Journal of Optimization in Industrial Engineering
Vol.14, Issue 1, Winter & Spring 2021, 169-184
DOI:10.22094/JOIE.2021.1890852.1718

Research Article

Minimizing the Operational Costs in a Flexible Flow Shop Scheduling Problem with Unrelated Parallel Machines

Ali Hasani [a], Seyed Mohammad Hassan Hosseini[a,*], Foroogh Behroozi[b]

[a] *Department of industrial engineering and management, Shahrood university of technology, Shahrood, Iran*
[b] *Department of engineering, Alzahra university, Tehran, Iran*
Received 14 January 2020; Revised 21 January 2021; Accepted 24 January 2021

Abstract

This paper investigates a flexible flow shop scheduling problem with the aim of minimizing the operational costs as a new objective function. In this production system, there are some unrelated parallel machines with different performances and different technology levels in the first stage and each other stage consists of a single machine. Setup times are assumed as sequence-dependent and are need when a machine starts to process a new job. Some of the parallel machines in the first stage are multifunctional and can do several processes on jobs. So, the jobs that are assigned to these machines do not need to be processed in some next stages. This problem is described with an example, and its parameters and decision variables are defined. Then a mathematical model based on mixed-integer linear programming (MIP) is developed to solve the problem in small-sized scales. As this problem is discussed in an Np-hard environment, the Genetic Algorithm (GA) is applied to solve the considered problem on practical-sized scales. Due to the result, the operational costs conflict with makespan as a common objective function in scheduling problems. Therefore, the supplementary analysis has been presented considering a restriction on the makespan.

Keywords: Scheduling; Flexible flow shop; Unrelated parallel machines; Operational costs

1. Introduction

Production planning and control is one of the most activities for operational managers. Due to the increasing effects of internal and external factors in the nowadays competitive environment, the importance of production planning has been increased. Especially, considering the rising trend of the cost of production and marketing, it is essential that we have a perfect production plan in order to increase resource productivity as a competitive approach for manufacturing industries. Scheduling as an important shop floor planning activity is one of the most critical issues in production planning and control systems that are interacting with the internal and external factors of a production system. This is because of the strong competition and limitation of resources in our environment (Cummings and Egbelu 1998;Maleki-Darounkolaei et al. 2012). Product sequencing and resource allocation are two important tasks in the scheduling process. Sequencing is related to the processing order of jobs on each machine or stage, and allocation refers to select a machine for processing each job. Both jobs and machines may have specific characteristics and limitations that can affect resource productivity and the cost of production. These complexities are more tangible in make to order (MTO) manufacturing systems. Flow shop is an important aspect of MTO manufacturing systems that has many applications in the real world (Enayati et al. 2021). In order to increase flexibility in producing different products and raising reliability, this manufacturing system is extended to the flexible flow shop in which we have more than one machine in some stages. Using a flexible flow shop causes reduction effects of disturbances especially in case of disturbance in machines such as unexpected breakdown and other resource availability. The flexible flow shop that also called hybrid flow shop (HFS), compound flow shop, and multi-processor flow shop is a generalization of the flow shop in such a way that every job can be processed by one among several machines on each processing stage (Gupta and Tunc 1991; Li 1997; Fattahi et al. 2013b; Raissi et al. 2019). Flexible flow shops are common manufacturing environments in which a set of n jobs are needed to be processed in a series of m stages optimizing a given objective function. The common assumptions and characteristics of the classic flexible flow shop have been defined as bellows in the last studies (Huang and Li 1998; Fattahi et al. 2013a; Hosseini 2016) :

1. The number of processing stages (m) is at least 2.
2. Each stage k consists of $M^{(k)}$ machines in parallel format ($M^{(k)} \geq 1$ for all stages and $M^{(k)} > 1$ for at least one stage).

*Corresponding author Email address: sh.hosseini51@gmail.com

3. All jobs are processed following the same production flow: stage 1, stage 2, . . ., stage m. A job might skip any number of stages provided it is processed in at least one of them.

4. Each job j requires a processing time P_{jk} in stage k. In this problem, P_{ijl} is the processing times of the job j on the machine i in stage l. In a specific stage, when $P_{ij} = P_i$ for all i and j, the machines have the same speed and the processing times of a job are identical on different machines and the machines are called identical. When P_j/S_i where P_j is the processing time of job j and S_i is the speed of machine i, then the machines are called uniform. If the P_{ij} are arbitrary, then the machines are called unrelated. Both of the uniform and unrelated cases belong to the non-identical parallel-machine scheduling problem.

The present study aims to conduct a flexible flow shop scheduling problem in which there are some unrelated parallel machines in the first stage and we have only one machine for the other stages. Each job requires L operations to be processed. In addition to the velocity, the parallel machines in the first stage have different levels of technology and operational cost. So, depending on the first stage, jobs may not require to be processed in all next stages. The total operational cost of l stages for all jobs is considered as a new objective to be minimizing.

The outline of the paper is as follows. Section 2 is devoted to the survey of works related to this paper. The problem definition and the mathematical model are presented through a numerical example in section 3. The solution approach is described in section 4. The design of test problems and experiments is provided in section 5 while the evaluation of the proposed algorithm and analysis of the result is done in section 5 too. Finally, a summary of the work and direction for future research are given in section 6.

2. Literature Review

The flexible flow shop scheduling (FFS) problem has received much attention in recent years because of its many applications in different manufacturing industries (Wang et al. 2013; Hosseini 2019). This problem was defined as a multi-stage flow shop with multiple parallel machines for the first time in the 1970s (Choi and Wang 2012). This system is usually considered as a generalized format of two fundamental scheduling systems: (1) the flow shop scheduling problem, and (2) the parallel machines scheduling problem (Wang 2005). The types of parallel machines in each stage are the most important criteria that determine the scheduling environment in the FFS problem. These criteria can affect the solution approaches and play a key role in defining the optimization constraints. In real-world cases, older machines are replaced with newer ones gradually because of high replacement costs. So, it is common that unrelated machines with various technology levels and different efficiency are used in parallel for the same production line. This situation is defined as unrelated parallel machines (Jungwattanakit et al. 2008).

