of all activities that are done on material in order to prepare machines and situations in process phase. Setup times can be divided into two important groups; sequence-dependent setup times and sequence-independent setup times [4]. On the other hand it can be divided as anticipatory and non-anticipatory setups. A setup is anticipatory if it can be started before the corresponding job becomes available on the machine. Otherwise, a setup is non- anticipatory [5]. Almost in all real industrial environments, precedence constraints are essential for sequencing and jobs scheduling. In most real situations, some cases occur in which the beginning of a job is at the completion of another job, and if the precedent job is not completed, the following job cannot be started. Here, in order to be more realistic, we consider sequence-dependent setup times, anticipatory setup and also precedence constraints. In a real manufacturing environment, several objectives frequently need to be considered at the same time. In reality, managers consider multiple objectives and try to find the best solution that meets the considerations. But in many situations, one is confronted with conditions that delay in delivery result in customers cancelling their orders. Although, many researchers have devoted considerable research efforts to minimize the total completion time of jobs on parallel machines, minimizing the number of tardy jobs is one of the objectives that have been considered less by previous researches. Decreasing the completion times can cause reducing job lateness and tardiness, and also leads to reduction in the total works-in-process inventories, and total completion costs. These two conflicting objectives can be considered together as the objectives of the problem. Reference [6] shows that scheduling jobs on two identical machines to minimize the makespan is NP-hard. As identified by [7] when the number of machines is greater than two, the problem is an even strong NP-hard. So, two multi objective genetic algorithms (MOGA) have been proposed to solve the bi-objective problem in this article. To evaluate the proposed algorithms, random test problems are produced in medium and large sizes with tight due dates, also the performances of these algorithms are evaluated using the concept of data envelopment analysis (DEA), distance method, and a number of non-dominated solutions which are another novelty aspects of this research.

2 LITERATURE REVIEW

The literature of parallel machine-scheduling problems with conventional performance measures based on due date, flow time and completion time has been widely reviewed by Cheng and Sin [8], On the other hand, Lam and Xing presented a review which focused on parallel machine scheduling problems with non-regular performance measures as a result of incorporating the concepts associated with flexible manufacturing systems and just-in-time manufacturing [9].

The research on unrelated parallel machine scheduling has focused on a variety of objectives; such as minimizing the makespan, maximum tardiness, total weighted tardiness and total weighted flow time. For makespan minimization, these include the algorithms with worst case performance [10], approximation algorithm [11], two-phase heuristics [12], local search heuristics [13], [14], the consideration of dynamic machine availability [15], and an effective heuristic through hashing [16]. So far, Uzsoy et al. discussed minimizing the maximum lateness with precedence constraints and sequence-dependency of jobs, in their article a neighbourhood search algorithm that obtained local optimal solution was presented along with a

branch and bound algorithm to obtain optimal solutions [17]. Hurink and Knust have presented scheduling of jobs on identical parallel machines with sequence-dependent setup times and precedence constraints to minimize maximum completion time, as an NP_hard problem. In this paper, they investigate designing an efficient list scheduling algorithm which produces a dominant set of list schedules if it is applied to all sequences of jobs which are compatible with the specified precedence [18]. Researches in the area of scheduling parallel machines with multiple objectives are as follows:

Shmoys and Tardos considered a problem of scheduling unrelated parallel machines to minimize makespan and total cost, they considered a range of possible processing times for each machine-job pair, and the cost linearly increased as the processing time decreased [19]. Suresh and Chaudhuri proposed a Tabu search method to solve a bi- criteria problem on unrelated parallel machines with the aim of minimizing makespan and maximum tardiness [20].

Reference [21] conducted a wide-ranging survey on bi-criteria parallel machines scheduling problems and presented several different heuristic algorithms to solve them. Reference [22] developed a fully polynomial approximation scheme for minimizing total cost and makespan. Funda et al employed the GA and considered the total weighted earliness and tardiness on the multi-objective scheduling problem of parallel machines [23]. Reference [24] proposed an efficient algorithm to solve optimally the bi-criteria problem of minimizing the weighted sum of makespan and mean flowtime on two identical parallel machines. Reference [25] also studied a scheduling problem to minimize flowtime subject to optimal makespan on two identical parallel machines. Gupta et al. discussed the multi-objective scheduling problem of parallel machines, which attempts to minimize the makespan under the consideration of minimizing the flowtime [26]. Yu et al. proposed a two-stage Lagrangian relaxation heuristic (LRH) method to solve the problem, and apply it to printed wiring board manufacturing [27]. Lin and Liao considered the identical parallel machines problem with makespan minimization subject to minimum total flowtime [28]. Reference [29] studied problems, they showed that it is possible to construct in polynomial time an approximate Pareto curve whenever the number of machines is unvarying. Instead of proceeding in a problem-by-problem basis, they identified a class of multi-objective optimization problems possessing a fully polynomial time approximation scheme (FPTAS) for computing an ε-approximate Pareto curve. Cochran et al. proposed a two-stage multi-population genetic algorithm (MPGA) to solve the parallel machine scheduling problem with two objectives of makespan and total weighted tardiness. They used the genetic algorithm to assign jobs to machines and then sequence them based on the different heuristics [30]. Huo et al. presented multi-objective parallel machines scheduling to minimize the number of tardy jobs and maximum weighted lateness, and solved this problem with heuristic algorithm [31].

