Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A hybrid algorithm for unrelated parallel machines scheduling

Mohsen Shafiei Nikabadi[a*] and Reihaneh Naderi[b]

[a]*Assistant Professor of Industrial Management Department, Faculty of Economics, Management and Administration Sciences, Semnan University, Semnan, Iran*
[b]*Ph.D. Student of Industrial Management Department, Faculty of Economics, Management and Administration Sciences, Semnan University, Semnan, Iran*

C H R O N I C L E

Article history:
Received November 4 2015
Received in Revised Format December 21 2015
Accepted February 18 2016
Available online February 19 2016

Keywords:
Scheduling, genetic algorithm
Simulated Annealing
Unrelated parallel machines
Analytic network process

A B S T R A C T

In this paper, a new hybrid algorithm based on multi-objective genetic algorithm (MOGA) using simulated annealing (SA) is proposed for scheduling unrelated parallel machines with sequence-dependent setup times, varying due dates, ready times and precedence relations among jobs. Our objective is to minimize makespan (Maximum completion time of all machines), number of tardy jobs, total tardiness and total earliness at the same time which can be more advantageous in real environment than considering each of objectives separately. For obtaining an optimal solution, hybrid algorithm based on MOGA and SA has been proposed in order to gain both good global and local search abilities. Simulation results and four well-known multi-objective performance metrics, indicate that the proposed hybrid algorithm outperforms the genetic algorithm (GA) and SA in terms of each objective and significantly in minimizing the total cost of the weighted function.

1. Introduction

From a theoretical viewpoint, parallel machines scheduling is significant as many algorithms can be reduced to solve single machine problems and from practical viewpoint, it is important because in a real manufacturing environment, most workshops have more than one machine (Lin, 2006). According to Pinedo (2002), when machines are not identical and cannot be completely correlated by simple rate adjustments, they are named unrelated parallel machines. One of the most complicated issues is scheduling of unrelated parallel machines among all types of parallel machines scheduling although there are a few studies on unrelated parallel ones or sequence-dependent setups times (Kayvanfar et al., 2014).

In this work, we have considered unrelated parallel machines with varying ready times and due dates. Due to the difference between technologies used, speed of works in different machines and the processing times for the jobs that must be scheduled on these machines are different, in the other hand all jobs are not available at the beginning of scheduling and also every job may have its own due date (Tavakkoli-

* Corresponding author.
E-mail: shafiei@profs.semnan.ac.ir (M. Shafiei Nikabadi)

doi: 10.5267/j.ijiec.2016.2.004

Moghaddam et al., 2008). Some researchers neglect setup times which will affect the solution's accuracy, Setup times consist of all activities that are done on material in order to prepare machines and situations.

There are sequence-dependent and sequence-independent setup times (Radhakrishnan & Ventura, 2000). 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 (Allahverdi et al., 2008). This paper has considered sequence-dependent setup times, anticipatory setup and also precedence constraints.

Sometimes, delays in delivery are unavoidable which result in cancelling customers' orders. While tardiness is relevant to customer concerns and may influence on customer satisfaction, in another side, earliness represent manufacturer concerns (Kayvanfar et al., 2014). Makespan is the completion time of the last job leaving the system and as is stated by Lin and Ying (2013). Although, many researchers have devoted considerable research efforts to minimize total completion time of jobs on parallel machines and makespan and the number of tardy jobs, but minimizing the total tardiness, total earliness are the objectives which have been considered less by previous researches. Decreasing tardiness has gained more attention than decreasing earliness in past literature. Real-world problems engage with multi-objective optimization in the objective function. 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, rather than one optimal solution (Tavakkoli-Moghaddam et al., 2007). Multi-objective optimization problems can be discerned in some approaches, among them Utility approach can be said one of the most used approaches. 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 which is used in this paper. There are some other approaches like Hierarchical approach, Goal programming, Interactive approach and Simultaneous or Pareto approach (Loukil et al., 2005).

The contribution of this paper is that four multi objective functions have been considered at the same time; makespan, number of tardy jobs and total tardiness and total earliness. Scheduling jobs on two identical machines- even with one objective- is stated as NP-hard (Garey & Johnson, 1976; Brucker, 1998; Cochran et al., 2003), so by increasing the number of machines more than two and adding more objectives, it becomes more difficult to solve such problems (Brucker,1998). Therefore more complicated algorithm needs to be created in order to obtain better optimal solutions. Recently, some hybrid algorithms based on genetic algorithm (GA) and simulated annealing (SA) have been presented in literature, including routing in wireless sensor networks (Shokouhifar & Jalali, 2014), task scheduling in multiprocessor platforms (Yoo & Gen, 2007), traveling salesman problem (Elhaddad, 2012), However, the hybridization methodology in is different (Shokouhifar & Jalali, 2014). To evaluate the proposed algorithms, random test problems are produced.

The rest of the paper is organized as follows: Section 2 gives related literature. Some approaches for multi-objective optimization problems are described in section 3. Problem description will be described in section 4. Section 5 explains our evolutionary proposed algorithm and discusses computational studies and simulation results. Finally, Section 6 includes conclusions and future researches.

2. Literature review

Cheng and Sin (1990) widely reviewed Parallel machine-scheduling problems with conventional performance measures based on due date, flow time and completion time, On the other hand, Lam and Xing (1997) reviewed 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. Genetic algorithm was used in Sridhar and Rajendran (1996) to minimize a makespan, total flow time and machine idle time and obtained better results in comparison with the work of Ho and

Chang in (1991). Demirkol et al. (1998) minimized maximum lateness with precedence constraints and sequence-dependent jobs by using neighbourhood search algorithm.

Gupta et al. (2000) proposed an algorithm for minimizing the weighted sum of makespan and mean flow time to solve bi-criteria problem. Yu et al. (2002) proposed a two-stage Lagrangian relaxation heuristic (LRH) method. Kim et al. (2002) presented unrelated parallel machines scheduling with sequence-dependent setup times to minimize total tardiness by using SA. Cochran et al. (2003) 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. Suresh and Mohanasundaram (2004) used simulated annealing algorithm for the flow shop scheduling problem to minimize makespan and total flow time. Loukil et al. (2005) applied simulated annealing algorithm and considered pairs of objectives of: average weighted completion time, average weighted tardiness, makespan, maximum tardiness, maximum earliness and the number of tardy jobs. Potential efficient solutions were presented based on computational time. Tavakkoli-Moghaddam et al. (2006) presented a mathematical model for minimizing the total earliness and tardiness penalties and machine costs in unrelated parallel machines, they used genetic algorithms for solving this problem. Huo et al. (2007) employed heuristic algorithm to solve multi-objective parallel machines scheduling to minimize the number of tardy jobs and maximum weighted lateness.

Logendran et al. (2007) considered unrelated parallel machine scheduling problem for minimizing the weighted tardiness of all jobs. The study was based on dynamic job releases and dynamic machine availability. They proposed six different algorithms based on tabu search. In another work, Ruiz and Andres (2007) scheduled an unrelated parallel machine problem that had both machine and job sequence dependent setup times, in their scheduling, the setup times were depend on machine and job sequence and also on a number of resources assigned in order to be more real. They used MIP model and some heuristics.

Eren (2009) 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. Dubois-Lacoste et al. (2011) presented a hybrid local search algorithm for minimizing different combinations of two objectives simultaneously: makespan and sum of the completion times of the jobs; makespan and total tardiness; makespan and the weighted total tardiness; total flow time and total tardiness; total flow time and weighted total tardiness and gained better results compared with multi-objective simulated annealing (Varadharajan & Rajendran, 2005) and genetic local search (Arroyo & Armentano, 2005). Ying et al. (2012) developed a restricted simulated annealing (RSA) algorithm and Lin and Ying (2014) presented a hybrid artificial bee colony (HABC) algorithm for minimizing the makespan. Three bi-objective optimization methods was developed by Jolai et al. (2013) based on simulated annealing.

Bozorgirad and Logendran (2012) studied a sequence-dependent group unrelated-parallel scheduling problem, where their objective was to minimize a linear combination of total weighted completion time and total weighted tardiness and developed meta-heuristic algorithms based on tabu search. Arnaout (2010) introduced an Ant Colony Optimization algorithm to minimize the makespan on the unrelated parallel machine scheduling problem with sequence-dependent setup times and machine-dependent problem. They compared their results with Tabu Search of Helal et al. (2006), and Partitioning Heuristic of Al-Salem (2004), and Meta-RaPS of Rabadi et al. (2006). Arnaout, Musa, and Rabadi in 2014, proposed an enhanced Ant Colony Optimization algorithm and obtained better performance than previous version.

Yenisey and Yagmahan (2014) provided a literature review of flow shop scheduling problem. According to their study, in the area of multiple objectives parallel machines scheduling, the most common objectives are minimizing the makespan, maximum tardiness, total weighted tardiness and total weighted flow time. Some of researches would be discussed in the following. Lee et al. (2014) proposed three heuristics for minimizing the total completion time of unrelated parallel machines.