A flexible flow shop scheduling problem with unrelated parallel machines was introduced by Low (Low 2005) in which the independent setup was considered for the jobs. After that, Sawik addressed the same problem as Low with multi-capacity machines in a make-to-order system for long- and short-term machine assignment. He developed a new hierarchical approach to solve this problem (Sawik 2006). Jenabi et al. conducted the FFS problem with unrelated parallel machines to determine the economic lot-sizing over a finite planning horizon (Jenabi et al. 2007). So, they considered minimizing the sum of setup and inventory holding costs as a new objective function and developed a new mixed zero–one nonlinear mathematical programming for this problem. Chen and Chen studied the FFS problem with unrelated parallel machines on each stage considering a bottleneck stage on the line to minimize the total tardiness. They proposed two bottleneck-based heuristics with three machine selection rules to solve this problem (Chen and Chen 2009). Seven commonly dispatching rules and a basic Tabu search algorithm were investigated for comparison purposes. Computational results show that the bottleneck-based heuristics significantly outperform all the dispatching rules in solving the problem. Li et al. studied an FFS problem with unrelated parallel machines on each stage. They represented the problem as a nonlinear programming (NLP) optimization model and proposed a new heuristic method to solve it with a real-sized scale. They considered makespan and weighted tardiness as two important objectives (Li et al. 2015). Shahvari et al. studied the FFS problem where there are some unrelated parallel machines in bottleneck stages aiming to minimize the total weighted completion time and tardiness simultaneously. They extend their study by considering dynamic machine availability and job release times (Shahvari and Logendran 2017). Arroyo et al. also studied this problem considering release times and arbitrarily sized jobs. They developed a mixed-integer linear programming (MIP) model and proposed some heuristic approaches based on first-fit and best-fit earliest job-ready time rules (Arroyo and Leung 2017a; Arroyo and Leung 2017b). Some new studies have considered the energy consumption issue in flexible flow shop scheduling. For example, Meng et al. studied the FFS problem with setup times and proposed a mixed integer model with minimizing total energy consumption as a new objective function (Meng et al. 2019). Unal et al. studied the FFS problem with unrelated parallel machines while a bottleneck process within the production system motivated their problem. They developed an integer programming (IP) model to minimize the total tardiness of jobs over a finite planning horizon (Ünal et al. 2019). A re-entrant hybrid flow shop scheduling problem (RHFSP) has been introduced by Geng et al. in a new study (Geng et al. 2020). They developed a new improved

multi-objective ant lion optimization (IMOALO) algorithm to tackle this problem with the aim of minimizing total completion time (makespan) and total cost of energy consumption under Time-of-Use electricity tariffs. Schulz et al. discussed the FFS problem considering different processing speeds for parallel machines aiming to minimize total tardiness and energy costs simultaneously. They presented two new formulations for this problem and applied them to solve some numerical examples (Schulz et al. 2020). Recently, Hasani and Hosseini studied the flexible flow shop scheduling problem with unrelated parallel machines by considering the production cost and energy consumption as two objective functions and a machine-dependent processing stage. they solved the MIP model using the $\varepsilon-$ constraint II method on a small-size scale and NSGA II and SPEA 2 on a large-size scale (Hasani and Hosseini 2020).

The setup time that is usually required when a machine switches to a new kind of job, plays an important role in the optimization process. Considering sequence-dependent setup times in flexible flow shop, make it even more difficult NP-hard problem as stated by Pinedo and Lee et al. (Pinedo 2008; Lee and Loong 2019). Under such a condition, the decision made for the processing flow influences the processing time of the jobs (Ahonen and de Alvarenga 2017). Kianfar et al. studied a sequence-dependent setup time of the FFS system by considering the non-deterministic and dynamic arrival of jobs (Kianfar et al. 2012). In the paper of Ebrahimi et al., a stochastic model has been proposed to solve the FFS problem with uncertain due dates and sequence-dependent family setup times. The due dates are assumed to be uncertain and followed a normal distribution in their study (Ebrahimi et al. 2014). The sequence and machine-dependent setup times in flexible flow shop scheduling problem with unrelated parallel machines were addressed by Edwin et al. aiming to minimize makespan. They emphasized on complexity of the problem and proposed a solution approach based on the imperialist competitive algorithm (ICA) and the genetic algorithm (GA) to solve this problem (Garavito-Hernández et al. 2019).

The objective function is another important issue in all of the scheduling problems. In shop floor planning, the objective functions can be addressed as time-related, job-related, and multi-objective functions. Time-related objective functions usually aim to minimize the total completion time (makespan), the completion time of each job, cycle time, and flow time. Job-related objectives are known as job tardiness and earliness (Lee and Loong 2019). Although single objective models cannot truly represent the exact situation facing real-world situations, the last studies show that there are not many papers that deal with multiple objectives (Lee and Loong 2019; Ruiz and Vázquez-Rodríguez 2010).

Figure (1) represents a distribution of past related studies considering different kinds of objective functions. Also, different kinds of parallel machines that have been investigated in flexible flow shop scheduling problems have been shown in figure (2) (Lee and Loong 2019).

As the best knowledge of the authors, no study conducted the FFS problem considering minimizing operational costs as the objective function. Due to the use of unrelated machines in some stages with different efficiency, the operational cost of each job will be different based on the processing path. Therefore, this study aims to minimize the operational costs in an unrelated flexible flow shop production system considering makespan as a constraint.

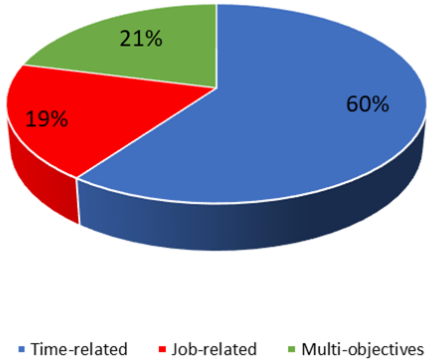


Fig. 1. Objective function classification in FFS.

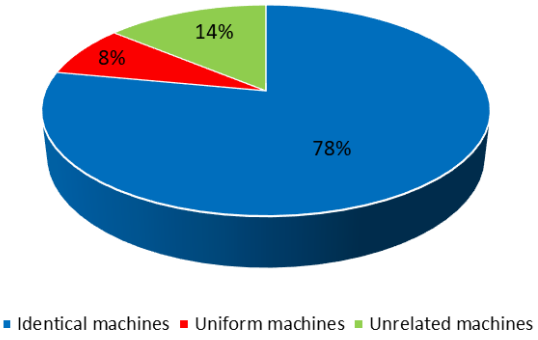


Fig. 2. Machines constraint distribution in FFS.

3. Problem Definition

As is evident in the existing literature, many studies have investigated the flexible flow shop scheduling problem in recent years, most of which seek to reduce production time and none of them have focused on reducing operational cost directly. In this section, we seek to provide a model to reduce operational costs in FFS considering a specified value for makespan as a constrain. Figure (3) presents a schematic view of the considered problem in this study. Suppose that there are N jobs that should be processed in a flexible flow shop system with some unrelated parallel machines in stage 1 and each other stages consist of a single machine. Setup times are assumed as sequence-dependent and are required when a

machine starts to process a new job. Some parallel machines in stage 1 are high-tech and so, the jobs that assign to these machines do not need to be processed in some next stages. To better understand the problem at hand, a proper formulation is provided as bellows.