Eren, T. developed a heuristic approach for minimization of the weighted sum of total completion time and total tardiness with a learning effect of setup times and removal times [32]. Jolai, F. et al. developed three bi-objective optimization methods based on simulated annealing, called CWSA (classical weighted simulated annealing), NWSA (normalized weighted simulated annealing), and FSA (fuzzy simulated annealing), to solve the problem with the goal of finding optimal pareto front and finally they proposed a new reliable method by mixing the Taguchi method and a Multi-objective Decision Making (MODM) approach. They used both small and large scale problems [33]. Lee, C., et al addressed a scheduling problem originated from the manufacturing plant and the objective is to generate a schedule of the jobs on the unrelated parallel machines to minimize the total completion time. They proposed three heuristics (DMH-1, DMH-2 and DMH-3) for solving the problem [34]. Min joo and kim used a hybrid genetic algorithms with three dispatching rules for large-sized problems. And for assessing the performance of algorithms, computational experiments were conducted [35]. Yang-Kuei Lin and Hao-Chen Lin Research propose a heuristic and a Tabu search algorithm for finding non-dominated solutions to bicriteria unrelated parallel machine scheduling problems with release dates [36].

3 MULTI-OBJECTIVE OPTIMIZATION

In the literature concerning multi-objective optimization problems five main approaches can be discerned:

i. Utility approach: a utility function or weighting function, often a weighted linear combination of the objectives is used to aggregate the considered objectives in a single one.
ii. Hierarchical approach: the considered objectives are ranked in a priority order and optimized in this order.
iii. Goal programming: all of the objectives are taken into account as constraints which express some satisfying levels (or goals) and the objective is to find a solution which provides a value as close as possible to the predefined goal for each objective. Sometimes one objective is chosen as the main objective and is optimized under the constraint related to other objectives.

iv. Interactive approach: at each step of the procedure, the decision maker expresses his preferences in regard to one (or several) solutions proposed, so that the method will progressively converge to a satisfying compromise among the considered objectives.
v. Simultaneous or Pareto approach: the aim is to generate, or to approximate in case of a heuristic method, the complete set of efficient solutions [37].

Many real-world problems engage simultaneous optimization of several objective functions. In general, these functions often compete and are in conflict with each other. Multi-objective optimization with such conflicting objective functions provides a set of optimal solutions that called Pareto optimal, rather than one optimal solution. The Multi-objective set includes solutions with no better solution with considering all objective functions. Without loss of generality, let us consider a typical multi-objective minimization problem with p decision variables and q objectives (q > 1):

$$\begin{aligned} &Miny = f(x) = (f_1(x), f_2(x), ..., f_q(x)) \\ &X \in R^P, and Y \in R^q \end{aligned} \quad (1)$$

Definition 3-1: Solution a, is said to dominate solution b if and only if:

$$\begin{aligned} &(1) f_i(a) \le \underline{f}_i(b); \forall i \in \{1, 2, ..., q\} \\ &(2) f_i(a) < f_i(b); \exists i \in \{1, 2, ..., q\} \end{aligned} \quad (2)$$

Solutions that dominate other solutions but do not dominate themselves are called non- dominated solutions.

Definition 3-2: Vector $\bar{a}$, is a globally Pareto-optimal solution if vector $\bar{b}$, does not exist such that $\bar{b}$, dominates $\bar{a}$ [38].

3.1. Evaluation methods

One important aspect that should be considered is how to evaluate the quality of the obtained non-dominated front. Several aspects have to be considered to determine how good the obtained front is. Some these methods are as follows:

1) The number of non-dominated solutions obtained,
2) The closeness between the obtained front and the Pareto optimal front (if known),
3) The coverage of the Pareto front, i.e. the spread and distribution of the non-dominated solutions [39].

In this paper, we use three methods to evaluate the quality of solutions.

1- The number of non-dominated solutions
2- Free disposal hull (FDH) method
3- Distance method

Section 3.1.1 and Section 3.1.2 describe the last two methods respectively.

3.1.1. FDH approach

Data envelopment analysis (DEA) is a non-parametric approach to measuring efficiency. Since its advent in 1978 [40], this method has been extensively utilized to analyze relative efficiency, and has covered a wide area of applications and theoretical extensions [41]. From the definition of efficiency, DEA will be defined as weighted sum of outputs divided by weighted sum of inputs. The FDH is a special case of DEA.

A score of one is assigned to a unit only when comparisons with other relevant units do not provide evidence of inefficiency in the use of any input or output, and a score less than one is assigned to relatively inefficient units and it means that a linear combination of other units from the sample, using a smaller vector of inputs could produce the same vector of outputs with a smaller vector of inputs.

One objective of this work is to compare the performance of our two proposed meta-heuristics. We do it by employing the FDH formulation and considering the corresponding degrees of efficiency. Further illustration of the application of FDH and its formulations can be seen in [42].

3.1.2. Distance method

Distance method, like previous method can evaluate the set of non-dominated solutions. In this method, after determining non-dominated solutions of each algorithm, the direct distance of these solutions in relation with the origin will be found. Here we suppose that the origin is (0, 0). The obtained average of these distances (D) is the radius of a bow in the space of solution. Each solution is along of a direct line which connects to the origin, which will be transferred to a point on the bow with the mentioned radius. In this way, the solutions will permanently keep their characteristic of being non-dominated. Now, we can infer, by analogy, these two sets of non-dominated solutions by considering the radius of the bow (D) which was obtained, or by considering the dispersion of point Where fi (x) and fi (x) are our objectives, d is the Euclidean distance between solution i and point (0, 0). The s and σ metric measure the uniformity of the spread of the points of the solution set.

$$d_i = \sqrt{f_1^{\,i}(\vec{X})^2 + f_2^{\,i}(\vec{X})^2} \quad (3)$$

$$d_i = \sqrt{(f_1^{\,i}(\vec{X}) - LB_{f1})^2 + (f_2^{\,i}(\vec{X}) - LB_{f2})^2} \quad (4)$$

$$\bar{D} = \frac{\sum_{i=1}^{n} d_i}{n} \quad (5)$$

$$S = \sqrt{\frac{\sum_{i=1}^{n}(\bar{D} - d_i)^2}{n-1}} \quad (6)$$

$$\delta = \sqrt{\frac{\sum_{i=1}^{n} d_i^{\,2}}{n}} \quad (7)$$

4 PROBLEM DESCRIPTION

We consider there are N jobs that have to be scheduled in M machines under the following assumptions:

- There is only one operation for each job, and this operation can be processed on any one of the M machines.
- Each machine can only process one job at a time and a job cannot be processed on different machines at the same time.
- The machines operate with different speeds, and all of them are available at the beginning of the scheduling.
- Jobs are not independent, and there are some precedence relations between them.
- All jobs are not available at the beginning of the scheduling, and each one of them has its own due date.
- Setup times are dependent on job sequence and machine type and anticipatory setups are considered.
- Pre-emption, i.e. job splitting is not allowed for jobs.