A hybrid genetic algorithms with three dispatching rules for large-sized problems was used by Joo and Kim (2015). Hassanpour et al. (2015) applied simulated annealing (SA), genetic algorithm (GA) and a bottleneck based heuristic (BB) algorithms for solving the problem of minimizing makespan of the jobs in no-wait reentrant flow shop production environment. A heuristic and a Tabu search algorithm was used by Lin and Lin (2015) for finding non-dominated solutions to bi-criteria unrelated parallel machine scheduling problems with release dates. There are some studies in 2016 which have taken tardiness and earliness into account in the scheduling problems, although there are different objective functions and different suppose in their problems, some of them are as following papers. Molina-Sánchez and González-Neira (2016), studied the scheduling problem to minimize the total weighted tardiness in a Permutation Flow Shop (PFS) environment and used Greedy Randomized Adaptive Search Procedure. Zarei et al. (2016), addressed both Job selection and scheduling together. The combination of considered cost in their study include jobs' processing costs and weighted earliness and tardiness penalties and has used simulated annealing and some other heuristics to maximize the total net profit as an objective, they resulted that through SA, solutions with more reasonable computational time could be achieved. Azizi et al. (2016) used GA and SA algorithm for scheduling m-machine with no-wait flowshop and according to their results, SA gained better optimal solutions in most instances.

Further to the literature reviews on scheduling problems, the most studied objectives are makespan, completion time and there are fewer works which have considered both earliness and tardiness at the same time. In the other side, more works in this area can be found that have studied both earliness and tardiness for identical parallel machines, but still there are few works about unrelated parallel machines scheduling. Biskup and Cheng (1999), studied parallel machines scheduling with their objective to minimize earliness, tardiness and completion time penalties. Sivrikaya and Ulusoy (1999) applied genetic algorithm for minimizing earliness and tardiness penalties for parallel machines with distinct due dates and sequence-dependent setup times. In 2001, Bank and Werner studied on Unrelated parallel machines scheduling for minimizing the weighted sum of earliness and tardiness penalties. Vallada and Ruiz (2012) considered unrelated parallel machines scheduling with machine and job sequence dependent setup times for the objective of minimizing the total weighted earliness and tardiness. de CM Nogueira et al. (2014) studied on minimizing the total earliness and tardiness penalties of unrelated parallel machine scheduling problem, also they considered Machine and job-sequence dependent setup times and idle times in their scheduling and tested it by Greedy Randomized Adaptive Search Procedure (GRASP) metaheuristic to determine near-optimal solutions. As it was stated, there are fewer works on scheduling unrelated parallel machines with sequence-dependent setup times, varying due dates, ready times and precedence relations between jobs. Also in existed papers, the observed objectives are mostly earliness and tardiness, while in this paper, four objectives are investigated at the same time, including makespan, number of tardy jobs, total tardiness and earliness, in order to be more applicable in real environment.

3. Multi- objective optimization

In the literature of multi-objective optimization problems, different approaches can be found:

In utility approach, a weighted linear combination of the objectives, is used to aggregate the considered objectives in a single one. for example The scalar objective function is the weighted sum of individual objectives, $F(X)= W_1 \times f_1(X) + W_2 \times f_2(X) + W_3 \times f_3(X) + W_4 \times f_4(X)$, where W_1, …, W_4 are non-negative weights given by the user. Or LP-Norm method in which: $F(X) = W_1 \times (f_1-f_1^*)/f_1^* + W_2 \times (f_2-f_2^*)/f_2^* + W_3 \times (f_3-f_3^*)/f_3^* + W_4 \times (f_4-f_4^*)/f_4^*$, In this way, the problem should be solved several times to find f_1^* to f_4^* and then using the optimal values combine all four objective functions using W_1 to W_4 into one single objective function and solve the resulted problem.

Goal programming is another approach: all of the objectives are taken into account as constraints which express some satisfying levels 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. (Loukil et al., 2005). There are different min-max methods also Interactive approach can be used and at each step of the procedure, the decision maker expresses his preferences about some proposed solutions so that the method will progressively converge to a satisfying compromise among the considered objectives. There is another approach which is called Simultaneous or Pareto approach and the goal is to generate, or to approximate in case of a heuristic method, the complete set of efficient solutions (Loukil et al., 2005). Some other methods as Hierarchical approach, and etc can be find in this regards.

In this paper pareto approach is applied to obtain the set of efficient and optimal solutions also for finding the minimized total cost of all objective functions simultanously, LP-Norm method is used to aggregate the considered objectives in a single one and w_1 to w_4, have been defined to determine the relative importance of the objectives f_1 to f_4, respectively. More specifically, the biggest weight the more importance. In this paper it is assumed that all of four objectives have the same importance for the special scheduling problem, so these wights are considered just as an example, while these weights can be defined differently based on condition and special situation of each scheduling problem.

4. Problem Description

It is assumed that there are *N* jobs that have to be scheduled in *M* machines. The precedence relations among the jobs can be considered from the job relations graph. The problem is optimized when simultaneously minimizes the makespan, number of tardy jobs, total tardiness, total earliness, under the following conditions:

• Each machine can only process one job at a time and a job cannot be processed on different machines at the same time
There is only one operation for each job, and this operation can be processed on any one of the M machines.
• 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 and deadline of each job is known.
• 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.

4.1. Notations and Their Definitions

M:	Total number of machines; (m=1, 2, 3… M)
N:	Total number of jobs; (i, j=1, 2, 3… N)
H:	(T, E) job sequence graph
x_i^m :	a binary parameter defining that job i is assigned on machine m or not
c_i^m :	completion time of job i on machine m
c_i :	($\sum_{m=1}^{M} c_i^m x_i^m, \forall i$), Total completion time of job i
d_i :	Due date of job i
t_i^E :	earliest ready time of job i

t_i^S :	real start time of job i
t_i^F :	finish time of job i
SU:	start-up time (Time at which job i is available for processing (ready time)
pre(i):	set of all predecessor jobs of job i
Suc(i):	set of all successor jobs of job i
e_{ij}:	a binary parameter defining that i is pre(i) or not
Ti:	Ti =1 if job i is tardy; Ti =0 otherwise,
Ei:	Ei =1 if job i is early; Ei =0 otherwise,
p_i^m	Processing time of job i on machine m

4.2. Problem Formulation

Based on the definition and notation described above, the proposed model can be formulated as follows, utility function is used to aggregate the considered objectives in a single weighted one:

Objective Function (ζ):

$$\min \zeta = w_1(f_1 - f_1^*)/f_1^* + w_2(f_2 - f_2^*)/f_2^* + w_3(f_3 - f_3^*)/f_3^* + w_4(f_4 - f_4^*)/f_4^* \quad (1)$$

$$f_1 = \max_{i,m}\left(\left(t_i^S + c_i^m\right)x_i^m\right) \quad (2)$$

$$f_2 = \sum_{i=1}^{N} T_i \quad (3)$$

$$f_3 = \sum_{i=1}^{N} max\left\{0, \sum_{m=1}^{M}\left(t_i^S + c_i^m - d_i\right)x_i^m\right\} \quad (4)$$

$$f_4 = \sum_{i=1}^{N} \max\left\{0, \sum_{m=1}^{M} d_i - t_i^s - c_i^m)x_i^m\right\} \quad (5)$$

subject to

$$T_i = \begin{cases} 1 \to if \sum_{m=1}^{M}(t_i^S + C_i^m - d_i)X_i^m > 0 \\ 0 \to Otherwise \end{cases} \quad (6)$$

$$E_i = \begin{cases} 1 \to if \sum_{m=1}^{M}(d_i - t_i^S - c_i^m)x_i^m > 0 \\ 0 \to Otherwise \end{cases} \quad (7)$$

$$x_i^m = \begin{cases} 1 & \textit{if machine m is selected for job i} \\ 0 & \textit{otherwise} \end{cases} \quad (8)$$

$$\sum_{m=1}^{M} x_i^m = 1 \quad (9)$$

$$t_i^S \geq t_i^E \quad (10)$$

$$t_i^S = t_i^E + SU \quad (11)$$

$$t_i^F = t_i^S + \sum_{m=1}^{M} c_i^m x_i^m \quad (12)$$

$$t_i^E \geq t_j^E + SU + \sum_{m=1}^{M} c_j^m x_j^m, i \in pre(i) \quad (13)$$

$$t_i^E = \begin{cases} 0, if \rightarrow pre(i) \in \{\phi\} \\ \max_j \left\{ t_j^E + SU + \sum_{m=1}^{M} c_j^m x_j^m \right\}, i \in pre(i) \end{cases} \tag{14}$$

$$e_{ij} = \begin{cases} 1 & if\ job_j \in pre(job_i) \\ 0 & otherwise \end{cases} \tag{15}$$

$$C_i \geq SU + \sum_{m=1}^{M} p_i^m * x_i^m \tag{16}$$

$$c_i - d_i = T_i - E_i \rightarrow \forall i \tag{17}$$

$$X_i^m, T_i, E_i \in \{0,1\}, C_i \geq 0 \text{ , } w_i = 0.25 \tag{18}$$

Eq. (1) is associated with the proposed multi-objective function to be minimized. Eqs. (2-5) mean to minimize: makespan (f_1), total number of tardy jobs (f_2), total tardiness (f_3) and total earliness (f_4). The constraint conditions are shown from Eq. (6) to Eq. (18): Eq. (6) and (7) define tardiness and earliness. Eq. (8) observes the assignment of each machine for one job. Eq. (9) means that every job is processed on one machine at a time. Eq. (10) means that the job can be started after its earliest possible start time. Eq. (11) defines the real start time of job i and Eq. (12) defines the finish time of job i. Eq. (13) ensures that the job can be started after all its predecessor jobs are done. Eq. (14) defines the earliest possible start time of the job i is calculated by the maximum finish time of all predecessor jobs of the job i. Eq. (15) observes precedence relationships. Eq. (16) guarantees that, interval between ready time and completion time of a job is enough for processing of that job on each machine. Eq. (17) defines the earliness and tardiness of job i. as each job could only be tardy or early, if it could not be delivered timely, so it is clear that Ti and Ei cannot take value simultaneously. Constraint (18) defines the type of decision variables and non-negativity.