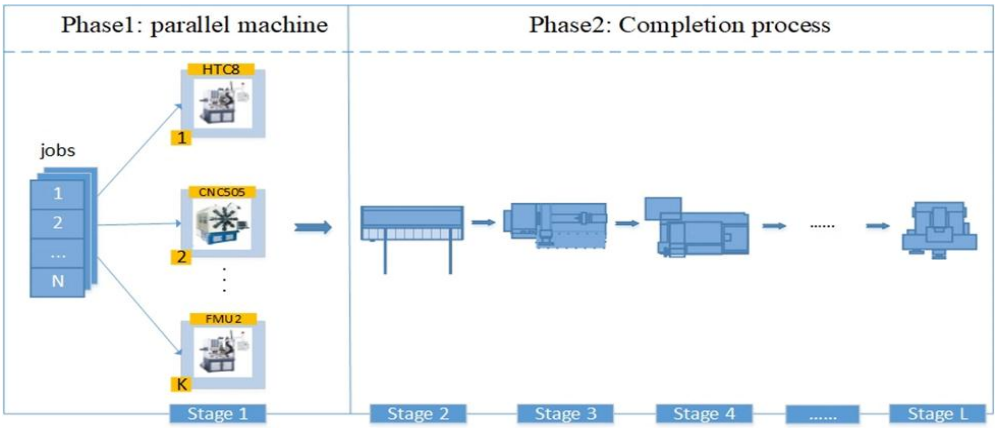


Fig. 3. A schematic view of the considered

3.1. Notations

Indices of the considered problem are as follows:

$j, i, h = 0,1,2,\dots,N$	Indices for jobs
$k, m = 1,2,\dots,K_l$	Indices for parallel machines in stage l
$l = 1,2,\dots,L$	Index for stages

Parameters of the considered problem are as follows:

$P_{kj}^{(l)}$	Processing time of job j on machine k in stage l
$S_{ijk}^{(l)}$	Setup time for switching job i to job j on machine k in stage l
$CP_{kj}^{(l)}$	Cost per hour working machine k on job j in stage l. (in processing mode or setup operation).
Y_{lkj}	Binary parameter taking value 1 if job j enter the stage l after processing by machine k in the first stage and 0 otherwise.
M	A large number.
V	An employer limit value for the makespan.

Also, the decision variables of the proposed model are as follows:

$Z_{jk} = \begin{cases} 1 \\ 0 \end{cases}$	Binary variable taking value 1 if job ***j*** is assigned to machine ***k*** in the first stage and 0 otherwise.
$X_{ijkl} = \begin{cases} 1 \\ 0 \end{cases}$	Binary variable taking value 1 if job ***j*** is processed directly after job ***i*** on machine ***k*** in stage ***l*** and 0 otherwise.
$C_j^{(l)}$	Completion time of job ***j*** in stage ***l***.
$OC_j^{(l)}$	Operational cost of job ***j*** in stage ***l***.

3.2. Assumptions

The main assumptions of the considered problem are as follows. Many of these items are known as general assumptions in FFSP and the last two have been added to this paper.

- All jobs and machines are available at time zero.
- There are some unrelated parallel machines in stage 1 and one machine in each other stages.
- Each machine can process only one job at the same time, and each job can be processed only by one machine at the same time.
- Each job can only be processed while its required machine is available during its processing time.
- The setup times are assumed as sequence-dependent.
- To consider the initial setup times, we assume that job 0 is a dummy job with zero release date and processing time which should be placed in position one on all machines.
- Transportation times of the jobs between machines are neglected
- The priority of processing the jobs after the first stage is FIFO.
- Production cost per hour for machines is the same both in processing mode and setup operation.
- The jobs assigned to the high-tech machines in the first stage will skip some next stages.

3.3. Mathematical formulation

In this section, a new mixed-integer linear programming (MIP) model is developed for the problem at hand. This model is developed based on the study of Afzali Rad et al. (Afzalirad and Rezaeian 2016) and Handbook on Scheduling (Blazewicz et al. 2019).

$$Min\ Z = \sum_{l=1}^{L}\sum_{j}^{N} OC_j^{(l)} \quad (1)$$

Subject to:

$$\sum_{i=0}^{N}\sum_{k=1}^{K} X_{ijkl} = 1 \quad , \quad \forall\, l = 1, j \neq 0\,, i \neq j \quad (2)$$

$$\sum_{i=0}^{N}\sum_{k=1}^{K} X_{ijkl} - \sum_{k=1}^{K} Z_{jk} * Y_{jkl} = 0 \quad , \quad \forall\, l \neq 1, j \neq 0\,, i \neq j \quad (3)$$

$$\sum_{j=0}^{N} X_{ijkl} \leq 1 \quad , \quad \forall\, k, l \neq 1, i \neq j \quad (4)$$

$$\sum_{i=0}^{N} X_{ihkl} - \sum_{j=0}^{N} X_{hjkl} = 0 \quad , \quad \forall\, j \neq h\,, i \neq h \quad (5)$$

$$C_j^{(l)} \geq X_{ijkl}\left(P_{kj}^{(l)} + S_{ijk}^{(l)}\right) \quad , \quad \forall\, l = 1, i \neq j \quad (6)$$

$$C_i^{(l)} + \sum_{k=1}^{K} X_{ijkl}\left(P_{kj}^{(l)} + S_{ijk}^{(l)}\right) + M\left(\sum_{k=1}^{K} X_{ijkl} - 1\right) \leq C_j^{(l)} \quad \forall\, i, j \neq 0 \quad (7)$$

$$C_j^{(l-1)} + X_{ijkl}\left(P_{kj}^{(l)} + S_{ijk}^{(l)}\right) \leq C_j^{(l)} \quad \forall\, l \neq 1, j \neq 0\,, i \neq j \quad (8)$$

$$C_j^{(l-1)} + M(1 - X_{ijkl}) \geq C_i^{(l-1)} \quad , \quad \forall\ l \neq 1 \quad (9)$$

$$Z_{jk} = \sum_{i=0}^{N} X_{ijkl} \quad , \quad \forall\, l = 1, i \neq j \quad (10)$$

$$OC_j^{(l)} \geq CP_{kj}^{(l)} * \left(P_{kj}^{(l)} + \left(S_{ijk}^{(l)} * X_{ijkl}\right)\right) \quad \forall\, l = 1, j \neq 0\ i \neq j \quad (11)$$

$$OC_j^{(l)} \geq CP_{kj}^{(l)} * \left(P_{kj}^{(l)} + \left(S_{ijk}^{(l)} * X_{ijkl}\right)\right) - M\left(1 - \sum_{k=1}^{K} Z_{jk} * Y_{lkj}\right) \quad \forall\, l \neq 1, j \neq 0\ i \neq j \quad (12)$$

$$OC_j^{(l)} - M\left(\sum_{k=1}^{K} Z_{jk} * Y_{lkj}\right) \leq 0\,, \quad \forall\, l \neq 1 \quad (13)$$

$$C_j^{(l)} \leq C_{max} \quad , \quad \forall\, j \quad (14)$$

$$C_{max} \leq V \quad (15)$$

$$X_{ijkl}, Z_{jk} \in \{0,1\} \quad (16)$$

$$OC_j^{(l)}, C_j^{(l)} \geq 0 \quad (17)$$

In the mathematical model, minimizing the total operational cost aspect of the problem is expressed by equation (1). Constraints (2), (3), (4), and (5) ensure that each job is processed precisely once at each stage. Constraints (6) take care of the completion time of job *j* in stage l. Constraints (7) indicate that the completion times $C_i^{(l)}$ and $C_j^{(l)}$ of jobs J_i and J_j scheduled consecutively on the same machine respect this order. On the other hand, Inequalities (8) imply that jobs go through the stages in the right order, i.e., from stage 1 to stage L. Constraints (9) use the FIFO sequencing rule to extend sequences between stages. More specifically, this set ensures that job i is sequenced before job j at stage l if the completion time of j is greater or equal than the completion time of i at stage $l-1$. Constraints (10) Show the machine selected in stage one. Constraints (11), (12) and (13) take care of the Operational cost of job *j* in stage l. Constraints (14) are used to compute the makespan. Due to the constraint (15), all of the jobs should be completed until a given time. Finally, the last two constraints specify the domains of the decision variables.