The objectives of the problem are minimizing the number of tardy jobs and the makespan.

4.1. Notations and their definitions

M: Total number of machines

N: Total number of jobs for processing

UB: Maximum number of situations on each machine that jobs are placed on them; and are: UB $= N + M - 1$

P_{im} : Processing time of job i on machine m; i $=1,\ 2...$ $N;\ m = 1,\ 2...\ M$

P_i = Processing time of job i

P_j = Processing time of job j

d_i : Due date of job i

r_i : Time at which job i is available for processing (ready time)

S_{ijm} : Setup time to switch from job i to job j on machine m; $i, j = 1,\ 2...\ N;\ m = 1,\ 2...\ M$

S'_{jm} : Setup time of job j on machine m, if it is the first job on the machine

S_{ij} : The same setup time for all machines, for processing job j immediately after job i

L′: A large positive number

$C_{i:}$ Completion time of job i

$U_{i:}$ $U_i = 1$ if job i is tardy; $U_i = 0$ otherwise,

X_{ikm} : if job i is assigned on situation k at machine m then $X_{ikm} = 1$ otherwise, $X_{ikm} = 0$; $k = 1, 2, ..., UB$

Pr $ec(i,\ j)$: precedent constraint between job i and j; if i precedes j then Pr $ec(i,\ j) = 1$; otherwise Pr $ec(i, j) = 0$

X_a: if setup times is anticipatory $X_a = 1$, else $X_a = 0$

r_a : If $C_i + S_{ij} \le r_j$ then $r_a = 1$; else $r_a = 0$.

4.2. Problem formulation

Based on the definition and notation described above, suggested model can be formulated as follows:

Objective Function:

$$Min(C_{\max}) \quad (8)$$

$$Min \sum_{i=1}^{n} Ui \quad (9)$$

Constraints:

$$\sum_{m=1}^{M}\sum_{k=1}^{UB} X_{ikm} = 1, i = 1, 2, ..., N \quad (10)$$

$$\sum_{i=1}^{N} X_{ikm} \le 1, k = 1, 2, ..., UB, m = 1, 2, ..., M \quad (11)$$

$$\sum_{i=1}^{N} X_{ikm} - \sum_{j=1}^{N} X_{j,k-1,m} \le 0, k = 2, ..., UB \quad m = 1, ..., M \quad (12)$$

$$(X_{jkm}.X_{i(k-1)m}.r_a).r_j = (X_{jkm}.X_{i(k-1)m}.r_a)(P_i + X_a.S_{ijm} + \left[(C_j - C_i) - (P_j + S_{ij})\right]) \quad i, j = 1, ..., N, i \ne j, k = 2, ..., UB, \quad m = 1, 2, ..., M, X_a, r_a \in \{0,1\} \quad (13)$$

$$C_i \ge r_i + \sum_{k=1}^{UB} P_{im} * X_{ikm} \quad i = 1, 2, ..., N, m = 1, 2, ..., M \quad (14)$$

$$C_j - C_i \geq \sum_{m=1}^{M}\sum_{k=1}^{UB} P_{jm} X_{jkm}, \Pr ec(i,j) = 1, \quad i \neq j = 1,...,N \tag{15}$$

$$(C_i - d_i) - LU_i \leq 0, i = 1,2,...,N \tag{16}$$

$$X_{ikm}, U_i \in \{0,1\}, C_i \geq 0 \tag{17}$$

Function (8) and (9) minimize the makespan and the total number of tardy jobs, respectively. Constraint (10) ensures that each job is assigned to one of the existing positions on the machines. Constraint (11) guarantees that on each existing positions, at most one job could be assigned. Constraint (12) ensures that until one position on a machine is empty, jobs do not assign to subsequent positions and jobs assigned to empty positions on each of the machines, respectively. Constraint (13) ensures that if setup times are anticipatory, i.e. X a=1; then setup can be started before the corresponding job becomes available on the machine. Constraint (14) guarantees that, interval between ready time and completion time of a job is enough for processing of that job on each machine. Constraint (15) observes precedence relationships. Constraint (16) specifies the tardy jobs. Constraint (17) defines the type of decision variables.

5 PROPOSED MULTI-OBJECTIVE GENETIC ALGORITHM

A GA is a search technique that imitates the natural selection and biological evolutionary process [43]. GAs have been used in a wide variety of applications, particularly in combinatorial optimization problems, and they were proved to be able to provide near optimal solutions in reasonable time.

A GA starts with a population of randomly generated solutions, called chromosomes. Each chromosome of population is evaluated using some measure of fitness. Parents (certain pairs of chromosomes) are selected based on their fitness value. Each of these pairs combines to produce new chromosomes and some of the chromosomes are modified to generate new population by replacing some of the original chromosomes by new chromosomes. The process is repeated until a stopping criterion is satisfied.

In this paper we have developed two multi-objective genetic algorithms; the so called MOGAC and MOGAT that are similar in generality, and differ in evaluation and selection mechanisms.