5. The proposed methodology

Genetic Algorithm (GA) is a search technique and a population-based meta-heuristic algorithm that belongs to the larger class of evolutionary algorithms. GA can generate near-optimal solutions for the optimization problems using techniques inspired by natural evolution such as selection, crossover, and mutation (Holland, 1975). GA has been used in a wide variety of applications, particularly in combinatorial optimization problems, and was proved to be able to provide near optimal solutions in reasonable time. Simulated Annealing (SA) is a single-solution meta-heuristic algorithm of locating a good approximation to the global optimum of a given function in the search space. It was inspired from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. (Kirkpatrick et al., 1983)

In this paper, a hybrid global and local search strategy based on GA and SA is used for the efficient job scheduling. Our motivation is to simultaneously gain with both good global and local search ability into the proposed algorithm. In the proposed method, at first, GA is utilized to perform a global searching among the whole search space. Then, SA is applied to search locally in the vicinity of the best solution found via GA, in order to improve the final solution of GA. Typically, SA is very sensitive to the initial solution, because it is a single-solution local search algorithm which starts from a random initial solution. In order to overcome this drawback, we consider the best solution of GA as the initial solution for SA. On the other hand, SA starts from a near-optimal solution rather than a random one. Also, a hybrid variable local-search mechanism is utilized to enhance the exploration mechanism and convergence speed of SA. Overall flowchart of proposed scheduling algorithm can be seen in Fig. 1.

5.1. Problem Representation

In job scheduling problem, the objective of scheduling is not only to optimize an execution order of jobs, but also to determine the specific machine for the execution of each job. The first is an ordering problem, and the last is an assignment problem. In order to solve the scheduling problem using the proposed hybrid algorithm, a mechanism must be employed to formulate the problem into feasible solutions.

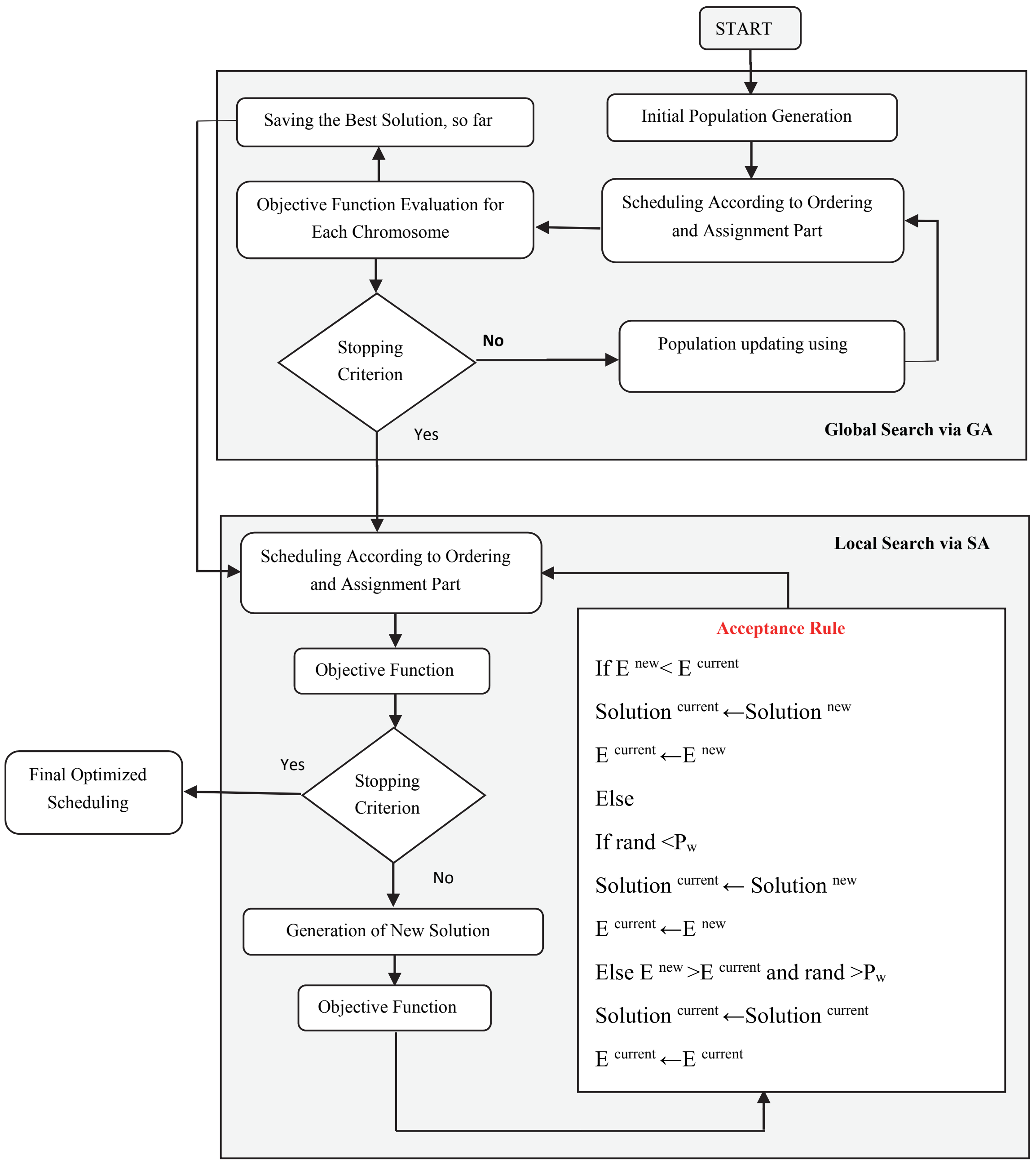

START
Saving the Best Solution, so far
Initial Population Generation
Objective Function Evaluation for Each Chromosome
Scheduling According to Ordering and Assignment Part
Stopping Criterion
No
Population updating using
Yes
Global Search via GA
Local Search via SA
Scheduling According to Ordering and Assignment Part
Acceptance Rule
If E new< E current
Solution current ←Solution new
E current ←E new
Else
If rand <Pw
Solution current ← Solution new
E current ←E new
Else E new >E current and rand >Pw
Solution current ←Solution current
E current ←E current
Objective Function
Final Optimized Scheduling
Yes
Stopping Criterion
No
Generation of New Solution
Objective Function

Fig. 1. Flowchart of the proposed scheduling algorithm

In this paper, a hybrid structure is used to represent a feasible solution, which is divided into two parts. The first part shows the hybrid overall execution order of jobs (ordering part), and the second part determines the machine number to which the job is assigned (assignment part). The length of each part is equal to the total number of all jobs. Therefore, the number of optimization variables is 2×N, where N is the number of all jobs. The ordering part should satisfy the precedence relationship with respect to the given job graph. A feasible solution in both GA and SA phases for a dataset with 10 jobs and 3 machines can be seen in Fig. 2.

	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Ordering Part	8	10	3	9	1	2	4	5	7	6
Assignment Part	1	2	3	1	2	3	2	3	2	3

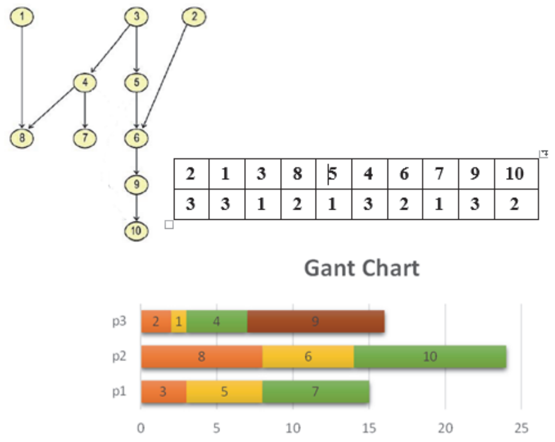
Fig. 2. Representation of a feasible solution for a dataset with 10 jobs and 3 machines

5.2. Global Search via GA

The initial population is randomly generated, at the first step in GA. Then, two steps are consequently done, until the maximum number of iterations of GA has been reached: fitness evaluation and population updating. After fitness evaluation of the current population, some of the best chromosomes are selected as the parents for updating the population. Then, offspring are constructed from the parents via crossover operator. After that all offspring have been generated, the mutation operator is applied to change randomly in the value of some genes within the generated offspring, aim at avoid trapping in local minima points (Shokouhifar & Hassanzadeh, 2014).