3.4. A numerical example

In order to clarify the considered problem and the proposed mathematical model, a simple numerical example is investigated in this section. Assume that, there are three parallel machines in stage 1 with three different technology levels. The first stage is followed by four other stages in a flow format that each stage consists of a single machine. Four jobs are needed to be produced in this system. The data for jobs, their processing times, setup times, and operational costs are given in table (1). Due to the difference in technology levels of the parallel machines, some stages may be deleted as the table (2).

Table 1

Parameters for the jobs in the numerical example.

	Process times					Operational costs			
Jobs	1	2	3	4		1	2	3	4
$P_{1j}^{(1)}$	8	10	6	10	$CP_{1j}^{(1)}$	30	30	30	30
$P_{2j}^{(1)}$	6	8	4	8	$CP_{2j}^{(1)}$	12	12	12	12
$P_{3j}^{(1)}$	4	5	3	5	$CP_{3j}^{(1)}$	10	10	10	10
$P_{1j}^{(2)}$	4	5	5	6	$CP_{1j}^{(2)}$	8	8	8	8
$P_{1j}^{(3)}$	10	14	12	15	$CP_{1j}^{(3)}$	6	6	6	6
$P_{1j}^{(4)}$	4	6	4	8	$CP_{1j}^{(4)}$	3	3	3	3
$P_{1j}^{(5)}$	5	5	4	5	$CP_{1j}^{(5)}$	2	2	2	2

Setup times

	$S_{ij1}^{(1)}$					$S_{ijk}^{(1)}$ $(k = 2,3)$			
i/j	1	2	3	4	i/j	1	2	3	4
0	4	6	4	5	0	2	3	2	2
1	0	6	4	5	1	0	3	2	2
2	4	0	4	5	2	2	0	2	2
3	4	6	0	2	3	2	3	0	2
4	4	6	2	0	4	2	3	2	0

The setup times in stages 2-4 are assumed as a part of process time

This example has been investigated in three modes. At first, we generated a random schedule for jobs, and the result is shown in table (3) and its Gant chart is represented as figure (4) with value Z=1260. Then we solved this problem with the proposed mathematical model. The result is shown in table (4) and figure (5) with value Z = 812. That is obvious from the result that we have more than 35% improvement in the operational costs as the objective function. The result of the optimal solution emphasizes that the high-tech machines with more operational costs (in this example machine 1) will not be assigned to process any jobs when the operational costs are the only objective. We also optimized the operational costs with restriction makespan on the obtained value (C_{max}=53) to more test the proposed model. As the result shows, the model could reduce the operational costs from 1260 to 1102 (more than 12.5% improvement) with the same makespan. This result is shown in table (5) and its Gant chart is represented in figure (6).

Table 2

Required to process according to the machine assigned in the first stage (Y_{lkj}).

Assignment in the First stage	stages			
	2	3	4	5
Y_{l11}	1	0	1	1
Y_{l21}	0	1	1	1
Y_{l31}	1	1	1	1
Y_{l12}	1	0	1	1
Y_{l22}	0	1	1	1
Y_{l32}	1	1	1	1
Y_{l13}	1	0	1	1
Y_{l23}	0	1	1	1
Y_{l33}	1	1	1	1
Y_{l14}	1	0	1	1
Y_{l24}	0	1	1	1
Y_{l34}	1	1	1	1

Table 3

Machine assignment in a random schedule.

jobs	stage 1			stage 2	stage 3	stage 4	stage 5	Operational cost
	Machine 1	Machine 2	Machine 3					
1	✓			✓		✓	✓	414
2		✓			✓	✓	✓	244
3	✓			✓		✓	✓	360
4			✓	✓	✓	✓	✓	242
								1260

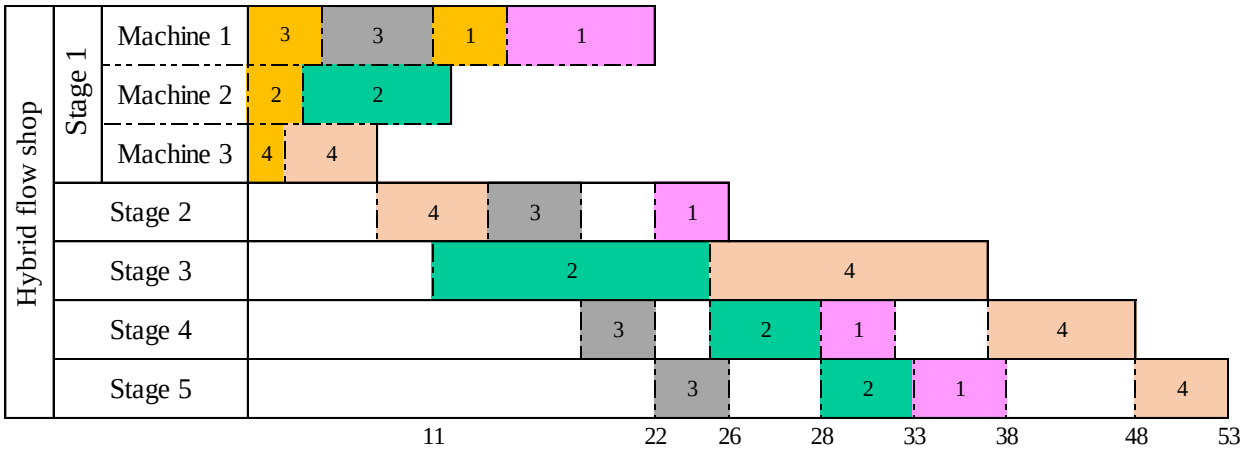


Fig. 4. The Gantt chart of the random schedule.