5.1. Representation scheme

In this research, we use the encoding scheme proposed by Cheng et al. to represent a solution (chromosome) to the problem at hand [44]. In this encoding scheme, integers are used to represent all sequences of jobs, and an asterisk '*' is used to represent the partition of jobs to machines. For example, for a schedule with 8 jobs and 3 machines, the chromosome can be presented as [248*13*756]. The completed schedule is thus: jobs 2, 4 and 8 on machine 1, jobs 1 and 3 on machine 2, and jobs 7, 5 and 6 on machine 3. Generally, for an M-machine N-job problem, a permissible chromosome contains (M-1) partitioning symbols and N job symbols, resulting in a total size of (M+ N-1). [45]

5.2. Initialization

We generate one third of an initial set of solutions to make up the initial population randomly according to reference [45]. Also, in order to give the GA good initial solutions and to increase the chances of generating good new chromosomes, we insert some solutions (chromosomes) generated by two heuristics proposed in this research. To generate the random solutions for one third of initial population, the procedure is as follows.

i. Set $i = 1$

ii. Produce $(M-1)$ asterisks "*" and assign them randomly to genes of the chromosome in which none of them assigned to the first and last genes, and between asterisks must be at least one unfilled genes.

iii. Assign numbers from 1 to N to the remaining unfilled genes of chromosome

iv. Set $i = i + 1$

v. if i> population _ size , STOP, else go to step ii

Our two heuristics are as follows:

1. At first for generating a chromosome, we sort jobs randomly next to each other, and then we swap some chromosomes to consider precedent constraints (in a linear form). Then we start from the first gene and assign that job to a machine with least completion time, and this procedure will be continued till the last gene. We generate one third of our initial population with this method.

2. In the second heuristic, Jobs would be assigned randomly to the machines and then, using the EDD method, they would be sorted on each machine according to their due-dates. We generate the last set of our initial population with this method.

5.3. Evaluation

It is necessary to describe how objectives should be computed before considering evaluation mechanism of chromosomes in the algorithm.

5.3.1. Evaluation of objectives

Makespan (Cmax) is computed as follows:

Suppose job i is immediately after job j on machine m

and job(s) k precedes job i, $C_i = \max\{r_i , C_j + S_{jim} ,CK\} + P_{im}$

$CK = \max\{C_k\}$; $k = 1, 2, ..., k'$
If job i does not have any precedents; $CK = 0$
If job i is the first job on machine m; $C_i = \max\{r_i , S'_{im} ,\text{CK}\} + P_{im}$
$C_{\max} = \max\{C_i\}$
Number of tardy jobs for each chromosome is computed as follows:
$U_i = 0$
For i=1 to N
If Ci > di, then $U_i = U_i + 1$
END

5.3.2. Evaluation of chromosomes in MOGAC (primary and secondary ranking)

We calculate the two objectives (makespan and number of tardy job) considered in this research for all the chromosomes. Next, all the chromosomes are ranked by the following procedure.
All solutions that are non-dominated with respect to each other are assigned as rank 1, and then removed from contention. For the remaining solutions, the next set of non- dominated solutions, those dominated by solutions whose rank is 1 but non-dominated amongst the rest, are assigned as rank 2, and then removed from contention. The procedure is continued until all chromosomes are ranked. We refer to this as primary ranking of chromosomes.
Next, we give the secondary ranking of all the chromosomes with respect to their crowding distances. The principle of a crowding technique is to give more preference to those chromosomes that are away from the peaks of multi-modal functions than to the chromosomes that are in the peaks of multi-modal functions. By doing this, the possibility of many solutions in a population converging to a single non-dominated solution is decreased, and diversity among chromosomes during crossover, and in the subsequent generation is maintained [45], [46]. Here, the method of computing crowding distance is based on [47]. Crowding distance of chromosome k with respect to all other chromosomes in the same rank is defined as c _ dk:

$$C - d_k = \sum_{l \in \{FfK\}, K \neq l} D_{ki} \tag{18}$$

Where D_{kl} is the crowding distance between chromosomes k, l and F_k denotes the rank on which chromosome k lies and $\{F_{fk}\}$ denotes the chromosomes that lie on the same front as that of chromosome k. D_{kl} is defined as follows:

$$D_{kl} = \sqrt{\sum_{r=1}^{R} \left(\frac{Z_r(k) - Z_r(l)}{\max_{k' \in \{F_{fk}\}} Z_r(k') - \min_{k' \in \{F_{fk}\}} Z_r(k')} \right)^2} \tag{19}$$

Where $Z_r(k)$ and $Z_r(l)$ denote the r^{th} objective function for chromosome k, l, and R denotes the number of objectives under consideration. Max $Z_r(k')$ and min $Z_r(k')$ denote the maximum and minimum r^{th} objective value of Chromosomes k' within the set of chromosomes $\{F_{fk}\}$. Note that if, Max $Z_r(k')$ = min $Z_r(k')$ then we set value of:

$$\left(\frac{Z_r(k) - Z_r(l)}{\max_{k' \in \{F_{fk}\}} Z_r(k') - \min_{k' \in \{F_{fk}\}} Z_r(k')} \right)^2 = 0 \tag{20}$$

The chromosomes with the largest value of crowding distance is ranked as 1 in comparison to other chromosomes on the same front, since the chromosome with the largest value of crowding distance lies farthest with respect to other chromosomes on the same front. The secondary ranks are assigned to chromosomes by arranging them in descending order of c _ dk.

5.3.3. Evaluation of chromosomes in MOGAT

According to [45], evaluating chromosomes in this algorithm is based on the TOPSIS method. This is a technique for ranking a limited number of alternatives using a number of decision criteria. It stands for total order preference by similarity to the ideal solution. It is based on the principles of geometry; plotted in Euclidean space, the optimal solution is the one with the shortest distance to the positive ideal solution and the longest distance from the negative ideal solution. The algorithm determines the relative distances and sorts chromosomes in terms of similarity as TOPSIS does, and giving higher priority to the extreme solutions. The order strongly depends on the weights that the decision maker assigns to each objective according to their preferences. We assign equal weights to each objective i.e. $W_1 = 0.5$ and $W_2 = 0.5$ to have no preference between objectives.