5.2.1. Generation of the Initial Population

At first, the initial population of GA is generated, by the sequence-based random generation strategy reported in Yoo and Gen (2007) and Shokouhifar and Jalali (2014). In this way, the ordering part is randomly generated by respecting the precedence relationship of the jobs. On the other hand, any job i could not be presented before the jobs within pre(i) set. In order to achieve this issue, at first all jobs that have not any predecessor jobs are ordered randomly and placed at the beginning of the ordering part. Then, all the predecessor jobs of them were selected and ordered randomly after the pre-ordered jobs. This process continues until all jobs are ordered by respecting the precedence relationship of the jobs. Also, for the assignment part, a machine is selected randomly for each job to be processed on that machine. Procedure of encoding and generation of initial population can be seen in Fig. 3.



2	1	3	8	5	4	6	7	9	10
3	3	1	2	1	3	2	1	3	2

Fig. 3. An example for initial population generation with 8 jobs

5.2.2. Fitness Evaluation

At the every iteration, the fitness of all chromosomes are evaluated according to the proposed objective function according to Eq. (1). Then, all chromosomes are sorted from the best to the worst, and some of the best chromosomes are selected based on elitism-selection strategy as the parents for updating the population.

5.2.3. Population Updating

In order to generate an offspring, two parents are randomly chosen among parents, and the crossover operator is performed on them. In this paper, the two-point permutation-based MOX crossover operator (Majumdara & Bhunia, 2011) is applied for the ordering part, and the uniform crossover operator is used for the assignment part. In the MOX, some parts of the first parent are copied in the offspring, and the rest between these parts are taken in the same order as in the second parent. In this study, the MOX operation carried out with three crossover points which are chosen at random positions. For the first offspring, the genes are copied from the first parent till the initial crossover point, also from the second point till the third point. Then the genes of the second parent are scanned, and the numbers of the second parent which are not within the first offspring, will be appear in the first offspring with the same order that they are came in the second parent. Illustration of MOX operator can be seen in Fig. 4.

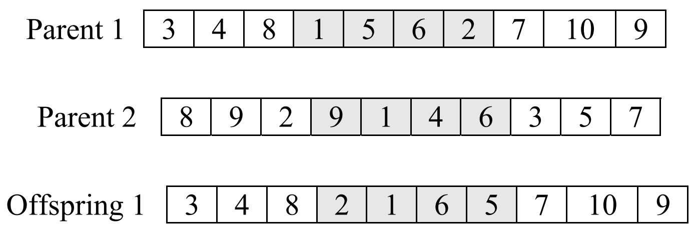


Fig. 4. MOX crossover operator applied for the ordering part

In uniform crossover operator (see Fig. 5.), each gene in the offspring is filled from the same gene of one of the two parents, with the same probability.

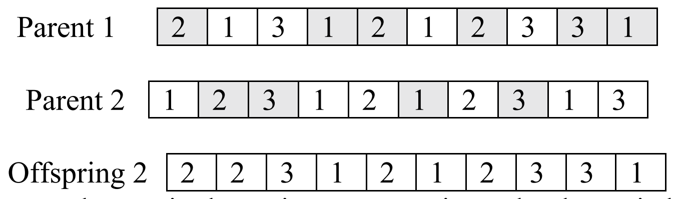


Fig. 5. Uniform crossover: each gene in the assignment part is randomly copied from the same gene of either parent 1 or parent 2

When all offspring have been generated, the mutation operator can be applied in order to change randomly in the value of some genes within the generated offspring, in order to avoid trapping in local minima points. In this paper, the exchange operator is utilized for the ordering part, and the swap operator is applied for the assignment part (see Fig. 6.)

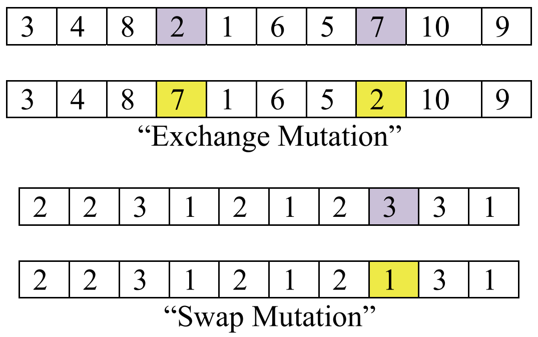


Fig. 6. Mutation Operators to change randomly in the value of some genes within the generated offspring for the ordering and assignment parts

5.3. Local Search via SA

In general, SA starts with a random initial solution. However, in the proposed hybrid algorithm, the final global best solution found via GA is used as the initial solution for SA. At the every iteration, the two steps generation of new solution and acceptance rule checking are consequently done, until the stopping criterion satisfied.

5.3.1. Generation of New Solution

At the every iteration, a new solution is generated in the vicinity of the current solution. As mentioned above, different local search operators are used for the generation of the new solution. Each operator can avoid trapping a type of local minima points. In this approach, Swap, Exchange, Relocation, Or-opt, and Reverse operators (Shokouhifar and Jalali, 2014) and (Caric and Gold, 2008) are applied. The Relocation, Or-opt, and Reverse operators are performed in the first part (ordering part) of the solution. The swap is applied in the second part (assignment part). Also, the Exchange can be used in both parts, called Exchange-1 and Exchange-2, respectively. The six local search operators can be seen in Fig. 7.

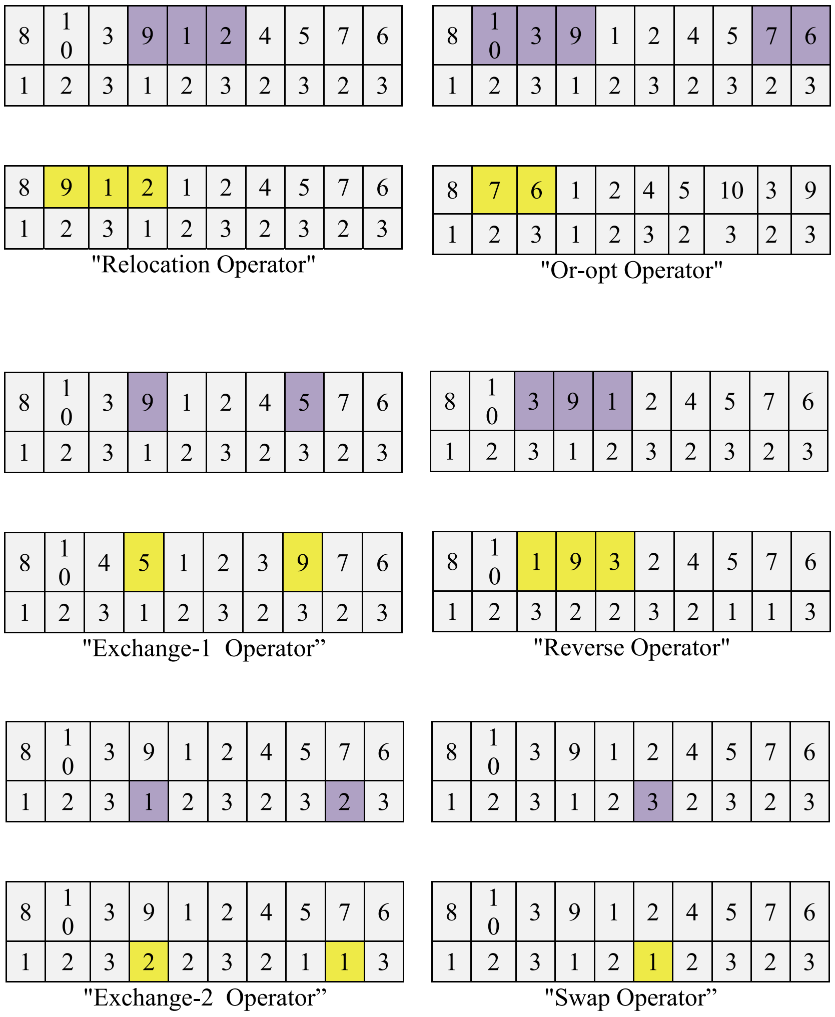


Fig. 7. Local search operators applied for the generation of new solution in the SA phase

5.3.2. Acceptance Rule Checking

At the every iteration of SA, a new solution (named Solution new) is generated in the neighborhood area of the current solution (named Solution current). If Enew ≤ Ecurrent, then the current solution is replaced with the new one. On the other hand, if Enew > Ecurrent, the new solution may be accepted with the probability of Pw, which is calculated as follows:

$$P_w = exp\left(-\frac{\left(E^{new} - E^{current}\right)}{T}\right) \tag{19}$$

where, E^{current} and E^{new} are the objective function values (according to Eq. (1)) for current and new solution, respectively. According to Eq. (18), the temperature *T* is considered to be decreased linearly from *T* initial (initial temperature) to *T* final (final temperature), during execute algorithm. T SA and iter SA are the current iteration, and the defined number of iterations in SA, respectively. If *T*=0, Solution new never could be accepted, when Enew > Ecurrent. On the other hand, the larger T, the more probability for accepting worse solutions.

$$T = T_{initial} + \frac{t_{SA}}{iter_{SA}} \times \left(T_{final} - T_{initial}\right) \tag{20}$$

5.3.3. Computational Results

The algorithms were coded in the MATLAB 7.5 environment, and the experiments were executed on a PC Quad Core 2.4 GHZ and 8 GB memory running on windows 8.1.