Table 4

Machine assignment in the optimal schedule

jobs	stage 1			stage 2	stage 3	stage 4	stage 5	Operational cost
	Machine 1	Machine 2	Machine 3					
1			✓	✓	✓	✓	✓	174
2			✓	✓	✓	✓	✓	232
3		✓			✓	✓	✓	164
4			✓	✓	✓	✓	✓	242
								812

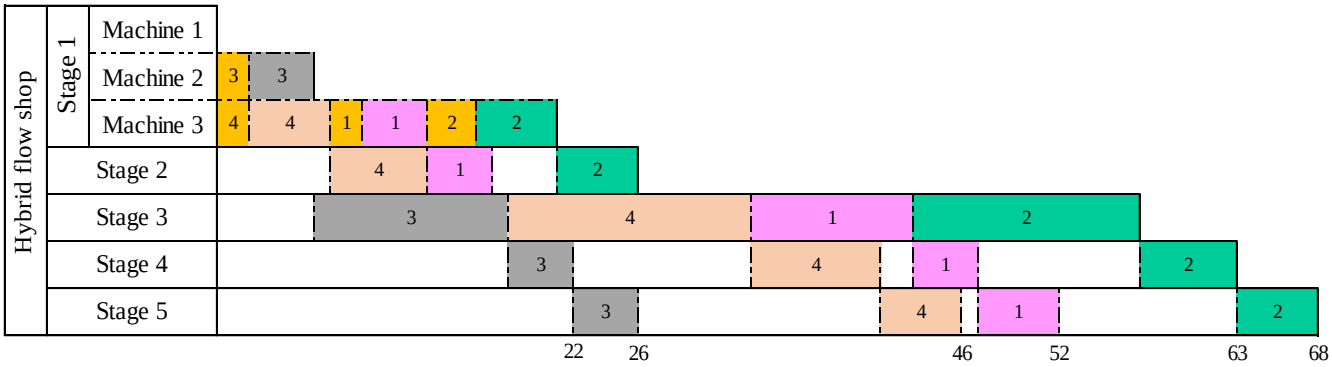


Fig. 5. The Gantt chart of the optimal schedule

Table 5

Machine assignment in improved the random schedule.

jobs	stage 1			stage 2	stage 3	stage 4	stage 5	Operational cost
	Machine 1	Machine 2	Machine 3					
1			✓	✓	✓	✓	✓	174
2			✓	✓	✓	✓	✓	232
3		✓			✓	✓	✓	164
4	✓			✓		✓	✓	532
								1102

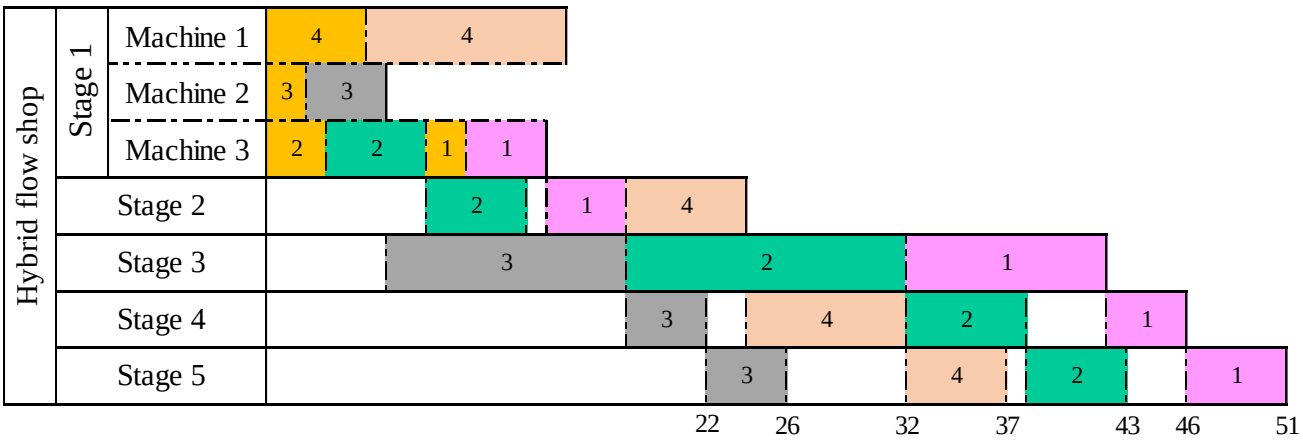


Fig. 6. The Gantt chart of improved the random schedule.

4. The proposed solution algorithm

The genetic algorithm (GA) is a population-based meta-heuristic approach introduced by Holland (1975) based on the evolution process. This algorithm starts with an initial population as primary feasible solution. After that, two operators including crossover and mutation are applying to create some offsprings as a new solution. A crossover operator is done to mix the good features of parents in order to generate more fitting offspring. The mutation operator helps the algorithm to maintain diversity within the population and escape from probably local optimum. Among total old and new solutions, a population with a definite size is selected as the new generation. This process is repeated until a specific stopping criterion is met (Hosseini 2017). Due to the complexity of the considered problem in this study, this algorithm is used to solve it in a real-size scale. In this section, a solving algorithm is proposed for the real-size scale of the considered problem based on GA. How to implement the proposed GA is described in the following.

4.1. Solution representation

Implementation of a meta-heuristic needs to decide how to represent solutions in an efficient way to the searching space (Fattahi et al. 2012). Representation should be easy to decode and calculated to reduce the run time of the algorithm. In the considered problem, several jobs need to be scheduled in a flexible flow shop in which that there are some unrelated machines in the first stage and there is only one machine in each of the remaining stages. By regarding these aspects, the appropriate schematic structure of the problem solution contains $L+1$ strings in which that a permutation of the N jobs is considered above the first string. In the first string, assigning the parallel machines to process the jobs is determined. In the second string, the priority of jobs is identified to be processed on an assigned machine. The third string to $L+1$ is completed for job sequencing on stage 2 to L. Figure (7) shows an example of this structure with $N=10$, $K_1=3$, and $L=5$.

		Jobs									
		1	2	3	4	5	6	7	8	9	10
Stage 1	Machine assigning:	2	1	1	3	2	2	3	3	3	1
	Job sequencing:	3	2	3	4	1	2	2	3	1	1
	Jobs sequence on stage 2:	8	2	3	4	5	1	7	10	9	6
	Jobs sequence on stage 3:	3	2	10	4	5	6	7	8	9	1
	Jobs sequence on stage 4:	1	9	3	4	2	6	7	8	10	5
	Jobs sequence on stage 5:	1	8	6	3	10	9	4	5	7	2

Fig. 7. The schematic structure of the problem solution for GA

Journal of Optimization in Industrial Engineering
Vol.14, Issue 1, Winter & Spring 2021, 169-184
DOI:10.22094/JOIE.2021.1890852.1718

4.2. Create initial population

The GA is a population-based meta-heuristic, so it needs an initial population of individuals to start. This population can either be produced randomly or based on some other algorithms. In this study, each chromosome of the initial population is generated randomly.