5.4. Reproduction

The best chromosomes of the current population are copied directly to the next generation (elitism mechanism). Here we reproduce non-dominated solutions of each generation to the next generation.

5.5. Selection

5.5.1. Selection in MOGAC

We use binary-tournament selection, elitist selection, and purely random selection as our selection mechanisms for MOGAC. In binary-tournament selection scheme, we randomly choose two chromosomes from the parent population. The fitter chromosome is then selected according to its primary rank. Any tie can be broken by the consideration of the secondary rank, and any further tie is broken randomly.

5.5.2. Selection in MOGAT

Binary-tournament selection, elitist selection, and purely random selection are employed as selection mechanisms for MOGAT. Here in tournament selection, the fitter chromosome is the one with the higher rank with respect to the TOPSIS method. 5.6. Local search we conduct two local search schemes on the chromosomes whose primary rank is 1. These approaches were also used by [46].

1. The first local search scheme is called random insertion scheme. To explain this scheme, consider chromosome [3 6 5 * 2 1 4 * 7 8]. We first randomly choose a job position; say job position 7 is chosen. Next, the chosen job is inserted in any other position chosen at random; say position 2 is chosen, and hence the generated chromosome is [3 4 6 5 * 2 1 * 7 8].

2. The second local search scheme is called random swap scheme. For the random swap scheme, two positions are taken at random and swapped.

5.6. Mutation operator

We use random pairwise exchanges as our mutation scheme. This approach was also used by [44] and [45]. First, we select two random genes and then exchange their positions. The randomly selected genes may be either a job or a star. If both selected genes are stars, we reselect the second gene until a job is found. Figure 1 shows an illustration of the mutation scheme.
Selection mechanism for mutation is purely random.

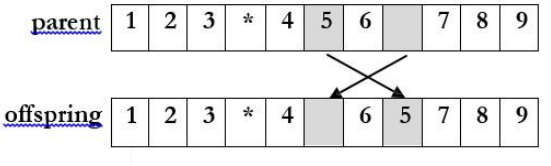

Fig. 1 Illustration of mutation scheme

5.7. Crossover operator

In this research, we employ the crossover scheme proposed by [43]. The crossover scheme is described as follows; first, we use binary-tournament selection scheme as described in section 5.5.1 and 5.5.2, to randomly choose two chromosomes. Next, the crossover scheme takes two parents, and creates a single offspring using the following procedure:

(1) It obtains asterisk positions (i.e. overall partitioning structure) from one parent;

(2) It obtains the remaining jobs from the other parent by making a left-to-right scan.

Figure 2 shows an illustration of the crossover scheme.

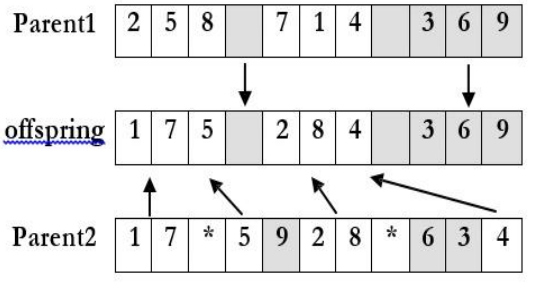

Fig. 2 Illustration of the crossover scheme

5.8. Termination condition

In this article, we use 4000 evaluations as the algorithm termination criteria.

5.9. Computational results

The algorithms were coded in the MATLAB 7.5 environment, and the experiments were executed on a Pentium 4 computer. The experimental parameters are the mutation rate, and the local searches ratio.

5.9.1. Data generation

For solving the presented model, sample problems are produced in medium and large sizes randomly. To produce processing times, setup times, and ready times, we have used uniform distribution [1,150] in a N −dimentional square matrix DU [1, 50] in a N− by −(N, M) matrix and DU[0, 60] in a 1− by− N matrix, respectively. But they are modified to generate only integer values. In this article for producing precedent relations we have developed a procedure that generates feasible precedents. The procedure for producing precedent constraints is as follows:

1. If $N \le 5$ then $L = 2$ otherwise $L = \frac{N}{3}$ for example for $N = 10,\ L = 3$.

2. Generate L random number between 1 and N provided that no two numbers are equal. For example, 3, 6, 7 can be a feasible solution.

3. Define each level, e.g. L_0: 1-2-3; L_1: 4-5-6; L_2: 7; L_3: 8-9-10.

4. Then randomly, it would be specified whether each jobs of level L precedes jobs of level L+1 or not?

Finally, we will have a matrix (Pr ec) in which its arrays included 0 and 1. If $\Pr ec\,(i,\ j) = 1$, it means that job i precedes job j.

The random number generation for the due dates are obtained by $di = (Q + (M\ /10)) \times DU\ [0.3*(P + S),\ 2*(P + S)]$ We have set $Q = 2$ for generating tight due dates.

5.10. Parameter setting

Input parameters of algorithms are as follows:

N: number of jobs

M: number of machines

P_m: mutation probability

Max- Gen: maximum number of generations

Pop- size: population size

It should be mentioned that Pc or crossover probability, and Pr or reproduction probability, in our case, are dependent parameters and are not considered as input parameters. We study the effect of two important parameters (mutation rate and local search ratio) on the performance of our algorithms.

Different levels of these factors are shown in Table 1.

Table 1 Factors and factor levels

Factor	symbol	Levels	Type
Mutation rate	P_m	3	$P_m(1) = 0.2$ $P_m(2) = 0.3$ $P_m(3) = 0.4$
Local Search rate	L	3	$L(1) = 2-1$ $L(2) = 2-2$ $L(3) = 3-2$

Note that L (1): 2-1, means on each chromosome with primary rank of 1, we consider twice local search #1 and once local search #2.