5.3.4. 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 (Safaei et al., 2016). But they are modified to generate only integer values. As it was described in section 5.2.1, for producing random precedent relations, we used sequence-based random generation strategy as below:

```
/1    E_mat=zeros (num_job, num_job);
/2        For i=1:num_job-1
/3          For j=i+1:num_job
/4             If rand<.1
/5                E_mat (i, j) =1;
/6             End
/7        End
/8    End
```

We will have a N×N matrix pre(i) in which its arrays included 0 and 1. If pre(i,j) = 1, it means that job *i* precedes job *j*. The random number generation for the due dates are obtained by:

D_mat=round(Pref_mat.×(10+2×and(1,num_job))+(50×rand(1,num_job))) (21)

While Pref_mat is considered as below:

```
/9    Pref_mat=zeros (1, num_job);
/10   a=find (sum (E_mat) ==0);
/11   Pref_mat (a) =1;
```

For example in “Pref_mat [1 3]” 1 means that it has no job as precedent and 3 means that there are 2 steps (jobs) before this special job, that jobs which have more steps or precedents before them, have higher due dates, so based on this procedure, we can have more realistic data. Also random completion times are produced through below M*N Matrix named C_mat, which is placed in dataset:

$$\text{C\_mat}=\text{round}(3+12\times\text{rand}(\text{max\_machine},\text{num\_job})) \tag{22}$$

5.3.5. Parameter Setting

Determination of the parameters is an important issue on the efficiency of the proposed algorithm, it means that the value that our parameters are set on, can change the performance of the meta heuristics algorithms (Arjmand & Najafi, 2015). In order to make a better comparison between GA, SA and proposed hybrid algorithm, different values were evaluated for each parameter, and the best values were selected for simulations, this work is also done in some papers for example in Shokouhifar, and Jalali 2015a and 2015b. In our experiments, the population size of chromosomes was set to 20, and the number of iterations in GA was set to 25. As SA is not population-based and is a single-solution-based algorithm, the number of iterations in SA is set to 500, much larger than the GA. Parameter settings for GA and SA can be summarized in Table 1.

For setting parameters of two algorithms we considered 4 test problems in the dataset of program, 2 medium size problems ($N=50,\ M=5$) and ($N=30,\ M=3$), and two large size problems ($N=80,\ M=8$) and ($N=100,\ M=10$), these parameters can be seen in Table 2.

Table 1
Values of algorithm’s parameters

Algorithm	Parameter	Value
GA Phase	Maximum iteration	50
	Population size	20
	Mutation Probability (P_m)	0.1
	NFE: Number of Fitness Evaluation	1000
	Cross over operators for ordering part	2-point MOX operators
	Cross over operators for assignment part	uniform crossover operator
	Mutation operator for ordering part	exchange operator
	Mutation operator for assignment part	swap operator
SA Phase	Maximum iteration	1000
	T_initial	0.01
	T_final	0
	NFE: Number of Fitness Evaluation	1000
Proposed Algorithm	Iteration=25, population=20	GA Phase
	Iteration=500	SA Phase
	NFE: Number of Fitness Evaluation	1000

Table 2
Dataset details of test problems

Data No.	No. of Job	No. of Machines
1	30	3
2	50	5
3	80	8
4	100	10

5.3.6. Evaluation methods

Evaluating the quality of the obtained solutions is one important aspect, and various metrics are defined in order to calculate the performance of obtained results of used algorithms such as Mean Ideal Distance (MID), Diversification Metric (MD), Spacing Metric (SM), Number of found solutions (NOS) and CPU Time, etc., and most of these metrics have been used in many previous works in the area of multi-objective obtimization. Multi-objective optimization with various and sometime conflicting objective functions, provides a set of optimal solutions that called Pareto optimal, rather than one optimal solution. The Multi-bjective set includes solutions with no better solution while considering all objective functions. For example, assume a typical multi-objective minimization problem with p decision variables and q objectives (q > 1):

$$\min y = f(x) = (f_1(x), f_2(x), \ldots, f_q(x)) \quad (23)$$
$$X \in R^P, \text{and} Y \in R^q$$

Solution a, is said to dominate solution b if and only if:

$$(1) f_i(a) \leq f_i(b); \forall i \in \{1,2,\ldots,q\} \quad (24)$$
$$(2) f_i(a) < f_i(b); \exists i \in \{1,2,\ldots,q\}$$

Solutions that dominate other solutions but do not dominate themselves are called non- dominated solutions. Vector $\overline{a}$, is a globally Pareto-optimal solution if vector $\overline{b}$, does not exist such that $\overline{b}$, dominates $\overline{a}$. (Tavakkoli-Moghaddam et al., 2007)

Based on these definitions, Pareto optimal front can be called as a set of solutions that can not dominate each others and this front has two feautures of good convergence and good diversity within the solutions of the pareto front (Deb, 2001). This issue is described also in Sarrafha et al. (2014), which states that, in Pareto based algorithms, there are two main goals which include good convergence and diversity. From the current metrics in this area, MID and CPU time, measure the convergence rate of the algorithms and the others are used the diversity of the algorithms. As it is stated by Zitzler and Thiele (1998), diversity is used to evaluate the spread of the front and MID evaluates the convergence rate of Pareto fronts to a certain point (0,0). And according to Zitzler (1999), Spacing assess the standard deviation of the distances among solutions of the pareto front and the NOS metric (Number of found solutions), compare the number of the pareto solutions in pareto optimal front. In this paper, we have used four criteria which are among the most used metrics in the area of multi-objective optimization specially in scheduling problems, to measure the solutions' quality for the proposed algorithms: (Geramianfar, Pakzad, Golhashem, and Tavakkoli-Moghaddam, 2013), (Fakhrzad et al., 2012) and (Arjmand & Najafi, 2015).

1. Mean Ideal Distance (MID): the closeness between Pareto solution and the ideal point (0,0) and it is obvious that less value of MID has more interest.

$$MID = \frac{\sum_{i=1}^{n} \sqrt{\left(\frac{f_{1i}-f_1^{best}}{f_{1,total}^{max}-f_{1,total}^{min}}\right)^2 + \left(\frac{f_{2i}-f_2^{best}}{f_{2,total}^{max}-f_{2,total}^{min}}\right)^2}}{n} \quad (25)$$

2. Spacing Metric (SM) which measure the uniformity of the point spread (the extent of spread) among the obtained solution, and the less value of SM has more credit and shows the uniform distributions of solutions within the pareto curves.

$$SM = \frac{\sum_{i=1}^{n-1}\left|\bar{d} - d_i\right|}{(n-1)\bar{d}} \tag{26}$$

where d represents the deviation between two pareto solutions and is calculated by Euclidian norm and $\bar{d}$ is the average of all di.

3. the spread of non-dominance solution (SNS) which shows the diversity in obtained solution: SNS, the higher value of SNS means the better solution quality that we have gained.

$$SNS = \sqrt{\frac{\sum_{i=1}^{n}(MID - ci)^{\wedge}2}{n-1}} \qquad ci = \sqrt{f_{1i}^2 + f_{2i}^2} \tag{27}$$

4. last metric is CPU Time which shows the duration of the meta-heuristic algorithmsand and it is clear that less value of CPU Time has more credit.

5.3.7. Simulation Results

The comparison of the obtained results based on the proposed algorithm with the two other scheduling algorithms has been performed. We ran 10 times, all possible status of Table 1. on our test problems mentioned in Table 2. Then based on gained results, the above mentioned criteria are calculated for each data and for each algorithm, Table 3 shows the results of our computations for data.1 to data.4. Although these comparison can be extended to more extended test problems in future studies.

According to the last row of Table 3, the average of each metric on 4 test problems is calculated. Based on these values in terms of the average, for mean ideal distance (MID), and spread of non-dominance solution (SNS), Proposed algorithm is better according to its values. For CPU Time, in terms of the average, GA has better performance with its less value. And in terms of the average for Spacing Metric (SM), Simulated Annealing (SA), works better than the two other algorithms. The average values with better performance are bold in the last row of Table 3.

Table 3
Computational results of the performance of GA, SA and proposed algorithm for medium-to-large size problems

	GA				SA				Proposed hybrid algorithm			
No. of data	Time	MID	SNS	SM	Time	MID	SNS	SM	Time	MID	SNS	SM
Data.1	10.943713	0.407144	1.80309962	0.234225	292.674982	0.403461	2.12497715	0.12936	147.059343	0.208515	3.10257542	0.208072
Data.2	12.899583	0.298144	5.32304428	0.082935	321.669554	0.292407	5.42817675	0.149457	150.946240	0.28795	5.61750986	0.063009
Data.3	14.678700	0.372773	8.42035439	0.06313	326.198518	0.359326	8.27625649	0.062997	159.16957	0.354266	8.45501337	0.04946
Data.4	16.375723	0.517439	11.1602261	0.25403	303.234797	0.504885	11.9990713	0.10731	163.536041	0.462679	12.0330123	0.22162
Average	**13.7244297**	0.398875	6.67668109	0.15858	310.944463	0.3900197	6.95712042	**0.110781**	155.177798	**0.3283525**	**7.30202773**	0.1355402

The results of Table 3 has been specified very clearly in Fig. 8, based on each metric. According to this figure, the proposed algorithm has the least mean ideal distance (MID) within the Pareto front as well as the most spread of non-dominance solution (SNS). However in the metric of CPU Time, Genetic Algorithm (GA), works completely better than the proposed algorithm and specially better than SM algorithm. The computational times are from running 1000 NFE (Number of Fitness Evaluation) which is based on 25 iterations in GA phase and 500 iterations in SA phase within the proposed algorithm. In addition, in SA based on 1000 iterations and in GA based on 50 iterations with 20 population, it means that the NFE is considered the same (1000) for the better comparison of results in each algorithm. It is so important to attend that a larger number of iterations would lead to larger CPU times and the large number of iterations are considered due to paying attention to the worst scenario.