4.3. Calculation the fitness value of the solutions

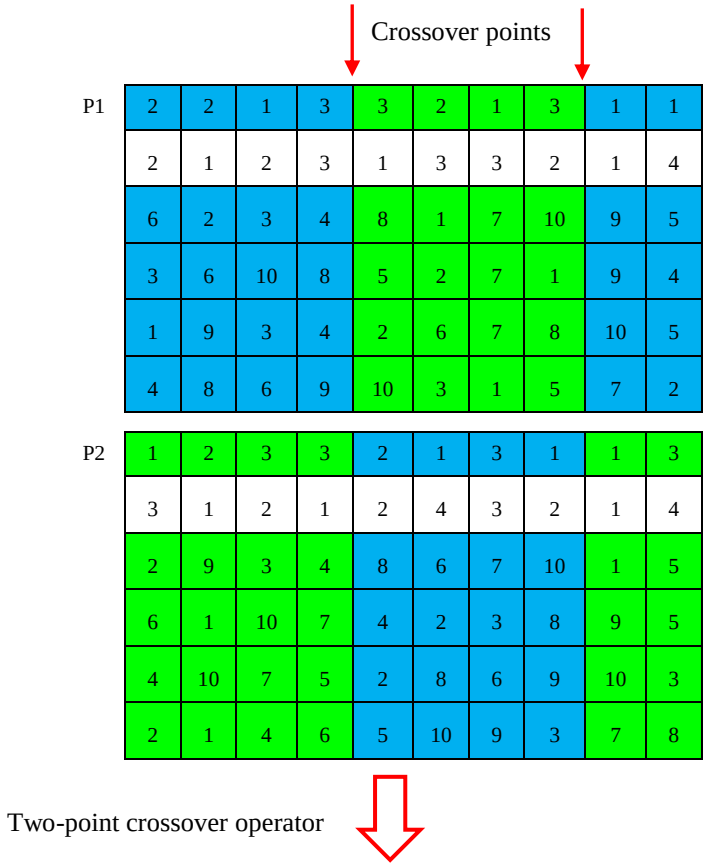
After generating the initial population, the value of operational costs is calculated for each solution and is considered as the fitness value of each solution (fv(x)). Then solutions (chromosomes) are ranked based on their fitness.

4.4. Parent selection method

The parent selection method determines how to choose the chromosomes from the current population for the crossing procedure. In this study, the parents are selected based on the roulette-wheel rule. After calculation, the fitness value (fv(x)) for each solution, the operation of the roulette-wheel is done to select the parents.

4.5. Recombination and mutation

In general, the crossover operator is done to generate a new chromosome (offspring) with more fitness. After that, the mutation operator is implemented to maintain the diversity of the population in the successive generations and to more exploit the solution space. Some different methods were tested for crossover and finally, two-point crossover (2PX) was recognized well than the others. The results also emphasized that the crossover operation on the jobs sequencing for each parallel machine does not any significant improvement on objective functions. So, in order to increase the efficiency of the proposed algorithm; the crossover is done only on the machine assignment in the first stage and jobs sequencing in stages 2-3. The mutation operator is used either for the machine assignment in the first stage, jobs sequencing for each parallel machine in the first stage, and on jobs sequencing after crossover. Here, replacement mutation is used with random selection. Figure (8) presented a sample of the two-point crossover (2PX) used for the considered algorithm.



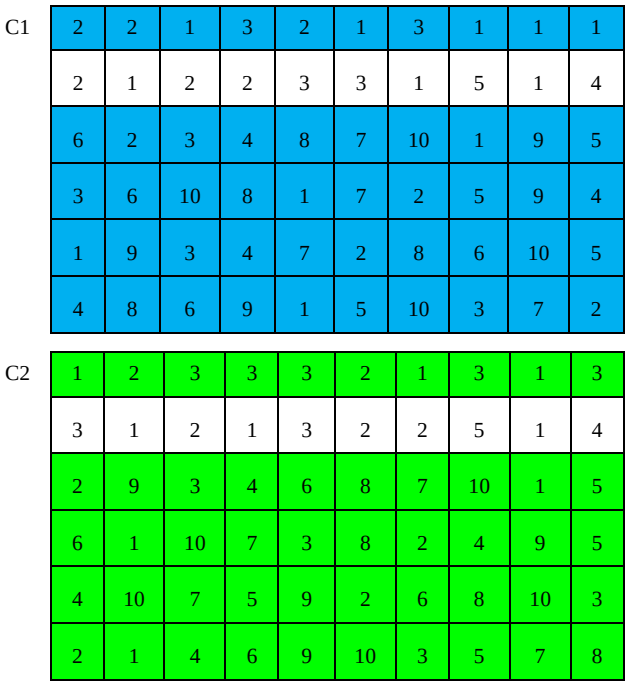
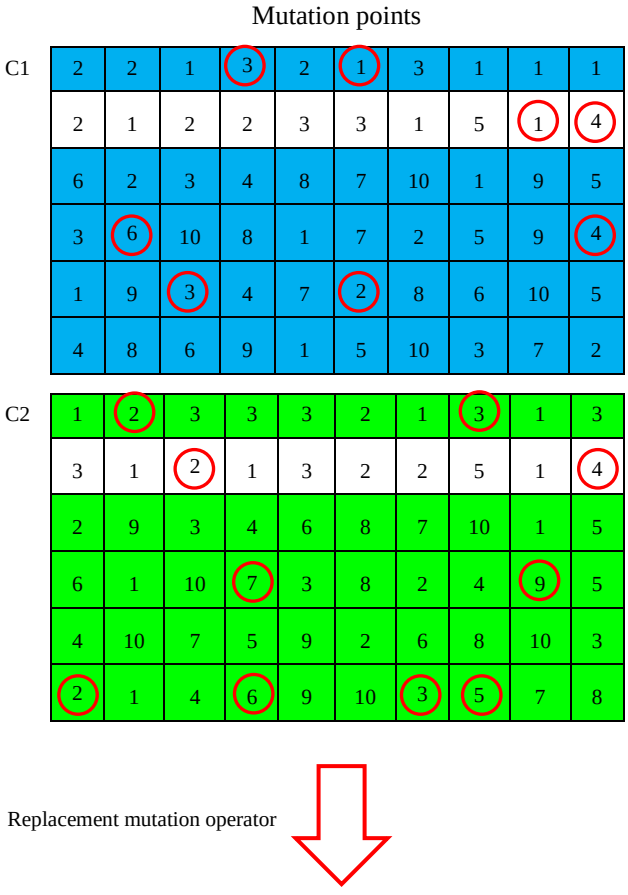


Fig. 8. The proposed crossover

A sample of the mutation operator used in this study has been presented via figure (9). The mutation is done with a couple of replacement of the randomly selected genes on the jobs sequencing and machine assignment. It is obvious that we need to correct jobs sequencing on the parallel machines after changing in machine assignment in the first stage. This is done by a correction sub-algorithm after complete the mutations.



C1

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

C2

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

Fig. 9. The proposed mutation

4.6. Selection the new generation P_{t+1}

The new generation P_{t+1} is selected from the parents and offsprings but not just based on the fitness values. Rather than, in each generation of the proposed GA, a new population is generated by replacing the second 50% of the current population with the first 50% of the new population (offsprings). Based on this procedure, there is no guarantee that the inserted solutions are always better than the worst solutions in the current population. This approach makes the algorithm towards searching the various regions and avoids form a premature convergence during the progress of the algorithm.

The best value of the parameters for the proposed GA algorithm is obtained using Taguchi settings considering the plan of L_{27} in three levels. In order to determine the best combination of these parameters, three levels of each parameter were examined as the table (6).