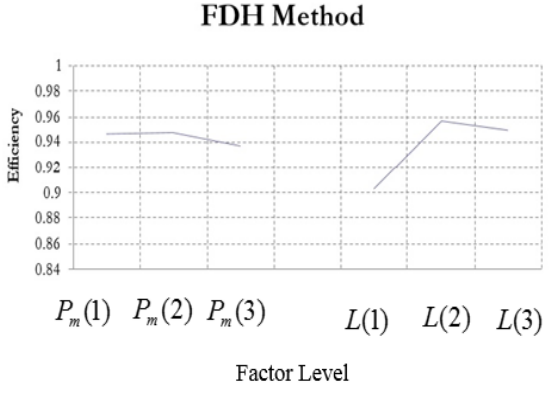

Fig. 3 Efficiency with FDH

For setting parameters of two algorithms we consider 4 test problems, 2 medium size problems ($N = 50,\ M = 5$) and ($N = 30,\ M = 3$), and two large size problems ($N = 80,\ M = 8$) and ($N = 100,\ M = 10$). We run 3 times, all possible status of Table 1 on our test problems. With respect to having multi objective model and possessing a set of non-dominated solutions instead of one discrete solution, here for comparing different parameters, we consider two methods of FDH and distance method to evaluate the quality of parameters. Results are as shown in Figure 3 and Figure 4.

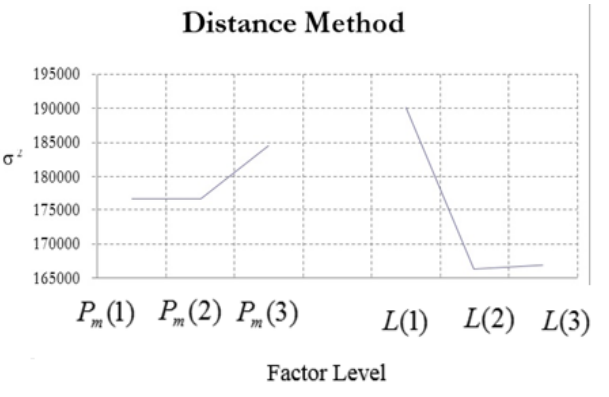

Fig. 4 δ^2 of distance method

As Figure 3 and Figure 4 show, optimal parameters with two methods confirm each other, and $P_m(2)$ and $L(2)$ are set for our algorithms Thus parameters of algorithms are set as follows:

The population size is 100. Corresponding to every chromosome whose primary rank is 1, we generate 2 chromosomes by implementing the random insertion scheme, and 2 chromosomes by implementing the random swap scheme. The probability of mutation is 0.3 for the remaining population (after reproduction and local search). The crossover would finally be performed on number of remaining chromosomes of population.

5.11. Comparison of MOGAT and MOGAC

In this section we compare performance of MOGAT with MOGAC. With respect to comparing two sets of final solutions, we consider three methods to evaluate the quality of solutions; they are FDH method, distance method, and number of non-dominated solutions.

5.11.1. Comparison based on FDH method

We calculated the degree of efficiency using weights W_1 = 0.25, W_2= 0.5, and W_3= 0.75. 10 large and 5 medium test problems were produced. Results are shown in Table 2. Wilcoxon signed-ranked test shows no significant difference between these two algorithms using this method.

Table 2 Efficiency of algorithms

Test Problem	MOGAT			MOGAC		
	W1= 0.75	W1 = 0.5	W1= 0.25	W1= 0.75	W1 = 0.5	W1= 0.25
Large 1	0.988	0.992	0.996	1	1	1
Large 2	0.980	0.987	0.993	1	1	1
Large 3	1	1	1	0.962	0.974	0.987
Large 4	1	1	1	0.951	0.967	0.984
Large 5	1	1	1	0.939	0.955	0.977
Large 6	0.970	0.980	0.990	1	1	1
Large 7	1	1	1	0.953	0.969	0.984
Large 8	1	1	1	0.987	0.991	0.996
Large 9	0.841	0.728	0.614	1	1	1
Large 10	0.970	0.952	0.934	1	1	1
Medium 1	0.966	0.957	0.947	1	1	1
Medium 2	0.954	0.963	0.974	1	1	1
Medium 3	0.903	0.890	0.877	1	1	1
Medium 4	1	1	1	0.978	0.971	0.965
Medium 5	1	1	1	0.985	0.987	0.993

5.11.2. Comparison based on Distance method

Results of Evaluation of δ^2 for both algorithms are shown in Table 3. Wilcoxon signed-ranked test shows no significant difference between these two algorithms using this method.

Table 3 δ^2 Of algorithms

Test Problem	MOGAT	MOGAC
Large 1	150259	150916.5
Large 2	176425	167306
Large 3	143706.5	160177.5
Large 4	213508	196370
Large 5	238265	282105
Large 6	145176.5	131792.5
Large 7	139880	123929
Large 8	182155	213033
Large 9	160817	145571
Large 10	180663	156653.6
Medium 1	227471.667	231718.333
Medium 2	400108	366095.5
Medium 3	191904.5	180641
Medium 4	173462	195834
Medium 5	219717	210389

5.11.3. Comparison based on Number of non-dominated solutions

Results of the number of non-dominated solution are shown in Table 4. Once again Wilcoxon signed-ranked test shows no significant difference between these two algorithms using this method.

Table 4 Number of non-dominated solution

Test Problem	MOGAT	MOGAC
Large 1	2	4
Large 2	1	1
Large 3	3	2
Large 4	4	1
Large 5	5	1
Large 6	2	3
Large 7	4	1
Large 8	2	2
Large 9	1	2
Large 10	2	5
Medium 1	4	3
Medium 2	4	6
Medium 3	4	2
Medium 4	3	4
Medium 5	3	4

6 CONCLUSION

Industrial scheduling has greatly benefited from the use of unrelated parallel machines due to their ability to perform the same function, but with varying capability or capacity. In this paper, two multi-objective genetic algorithms (MOGAT&MOGAC) were proposed to find the unrelated parallel machines scheduling problem that minimizes makespan and number of tardy jobs with considering that jobs have non-identical due dates, and ready times with some precedence relations, Furthermore sequence-dependent setup times and anticipatory setups were included in the model. In this paper, we proposed two heuristics, each of which produced one third of our initial population.