About Spacing Metric (SM), none of algorithm has the best result (with less value) for all test problems, but the proposed algorithm works better in data 2 and data 3, and in smaller or larger than these two data, SA performs better with its less value. So it should be noted that the performance of the algorithms can be changed according to the number of jobs and machines.

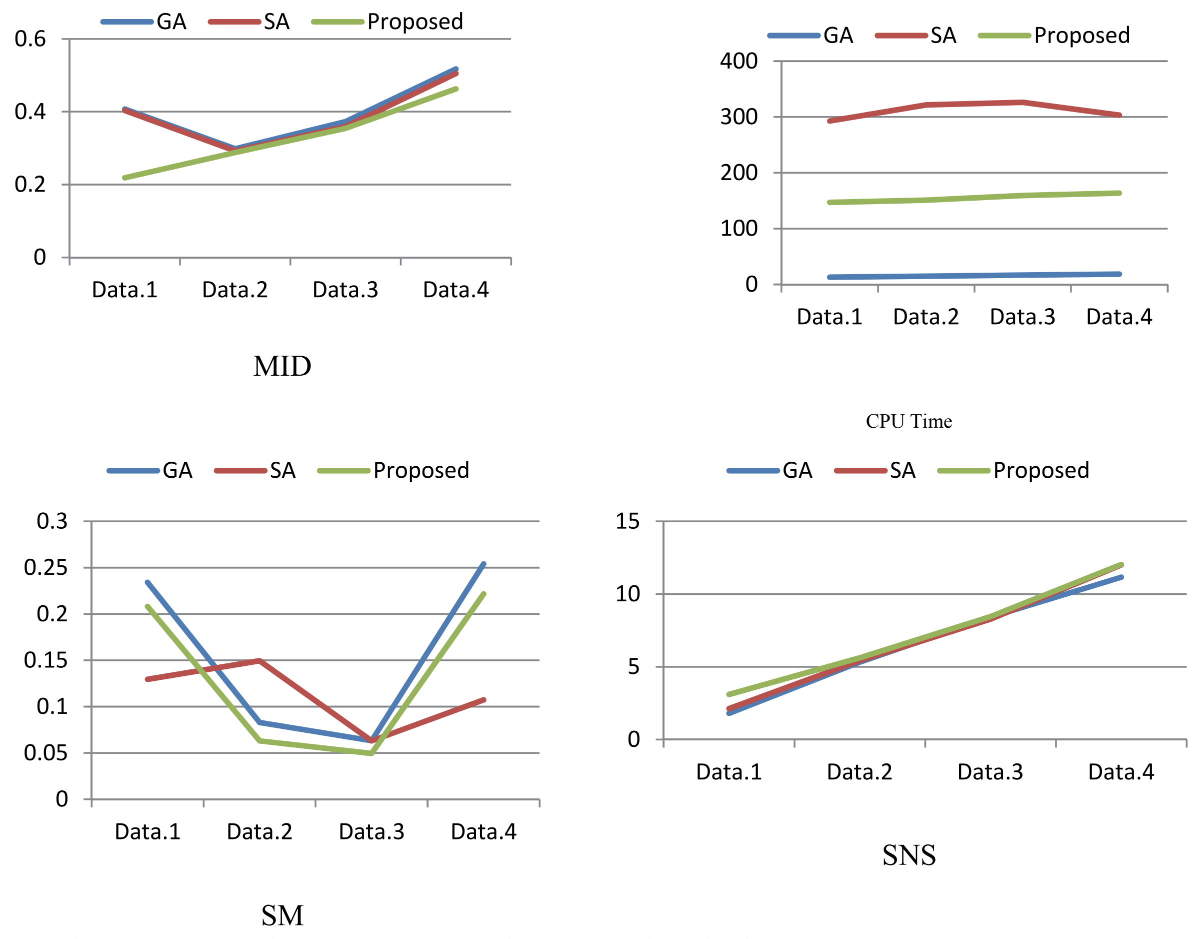


Fig. 8. The summary of the performance of the proposed method in terms of MID, CPU Time, SM and SNS metrics based on different data

In Fig. 9; the average weighted cost of the proposed hybrid algorithm with respect to the obtained solutions in large size problems is demonstrated. The blue color shows GA results and the red color shows SA results and the results are shown based on NFE.

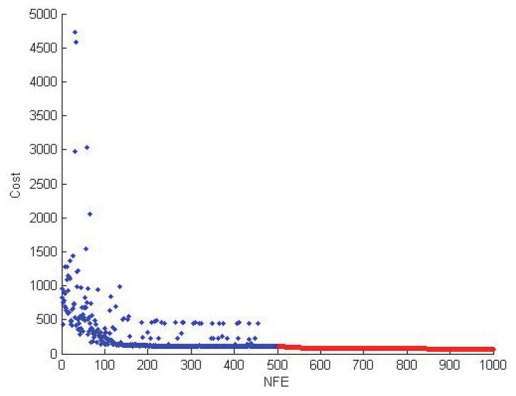


Fig. 9. The average weighted cost of hybrid algorithm in large problems

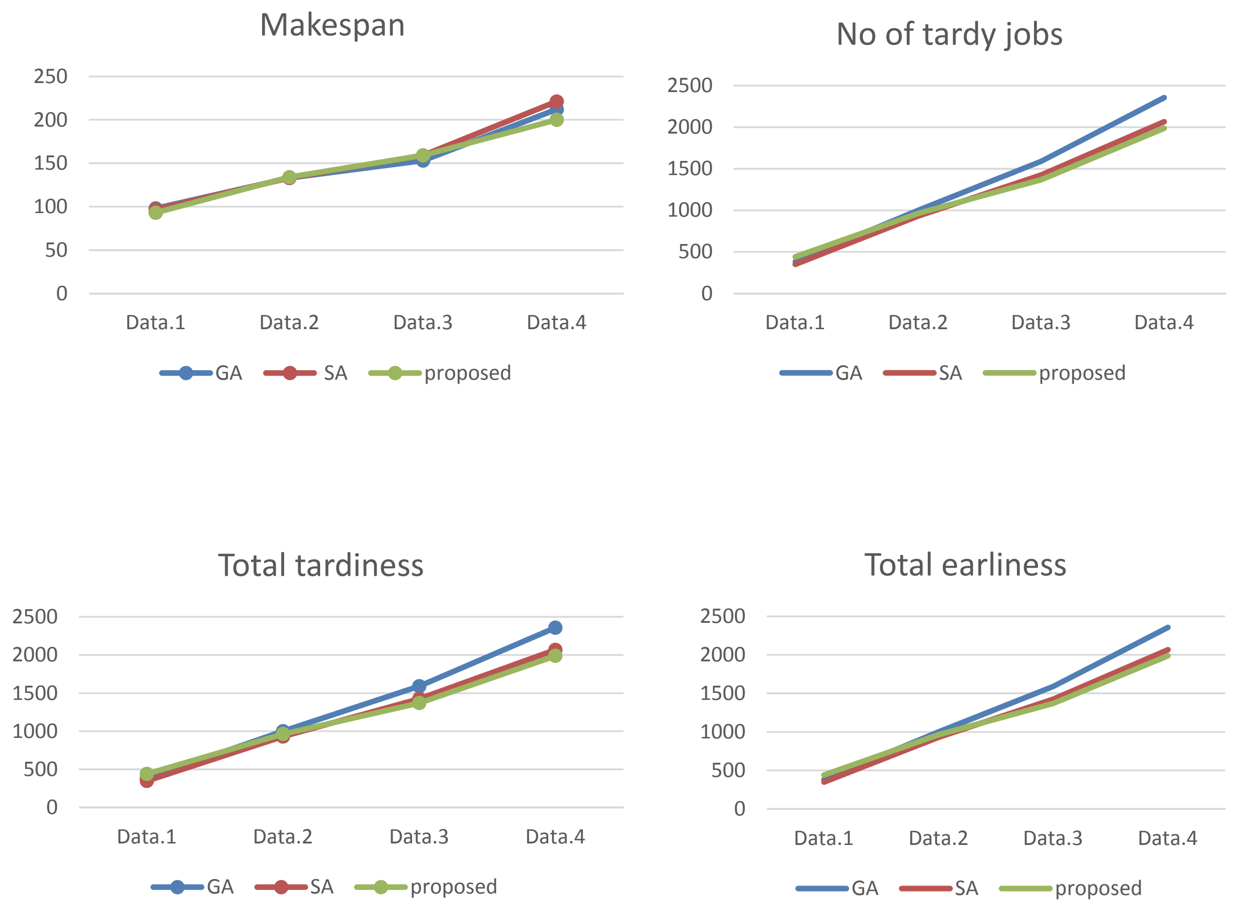


Fig. 10. Obtained Makespan, No of tardy jobs, Total earliness and Total tardiness for each algorithm and each data

Fig. 10 shows the comparison between obtained average objective in each algorithm and for each data. The results show the superiority of proposed hybrid algorithm respectively for medium to large size problems. Since none of 3 algorithms has been better in all the metrics, we cannot be sure which algorithm has the best performance, therefore we used TOPSIS approach, that can rank the given alternatives of the Pareto solutions gained by GA, SA and the proposed algorithm. The positive ideal solution has the smallest makespan, the smallest no. of tardy jobs, the smallest total tardiness and the smallest total earliness in the Pareto solutions. The basic idea of TOPSIS is to find the best compromise solution which is the closest to the positive ideal solution (d_i^+) and the farthest from the negative ideal solution. (d_i^-):

$CL_i = \dfrac{d_i^-}{d_i^- + d_i^+}$, where 0<CLi<1, and the solution with more relative closeness to 1 is better.