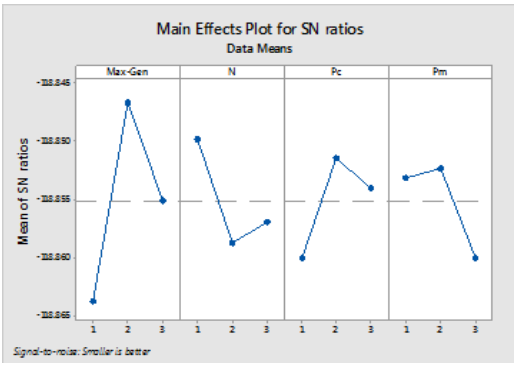


Fig. 10. Main effects plot for S/N ratios with the Taguchi method.

Table 6

The test values of the parameters for the proposed GA

Parameter	Number of levels	Test values
Max-Gen	3	50, 150, 300
N	3	20, 30, 50
P_c	3	0.7, 0.8, 0.9
P_m	3	0.05, 0.1, 0.15

Finally, after doing experiments as figure (10), the best combination of the parameters for the proposed genetic algorithm determined as below:

Max-Gen: 150, N: 20, Pc: 0.8, Pm: 0.1

5. Comparison of results

This section presents the results of solving test problems using the mathematical model and GA. The mathematical model was run in GAMS by the CPLEX solver and the proposed algorithm was coded in MATLAB 7/10/0/499 (R2010a). The experiments are executed on a Pc with a 2.0GHz Intel Core 2 Duo processor and 4GB of RAM memory. The test problems were categorized into three classes contain small, medium, and large-sized problems. For the proposed algorithm, each problem has been run ten times and the best and or the average of results are evaluated.

Characteristics of test problems are shown in table (7). Additional information about these problems is that all jobs assigned to new machines in the first stage will pass stage 2

Table 7

Data of the test problems data

Problem size	Problem name	N	K_1		$S_{ijk}^{(1)}$		$P_{kj}^{(1)}$		$S_{ij1}^{(l)}$ $(l \geq 2)$	$P_{1j}^{(2)}$	$P_{1j}^{(3)}$	$P_{1j}^{(4)}$	$P_{1j}^{(5)}$	$CP_{kj}^{(1)}$		$CP_{kj}^{(l)}$ $(l \geq 2)$
			new	old	new	old	New	old						new	old	
Small	I	4	1	2	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
	II	5	1	3	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
Mediocre	III	10	2	4	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
	IV	15	3	5	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
	V	20	4	6	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
	VI	25	4	7	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
Large	VII	50	5	7	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
	VIII	75	5	8	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
	IX	100	6	10	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]
	X	150	7	10	[200 ,300]	[100 , 250]	[30 , 80]	[25 , 75]	[100 , 200]	[15 , 25]	[5 , 10]	[50 , 75]	[20 , 30]	[50 , 100]	[10 , 20]	[15 , 50]

Table 8

Machine assignment to process jobs considering restriction on total completion times

Problem	N	New 1	New 2	New 3	New 4	New 5	New 6	New 7	Old 1	Old 2	Old 3	Old 4	Old 5	Old 6	Old 7	Old 8	Old 9	Old 10
I	4	1	-	-	-	-	-	-	1	2	-	-	-	-	-	-	-	-
II	5	2	-	-	-	-	-	-	1	1	1	-	-	-	-	-	-	-
III	10	2	2	-	-	-	-	-	2	2	1	1	-	-	-	-	-	-
IV	15	2	1	2	-	-	-	-	2	2	2	2	2	-	-	-	-	-
V	20	2	2	2	2	-	-	-	2	2	2	2	2	2	-	-	-	-
VI	25	3	3	3	2	-	-	-	2	2	2	2	2	2	2	-	-	-
VII	50	3	4	4	4	3	-	-	4	4	5	5	5	5	4	-	-	-
VIII	75	5	5	5	4	5	-	-	7	6	6	7	7	6	6	6	-	-
IX	100	6	5	6	5	5	5	-	7	7	7	6	7	7	6	7	7	7
X	150	6	7	6	6	6	6	7	10	10	11	11	11	10	11	10	11	11

Journal of Optimization in Industrial Engineering
Vol.14, Issue 1, Winter & Spring 2021, 155-170
DOI:10.22094/JOIE.2021.1890852.1718

Table (9) shows the result of solving the test problems using the proposed genetic algorithm. This result has been obtained without any restriction on the total completion time of the jobs. For the small-sized problems, this algorithm could achieve the optimal solution with $\%D = 0$. For these solutions, no restriction was applied to the total completion times, and this function just has been calculated as an output. Due to this result, both of mathematical model and GA presented the same value for makespan in small-sized problems.

The other result that is presented in table (10) is about solving the same ten problems considering a restriction on the total completion times for each problem. These restrictions have been considered as a desirable level for the total completion time of the jobs in a way that the problems be feasible with a challenging amount of the operational costs. This result emphasizes the conflict between these two important objectives. Reducing the makespan of the problem, force the algorithm to assign more jobs to newer machines in the first stage and so, inevitably the total operational costs will increase.

Table (8) addresses machine assignments in the first stage based on new and old machines separately. It is obvious that more new machines will assign by reduction of the total completion time until all jobs should be processed by new and more expensive machines. In addition, fewer jobs will need to be processed with new machines by allowing to rise the total completion time.

Table 9

The test problems data without any restriction on C_{max}

Problem size	Problem name		Total cost			C_{max}			
		N	GA	GAMS	D%	GA	GAMS	D%	CPU time
Small	I	4	34750	34750	0	1747	1747	0	2.1
	II	5	44200	44200	0	1953	1953	0	2.6
Mediocre	III	10	87500	-	-	3724	-	-	4.8
	IV	15	131250	-	-	5402	-	-	7.2
	V	20	175000	-	-	7020	-	-	11.6
	VI	25	218750	-	-	8620	-	-	12.8
Large	VII	50	437500	-	-	17240	-	-	19.4
	VIII	75	656250	-	-	25515	-	-	32.7
	IX	100	875000	-	-	33367	-	-	45.7
	X	150	1321800	-	-	45176	-	-	54.9

Table 10

The test problems data with restriction on C_{max}

Problem size	Problem name	N	Total cost			Restriction value on C_{max}	CPU time
			GA	GAMS	D%		
Small	I	4	41125	41027	0.24%	1355	1.9
	II	5	50256	50102	0.31%	1572	2.4
Mediocre	III	10	112601	-	-	3054	11.2
	IV	15	173216	-	-	4325	14.5
	V	20	198674	-	-	5617	18.9
	VI	25	262190	-	-	6957	33.7
Large	VII	50	543297	-	-	14252	78.9
	VIII	75	798509	-	-	20212	125.8
	IX	100	10275431	-	-	26710	212.3
	X	150	1674393	-	-	36221	345.8

According to this result, the proposed algorithm has to do more computing to obtain the best value for the operational costs regarding a specified amount of makespan. Therefore, the computation times increase in these cases and we have a little deviation from the optimal solution according to the result of the problems' number I and II.