The difference between MOGAT and MOGAC is in the evaluation and selection mechanisms. Sorting the chromosomes in MOGAT is based on TOPSIS method, and in MOGAC it is based on non-dominated sorting (primary sorting) and crowding distance (secondary sorting). For solving the model, sample problems were produced with a new proposed procedure for generating precedent constraints, and a new proposed formulation for producing due dates.

After comparing the results of two algorithms by using FDH method, distance method, and number of non-dominated solutions, we concluded there is no statistically significant difference between these two

algorithms using Wilcoxon signed ranks test. The approaches proposed in this article are fairly general in that they can be extended to other multi-criteria problems. Future research would involve further exploration of objectives and consideration of some realistic assumptions, such as, machine availability constraints.

REFERENCES

[1] Lin, Yang-Kuei, "Data Generation and heuristics for unrelated parallel machine scheduling problems", Doctoral Dissertation, Arizona State University, 2006.

[2] Blidgue, U., Kirac, F., Kurtulan, M., and Pekgun, P., "A Tabu Search Algorithm for Parallel Machine total Tardiness Problem. Computers and Operation Research", Vol. 31, No. 3, 2004, pp. 397-414.

[3] Pinedo, M., "Scheduling Theory, Algorithms and Systems", Prentice Hall, Englewood Cliffs, NJ, 2002.

[4] Sanjay Radhakrishnan, Jose, A., "Ventura Simulated Annealing for Parallel Machine Scheduling with Earliness/Tardiness Penalties and Sequence-Dependent Setup Tmes", International Journal of Production Research, Vol. 38, No. 10, 2000, pp. 2233-2252.

[5] Allahverdi, A., Ng, C. T, Cheng, T. C. E, and Kovalyov Mikhail, Y., "A Survey of Scheduling Pproblems with Setup Times or Costs", European Journal of Operational Research 187, 2008, pp. 985-1032.

[6] Garey, M. R., Johnson, V., "Scheduling Tasks with Non uniform Deadlines on Two Processors", Journal of the ACM 23, 1976, pp. 461-467.

[7] Brucker, P., "Scheduling Algorithm", Berlin: Springer 1998.

[8] Cheng, T. C. E., Sin, C.C.S., "A State-of-the-art Review of Parallel Machine Scheduling Research", European Journal of Operational Research, Vol. 47, No. 3, 1990, pp. 271-292.

[9] Lam, K., Xing, W., "New Trends in Parallel Machine Scheduling", International Journal of Operations and Production Management, Vol. 17, No. 3, 1997, pp. 326-338.

[10] Davis, E., Jaffe, J. M., "Algorithms for Scheduling Tasks on Unrelated Processors", Laboratory for Computer Science, Massachusetts Institute of Technology, 1979.

[11] Lenstra, J. K., Rinnooy, Kan, A. H. G., and Brucker, P., "Complexity of Machine Cheduling Problems", Annals of Discrete Mathematics 1, 1977, pp. 343-362.

[12] Hariri, A. M. A., Potts, C. N., "Heuristics for Scheduling Unrelated Parallel Machines", Journal of Computers and Operations Research, Vol. 18, No. 3, 1991, pp. 323-331.

[13] Glass, C. A., Potts, C. N., and Shade, P., "Unrelated Parallel Machine Scheduling using Local Search", Mathematical and Computer Modeling, Vol. 20, No. 2, 1994, pp. 41-52.

[14] Piersma, N., Van Dijk, W., "A Local Search Heuristic for Unrelated Parallel Machine Scheduling with Efficient Neighborhood Search", Mathematical and Computer Modeling, Vol. 24, No. 9, 1996, pp. 9-11.

[15] Suresh, V., Chaudhuri, D., "Scheduling of Unrelated Parallel Machines when Machine Availability is Specified", Production Planning and Control, Vol. 7, No. 4, 1996, pp. 393-400.

[16] Srivastava, B., "An Effective Heuristics for Minimizing Makespan on Unrelated Parallel Machines", Journal of the Operational Research Society, Vol. 49, 1998, pp. 886-94.

[17] Uzsoy, R., Martin-Vega, L. A., Lee, C. Y., and Leonard, P. A., "Production Algorithms for Semiconductor Test Facility", IEEE Transactions on Semiconductor Manufacturing, Vol. 4, 1991, pp. 270-280.

[18] Hurink, J., Knust, S., "List Scheduling in a Parallel Machine Environment with Precedence Constraints and Setup Times", Operations Research Letters, Vol. 29, 2001, pp. 231-239.

[19] Shmoys, D. B., Tardos, E., "Scheduling Unrelated Machines with Costs", Proceeding of the 4th Annual ACM-SIAM Symposium, Austin, TX, Jan 1993, pp. 448-454.

[20] Suresh, V., Chaudhuri, D., "Bi-criteria Scheduling Problem for Unrelated Parallel-Machines", Computers and Industrial Engineering, Vol. 30, 1996, pp. 77-82.

[21] Prakash, D., "Bi-Criteria Scheduling Problems on Parallel Machines", M.Sc. Thesis, Department of Industrial and System Engineering, Virginia Polytechnic Institute and State University, U.S.A. 1997.

[22] Porkolab, L., Jansen, K., "Improved approximation schemes for scheduling unrelated parallel- machines", ACM Symposium on Theory of Computing, 1999, pp. 408-417.

[23] Funda S., Serifoglu, G. U., "Parallel Machine Scheduling with Earliness Tardiness Penalties", Computers and Operation Researches, Vol. 26, 1999, pp. 773-787.