The process of TOPSIS for determining the best compromise solution is presented in the works of Arjmand and Najafi (2015). We used this method for analysing our results based on each data. The obtained results are represented in Table 4, in the last row, TOPSIS has been applied with consideration of all test problems. According to the TOPSIS results in Table 4, the proposed algorithm has better value in all situations (each data) and in all the examples as well, and when increasing the number of jobs and machines its value has increased and has a better trend in its TOPSIS values. As a result, we can conclude that the proposed hybrid algorithm has better performance than GA and SA seperately.

Table 4
TOPSIS Results

No. of data	GA	SA	Proposed algorithm
Data.1	0.105	0.155	0.847
Data.2	0.004	0.304	0.981
Data.3	0.017	0.681	0.983
Data.4	0.023	0.551	0.998
All of the test problems	0.005	0.248	1

6. Conclusion

There are vast numbers of papers considered solving scheduling problems for unrelated parallel machines and different kinds of heuristics are developed to get near optimal solutions accordingly. As it is stated in the literature (Garey & Johnson, 1976; Brucker, 1998; Cochran et al., 2003), solving scheduling problems even with one objective is NP-hard. And it is obvious that adding more objectives, makes such problems more difficult to solve, as in this paper four objectives were considered simultaneously, and when the size and complexity of the problems increases, there would be more difficulty for solving problem, therefore more complicated algorithm needs to be created in order to obtain better optimal solutions. One of the most used algorithms in this area is genetic algorithm, Sivrikaya et al. (1999) stated that GA with or without crossover operator is an efficient algorithm for parallel machines scheduling, although neighborhood exchange type of search (only by mutation) can be just effective for small and easy problems, so they proposed GA-MCUOX (GA with crossover operator) which showed better performance for large size and more complicated problems. Cochran et al. (2003) proposed the two-stage multiple population algorithm (MPGA) combining the vector evaluation genetic algorithm and the multiple objective genetic algorithm. On the other hand some researchers have used simulated annealing (SA) for optimizing scheduling problems like Kim et al. (2002) that used SA approach for an unrelated parallel machine scheduling and resulted that it takes longer as the size of a problem increases. In order to rectify the deficiency of using these mentioned algorithms and for gaining the benefits of both GA and SA, we proposed a new hybrid algorithm that showed better performance than other two algorithms.

In this paper, a new hybrid algorithm based on multi-objective genetic algorithm and by using local search algorithms like simulated annealing (SA), was proposed for scheduling *N* jobs on *M* unrelated parallel machines with sequence-dependent setup times and jobs had varying due dates and ready times and there are some precedence relations among them. In global search via GA, the two-point permutation-based MOX crossover operator (Majumdara & Bhunia, 2011) is applied for the ordering part, and the uniform crossover operator has been used for the assignment part. Also for mutation phase, the exchange operator is utilized for the ordering part, and the swap operator has been applied for the assignment part. In local search via SA, we used six different local search operators. In this approach, Swap, Exchange, Relocation, Or-opt, and Reverse operators Shokouhifar and Jalali (2014) and Caric and Gold (2008) are applied. The Relocation, Or-opt, and Reverse operators are performed in the first part (ordering part) of the solution. The swap is applied in the second part (assignment part). Also, the Exchange can be used in both parts, called Exchange-1 and Exchange-2, respectively and each operator can avoid trapping a type of local minima points.

The objective of the proposed algorithm was to minimize makespan (Maximum completion time of all machines), number of tardy jobs, total tardiness and total earliness simultaneously. The contribution of this paper in objective function was to consider all of mentioned objectives simultaneously by new mathematical modeling. Random test problems were produced from medium to large size to examine the performance of proposed hybrid algorithm. Number of tardy jobs and total tardiness and total earliness also the total cost gained by the proposed algorithm, appeared quite more desirable than those of the others. Makespan gained close results in each of algorithms. To analyze and make comparison between the performance of the algorithms (GA, SA and proposed hybrid algorithm), four performance criteria for multi-objective optimization, were used. According to the results, the proposed algorithm has the least mean ideal distance (MID) within the Pareto front as well as the more spread of non-dominance solution (SNS). However in the metric of CPU Time, Genetic Algorithm (GA), worked completely better than proposed algorithm and specially better than SM algorithm and for Spacing Metric (SM), proposed algorithm, worked better in data 2 and data 3 and for smaller or larger than these two data, SA performs better with its less value. For determining the most efficient algorithm, TOPSIS method was applied and resulted in the most efficiency of the proposed algorithm than other two algorithms.

Future research would involve further exploration of objectives and constraints, also research efforts should be made to compare the results of hybrid algorithm with different parameters in order to gain the best condition.

Acknowledgement

The authors would like to thank the annonymous referees for constrictive comments on earlier version of this paper.

References

Allahverdi, A., Ng, C. T., Cheng, T. E., & Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, *187*(3), 985-1032.

Arjmand, M., & Najafi, A. A. (2015). Solving a multi-mode bi-objective resource investment problem using meta-heuristic algorithms. *Advanced Computational Techniques in Electromagnetics, 1*, 41-58.

Arnaout, J. P., Musa, R., & Rabadi, G. (2014). A two-stage Ant Colony optimization algorithm to minimize the makespan on unrelated parallel machines—part II: enhancements and experimentations. *Journal of Intelligent Manufacturing, 25*(1), 43-53.

Arnaout, J. P., Rabadi, G., & Musa, R. (2010). A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times. *Journal of Intelligent Manufacturing, 21*(6), 693-701.

Arroyo, J. E. C., & Armentano, V. A. (2005). Genetic local search for multi-objective flowshop scheduling problems. *European Journal of Operational Research*, *167*(3), 717-738.

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

Bank, J., & Werner, F. (2001). Heuristic algorithms for unrelated parallel machine scheduling with a common due date, release dates, and linear earliness and tardiness penalties. *Mathematical and computer modelling*, *33*(4), 363-383.

Biskup, D., & Cheng, T. E. (1999). Multiple-machine scheduling with earliness, tardiness and completion time penalties. *Computers & operations research*, *26*(1), 45-57.

Bozorgirad, M. A., & Logendran, R. (2012). Sequence-dependent group scheduling problem on unrelated-parallel machines. *Expert Systems with Applications, 39*(10), 9021-9030.

Brucker, P. (1998). *Scheduling algorithm*. Berlin: Springer

Caric, T., & Gold, H. (Eds.). (2008). Vehicle routing problem. Sciyo. com.

Cheng, T. C. E., & Sin, C. C. S. (1990). A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research*,*47*(3), 271-292.

Cochran, J. K., Horng, S. M., & Fowler, J. W. (2003). A multi-population genetic algorithm to solve multi-objective scheduling problems for parallel machines. *Computers & Operations Research*, *30*(7), 1087-1102.

Deb, K. (2001). Multi-objective optimization using evolutionary algorithms (Vol. 16). John Wiley & Sons.

Demirkol, E., Mehta, S., & Uzsoy, R. (1998). Benchmarks for shop scheduling problems. *European Journal of Operational Research*, *109*(1), 137-141.

De CM Nogueira, J. P., Arroyo, J. E. C., Villadiego, H. M. M., & Gonçalves, L. B. (2014). Hybrid GRASP heuristics to solve an unrelated parallel machine scheduling problem with earliness and tardiness penalties. *Electronic Notes in Theoretical Computer Science, 302*, 53-72.

Dubois-Lacoste, J., López-Ibáñez, M., & Stützle, T. (2011). A hybrid TP+ PLS algorithm for bi-objective flow-shop scheduling problems. *Computers & Operations Research*, *38*(8), 1219-1236.

Elhaddad, Y. R. (2012). Combined Simulated Annealing and Genetic Algorithm to Solve Optimization Problems. *World Academy of Science, Engineering and Technology*, *68*, 1508-1510.

Eren, T. (2009). A bicriteria parallel machine scheduling with a learning effect of setup and removal times. *Applied Mathematical Modelling*, *33*(2), 1141-1150.

Fakhrzad, M., Sadeghieh, A., & Emami, L. (2012). A new multi-objective job shop scheduling with setup times using a hybrid genetic algorithm. *International Journal of Engineering-Transactions B: Applications, 26*(2), 207.

Garey, M. R., & Johnson, D. S. (1976). Scheduling tasks with nonuniform deadlines on two processors. *Journal of the ACM (JACM)*, *23*(3), 461-467.