[24] Gupta, J. N. D., Ho, J. C., and Webster, S., "Bi-Criteria Optimisation of the Makespan and Mean Flowtime on Two Identical Parallel Machines", Journal of the Operational Research Society, Vol. 51, No. 11, 2000, pp. 1330-1339.

[25] Gupta, J. N. D., Ho, J. C., "Minimizing Makespan Subject to Minimum Flowtime on two Identical Parallel Machines", Computers and Operations Research, Vol. 28, 2001, pp. 705-717

[26] Gupta, J. N. D., Ruize-Torres, J. A., "LISTFIT Heuristic for Minimizing Makespan on Identical Parallel Machines", Vol. 12, 2001, pp. 28-36.

[27] Yu, L., Shih, H. M., Pfund, M., Carlyle, W. M., and Fowler, J. W., "Scheduling of Unrelated Parallel Machines: an Application to PWB Manufacturing", IIE Transactions, Vol. 34, No. 11, 2002, pp. 921-931.

[28] Lin, C. H, Liao, C. J., "Makespan Minimization Subject to Flowtime Optimality on Identical Parallel Machines", Computers and Operations Research, Vol. 31, 2003, pp. 1655-1666.

[29] Angel, E., Bampis, E., and Kononov, A., "On the Approximate Tradeoff for bi-criteria Batching and Parallel Machine Scheduling Problems", Theoretical Computer Science, Vol. 306, No. 1-3, 2003, pp. 319-338

[30] Cochran, J. K., Horng, S.-M., and Fowler, J. W., "A Multi-population Genetic Algorithm to Solve Multi-objective Scheduling Problems for Parallel Machines", Computers and Operations Research, Vol. 30, 2003, pp. 1087-1102.

[31] Huo, Y., Leung, J., and Zhao, H., "Bi-Criteria Scheduling Problems: Number of Tardy Jobs and Maximum Weighted Tardiness", European Journal of Operational Research, Vol. 177, 2007, pp. 116-134.

[32] Eren, T., "A Bi-Criteria Parallel Machine Scheduling with a Learning Effect of Setup and Removal Times", Applied Mathematical Modeling, Vol. 33, 2009, pp. 1141-1150.

[33] Jolai, F., Asefi, H., Rabiee, M., and Ramezani, P., "Bi-objective Simulated Annealing Approaches for No-wait Two Stage Flexible Flow Shop Scheduling Problem", Scientia Iranica (Sharif University of Technology), Vol. 20, No. 3, 2013, pp. 861-872.

[34] Lee, C. -H., Liao, C. -J., and Chao, C. -W., "Unrelated Parallel Machine Scheduling with Dedicated Machines and Common Deadline", Computer and industrial engineering, Vol. 74, 2014, pp. 161-168.

[35] Min Joo, C., Soo Kim, B., "Hybrid Genetic Algorithms with Dispatching Rules for Unrelated Parallel Machine Scheduling with Setup Time and Production Availability", Computer and Industrial Engineering, Vol. 86, 2015, pp. 102-109.

[36] Lin, Y. –K, Lin, H. -C., "Bicriteria Scheduling Problem for Unrelated Parallel Machines with Release Dates", Computers & Operations Research, Vol. 64, 2015, pp. 28–39.

[37] Loukil, T., Teghem, J., and Tuiyttens, D., "Solving Multi-Objective Production Scheduling Problems using Metaheuristics", European Journal of Operational Research, Vol. 161, 2005, pp. 42-61.

[38] Tavakkoli-Moghaddam, R., Rahimi-Vahed, A., and Hossein Mirzaei, A., "A Hybrid Multi-Objective Immune Algorithm for a Flow Shop Scheduling Problem with Bi-Objectives: Weighted Mean Completion Time and Weighted Mean Tardiness", International Journal of Information Science, Vol. 177, 2007, pp. 5072-5090.

[39] Deb, K., "Multi-Objective Optimization Using Evolutionary Algorithms", John Wiley& Sons, Inc., New York, NY, 2001.

[40] Charnes, A., Cooper, W. W., and Rhodes, E., "Measuring the Efficiency of Decision Making Units", European Journal of Operational Research, Vol. 2, No. 6, 1978, pp. 429-444.

[41] Allen, R., Athanassopoulos, A., Dyson, R. G., and Thanassoulis, E., "Weights Restrictions and Value Judgements in Data Envelopment Analysis: Evolution, Development and Future Directions", Annals of Operational Research, Vol. 73, 1997, pp. 13-34.

[42] Ruiz-Torres, A. J., Lopez, F. J., "Using the FDH Formulation of DEA to Evaluate a Multi-Criteria Problem in Parallel Machine Scheduling", Computers and Industrial Engineering, Vol. 47, 2004, pp. 107-121. 16.

[43] Holland, J. H., "Adaptation in Natural and Artificial Systems", University of Michigan Press, 1975.

[44] Cheng, R., Gen, M., and Tosawa, T., "Minmax Earliness/ Tardiness Scheduling in Identical Parallel Machine System using Genetic Algorithms", Computers and Industrial Engineering, Vol. 29, 1995, pp. 513-517.

[45] Lin, Y. K., Fowler, J. W., and Pfund, M. E., "Multiple-Objective Heuristics for Scheduling Unrelated Parallel Machines", European Journal of Operational Research, Vol. 227, No. 2, 2013, pp. 239-253.

[46] Lin, Y. –K., "Data Generatiion and Heuristics for Unrelated Parallel Machine Scheduling Problems", Doctoral Dissertation, Arizona State University, 2006.

[47] Pasupathy, T., Rajendran, C., and Suresh, R. K., "A multi-Objective Genetic Algorithm for Scheduling in Flow Shops to Minimize the Makespan and Total Flow Time of Jobs", International Journal of Advanced Manufacturing Technology, Vol. 27, 2006, pp. 804-815.