Geramianfar, R., Pakzad, M., Golhashem, H., & Tavakkoli-Moghaddam, R. (2013). A multi-objective hub covering location problem under congestion using simulated annealing algorithm. *Uncertain Supply Chain Management, 1*(3), 153-164.

Gupta, J. N. D., Ho, J. C., & Webster, S. (2000). Bicriteria optimisation of the makespan and mean flowtime on two identical parallel machines. *Journal of the Operational Research Society*, *51*(11), 1330-1339.

Hassanpour, S. T., Naseri, M. A., & Nahavandi, N. (2015). Solving re-entrant no-wait flow shop scheduling problem. *International Journal of Engineering-Transactions C: Aspects*, *28*(6), 903.

Holland, J. H. (1975). Adaptation in natural and artificial system: an introduction with application to biology, control and artificial intelligence. *Ann Arbor, University of Michigan Press*.

Ho, J. C., & Chang, Y. L. (1991). A new heuristic for the n-job, M-machine flow-shop problem. *European Journal of Operational Research*, *52*(2), 194-202.

Huo, Y., Leung, J. Y. T., & Zhao, H. (2007). Bi-criteria scheduling problems: Number of tardy jobs and maximum weighted tardiness. *European Journal of Operational Research*, *177*(1), 116-134.

Jolai, F., Asefi, H., Rabiee, M., & Ramezani, P. (2013). Bi-objective simulated annealing approaches for no-wait two-stage flexible flow shop scheduling problem. *Scientia Iranica*, *20*(3), 861-872.

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

Kayvanfar, V., Komaki, G. M., Aalaei, A., & Zandieh, M. (2014). Minimizing total tardiness and earliness on unrelated parallel machines with controllable processing times. *Computers & Operations Research*, *41*, 31-43.

Kim, D. W., Kim, K. H., Jang, W., & Chen, F. F. (2002). Unrelated parallel machine scheduling with setup times using simulated annealing. *Robotics and Computer-Integrated Manufacturing*, *18*(3), 223-231.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, *220*(4598), 671-680.

Lam, K., & Xing, W. (1997). New trends in parallel machine scheduling.*International Journal of Operations & Production Management*, *17*(3), 326-338.

Lee, C. H., Liao, C. J., & Chao, C. W. (2014). Unrelated parallel machine scheduling with dedicated machines and common deadline. *Computers & Industrial Engineering*, *74*, 161-168.

Lin, S. W., & Ying, K. C. (2013). Minimizing makespan and total flowtime in permutation flowshops by a bi-objective multi-start simulated-annealing algorithm. *Computers & Operations Research*, *40*(6), 1625-1647.

Lin, S. W., & Ying, K. C. (2014). ABC-based manufacturing scheduling for unrelated parallel machines with machine-dependent and job sequence-dependent setup times. *Computers & Operations*

Research, *51*, 172-181.

Lin, Y. K. (2006). *Data generation and heuristics for unrelated parallel machine scheduling problems* (Vol. 67, No. 11).

Lin, Y. K., & Lin, H. C. (2015). Bicriteria scheduling problem for unrelated parallel machines with release dates. *Computers & Operations Research*, *64*, 28-39.

Logendran, R., McDonell, B., & Smucker, B. (2007). Scheduling unrelated parallel machines with sequence-dependent setups. *Computers & Operations Research, 34*(11), 3420-3438.

Loukil, T., Teghem, J., & Tuyttens, D. (2005). Solving multi-objective production scheduling problems using metaheuristics. *European Journal of Operational Research*, *161*(1), 42-61.

Majumdar, J., & Bhunia, A. K. (2011). Genetic algorithm for asymmetric traveling salesman problem with imprecise travel times. *Journal of Computational and Applied Mathematics*, *235*(9), 3063-3078.

Pinedo, M. (2002). *Scheduling: theory, algorithms, and systems*.

Molina-Sánchez, L. P., & González-Neira, E. M. (2016). GRASP to minimize total weighted tardiness in a permutation flow shop environment.International *Journal of Industrial Engineering Computations, 7*(1), 161.

Radhakrishnan, S., & Ventura, J. A. (2000). Simulated annealing for parallel machine scheduling with earliness-tardiness penalties and sequence-dependent set-up times. *International Journal of Production Research*,*38*(10), 2233-2252.

Ruiz, R., & Andrés, C. (2007). Scheduling unrelated parallel machines with resource-assignable sequence dependent setup times. Technical Report DEIOAC-2007-01, Universidad Politécnica de Valencia.

Sarrafha, K., Kazemi, A., & Alinezhad, A. (2014). A multi-objective evolutionary approach for integrated production-distribution planning problem in a supply chain network. *Journal of Optimization in Industrial Engineering,7*(14), 89-102.

Safaei, S., Naderi, R., Sohrabi, A., Hatami, A. (2016). Scheduling of Unrelated Parallel Machines using two Multi Objective Genetic Algorithm with Sequence-Dependent Setup Times and Precedent Constraints. *International Journal of Advanced Design and Manufacturing Technology (ADMT).*

Shokouhifar, M., & Jalali, A. (2014, May). Real-time task scheduling in heterogeneous multiprocessor systems using artificial bee colony. In*Electrical Engineering (ICEE), 2014 22nd Iranian Conference on* (pp. 1007-1012). IEEE.

Shokouhifar, M., & Hassanzadeh, A. (2014). An energy efficient routing protocol in wireless sensor networks using genetic algorithm. *Advances in Environmental Biology, 8*(21), 86-93.

Shokouhifar, M., & Jalali, A. (2015a). A new evolutionary based application specific routing protocol for clustered wireless sensor networks. *AEU-International Journal of Electronics and Communications, 69*(1), 432-441.

Shokouhifar, M., & Jalali, A. (2015b). An evolutionary-based methodology for symbolic simplification of analog circuits using genetic algorithm and simulated annealing. *Expert Systems with Applications, 42*(3), 1189-1201.

Sivrikaya-Şerifoğlu, F., & Ulusoy, G. (1999). Parallel machine scheduling with earliness and tardiness penalties. *Computers & Operations Research*,*26*(8), 773-787.

Sivrikaya, F., & Ulusoy, G. (1999). Parallel machine scheduling with earliness and tardiness penalties, *Computers & Operations Research, 26* 773 – 787.

Sridhar, J., & Rajendran, C. (1996). Scheduling in flowshop and cellular manufacturing systems with multiple objectives—a genetic algorithmic approach. *Production Planning & Control*, *7*(4), 374-382.

Suresh, R. K., & Mohanasundaram, K. M. (2004, December). Pareto archived simulated annealing for permutation flow shop scheduling with multiple objectives. In *Cybernetics and Intelligent Systems, 2004 IEEE Conference on* (Vol. 2, pp. 712-717). IEEE.

Tavakkoli-Moghaddam, R., Jolai Ghazvini, F., Khodadadeghan, Y., & Haghnevis, M. (2006). A Mathematical Model of a Multi-Criteria Parallel Machine Scheduling Problem: a Genetic Algorithm (RESEARCH NOTE).*International Journal of Engineering-Transactions A: Basics*, *19*(1), 79.

Tavakkoli-Moghaddam, R., Rahimi-Vahed, A., & Mirzaei, A. H. (2007). A hybrid multi-objective immune algorithm for a flow shop scheduling problem with bi-objectives: weighted mean completion time and weighted mean tardiness. *Information Sciences*, *177*(22), 5072-5090.

Tavakkoli-Moghaddam, R., Taheri, F., & Bazzazi, M. (2008). Multi-Objective Unrelated Parallel Machines Scheduling with Sequence-Dependent Setup Times and Precedence Constraints. *International Journal of Engineering, Transactions A: Basics*, *21*(3), 269-278.

Vallada, E., & Ruiz, R. (2012). Scheduling unrelated parallel machines with sequence dependent setup times and weighted earliness–tardiness minimization. In Just-in-Time Systems (pp. 67-90). Springer New York.

Varadharajan, T. K., & Rajendran, C. (2005). A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs. *European Journal of Operational Research*, *167*(3), 772-795.

Yenisey, M. M., & Yagmahan, B. (2014). Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. *Omega, 45, 119-135.*

Ying, K. C., Lee, Z. J., & Lin, S. W. (2012). Makespan minimization for scheduling unrelated parallel machines with setup times. *Journal of Intelligent Manufacturing*, *23*(5), 1795-1803.

Yoo, M., & Gen, M. (2007). Scheduling algorithm for real-time tasks using multiobjective hybrid genetic algorithm in heterogeneous multiprocessors system. *Computers & Operations Research*, *34*(10), 3084-3098.

Yu, L., Shih, H. M., Pfund, M., Carlyle, W. M., & Fowler, J. W. (2002). Scheduling of unrelated parallel machines: an application to PWB manufacturing. *IIE transactions*, *34*(11), 921-931.

Zarei, M. H., Davvari, M., Kolahan, F., & Wong, K. Y. (2016). Simultaneous selection and scheduling with sequence-dependent setup times, lateness penalties, and machine availability constraint: Heuristic approaches. *International Journal of Industrial Engineering Computations, 7*(1), 147.

Zitzler, E., & Thiele, L. (1998, September). *Multiobjective optimization using evolutionary algorithms—a comparative case study*. In Parallel problem solving from nature—PPSN V (pp. 292-301). Springer Berlin Heidelberg.