Journal of Optimization in Industrial Engineering
Vol.14, Issue 1, Winter & Spring 2021, 155-170
DOI:10.22094/JOIE.2021.1890852.1718

Another output of solving these problems is about algorithm convergence in iterations. Figure (11) shows the algorithm convergence for problem VII as an example in two different conditions. When there is no restriction on the makespan, the proposed algorithm shows a fast convergence in comparison to it forced to regard a specified value for the C_{max}. In the case of the C_{max} is not restricted to any defined value, the algorithm could achieve the last solution during less than 60 iterations. This was anticipated because the total of the jobs is assigned to the old parallel machines with the least possible operational costs. The challenging issue happened when we want to reduce the operational costs considering a specified amount for the makespan. Under this condition tries to minimize the operational costs but the obtained solution may not be acceptable because of the defined constraint related to makespan. As the result shows, the algorithm could convergence on the final solution after 100 iterations.

Fig. 11. Convergence of the proposed algorithm in solving the problem VII

6. Summary and conclusion

A flexible flow shop production system consists of some unrelated parallel machines in the first stage and one machine in each other stages was studied in this paper. In this system, several jobs are ordered to be produced. All parallel machines can process all jobs at the first stage but some of these machines are high-tech and multi-functional with different operational costs per hour in comparison to the others that are low-tech machines. The jobs that are processed by high-tech machines do not need to be processed in some next stages. Setup times were considered as sequence-dependent and the objective function was to minimize the total operational costs. So, there are two key decision variables in this problem include: assigning the parallel machines to process the jobs in the first stage, and sequencing the jobs to be processed in all stages. This problem was described through a numerical example and modeled as mixed-integer linear programming (MIP). Since the problem has been proved strongly NP-hard, a method was developed based on GA to solve the medium- and large-sized problems.

The mathematical model has applied on a numerical example in small size to better understanding the considered problem and to investigate the performance of the proposed model. When we don't have any restrictions on makespan, the model assigns all jobs to the low-tech machines in the first stage to reduce the operational costs as the considered objective function. In case of makespan is restricted on a specified value, the jobs with the least operational costs on high-tech parallel machines are assigned to these machines to keep the total completion time.

In addition to the numerical example, some other test problems in practical sizes with various parameters were generated and solved using the proposed genetic algorithm. The result emphasized the conflict between C_{max} and operational cost in this production system again. Due to the achieved result, the proposed algorithm has a good performance in solving problems both in solution quality and run time. Of course, in the case of that, there is a restricted value on the C_{max}, the algorithm faces a more challenging situation and so it shows that the CPU time index raises in solving these kinds of problems. The convergence trend of the proposed algorithm confirms that the iterations converge on the final solution very fast when there is no constraint on the C_{max}.

For future studies, this problem can be investigated under uncertainty especially for the process times. Furthermore, it is recommended to apply other efficient heuristic or meta-heuristic algorithms for solving the considered

problem and comparison results with the proposed algorithm in this paper. Considering other important objective functions such as energy consumption can be another interesting field to study.

References

Afzalirad M, Rezaeian J (2016) Resource-constrained unrelated parallel machine scheduling problem with sequence dependent setup times, precedence constraints and machine eligibility restrictions. Comput Ind Eng 98:40–52

Ahonen H, de Alvarenga AG (2017) Scheduling flexible flow shop with recirculation and machine sequence-dependent processing times: formulation and solution procedures. Int J Adv Manuf Technol 89:765–777

Arroyo JEC, Leung JY-T (2017a) Scheduling unrelated parallel batch processing machines with non-identical job sizes and unequal ready times. Comput Oper Res 78:117–128

Arroyo JEC, Leung JY-T (2017b) An effective iterated greedy algorithm for scheduling unrelated parallel batch machines with non-identical capacities and unequal ready times. Comput Ind Eng 105:84–100

Blazewicz J, Ecker K, Pesch E, Schmidt G, Weglarz J (2019) Handbook on scheduling. Springer

Chen C-L, Chen C-L (2009) A bottleneck-based heuristic for minimizing makespan in a flexible flow line with unrelated parallel machines. Comput Oper Res 36:3073–3081

Choi SH, Wang K (2012) Flexible flow shop scheduling with stochastic processing times: A decomposition-based approach. Comput Ind Eng 63:362–373

Cummings DH, Egbelu MPJ (1998) Minimizing production flow time in a process and assembly job shop. Int J Prod Res 36:2315–2332

Ebrahimi M, Ghomi SMTF, Karimi B (2014) Hybrid flow shop scheduling with sequence dependent family setup time and uncertain due dates. Appl Math Model 38:2490–2504

Enayati M, Asadi-Gangraj E, Paydar MM (2021) Scheduling on flexible flow shop with cost-related objective function considering outsourcing options. J Optim Ind Eng 14:69–88

Fattahi P, Hosseini S, Jolai F (2013a) Some heuristics for the hybrid flow shop scheduling problem with setup and assembly operations. Int J Ind Eng Comput 4:393–416

Fattahi P, Hosseini SMH, Jolai F (2012) Using the Simulated Annealing to Solve a JIT Scheduling Problem in the Two-Machine Flow Shop. Int J Ind Eng 23:265–281

Fattahi P, Hosseini SMH, Jolai F (2013b) A mathematical model and extension algorithm for assembly flexible flow shop scheduling problem. Int J Adv Manuf Technol 65:787–802

Garavito-Hernández EA, Peña-Tibaduiza E, Perez-Figueredo LE, Moratto-Chimenty E (2019) A meta-heuristic based on the Imperialist Competitive Algorithm (ICA) for solving Hybrid Flow Shop (HFS) scheduling problem with unrelated parallel machines. J Ind Prod Eng 1–9

Geng K, Ye C, Liu L (2020) Bi-Objective Re-Entrant Hybrid Flow Shop Scheduling considering Energy Consumption Cost under Time-of-Use Electricity Tariffs. Complexity 2020:

Gupta JND, Tunc EA (1991) Schedules for a two-stage hybrid flowshop with parallel machines at the second stage. Int J Prod Res 29:1489–1502

Hasani A, Hosseini SMH (2020) A bi-objective flexible flow shop scheduling problem with machine-dependent processing stages: Trade-off between production costs and energy consumption. Appl Math Comput 386:in press . https://doi.org/10.1016/j.amc.2020.125533

Hosseini SMH (2016) Modeling the hybrid flow shop scheduling problem followed by an assembly stage considering aging effects and preventive maintenance activities. Int J Supply Oper Manag 3:1215

Hosseini SMH (2019) Modelling and solving the job shop scheduling Problem followed by an assembly stage considering maintenance operations and access restrictions to machines. J Optim Ind Eng 12:63–78

Hosseini SMH (2017) A multi-objective genetic algorithm (MOGA) for hybrid flow shop scheduling problem with assembly operation. J Ind Syst Eng 10:132–154

Huang W, Li S (1998) A two-stage hybrid flowshop with uniform machines and setup times. Math Comput Model 27:27–45

Jenabi M, Ghomi SMTF, Torabi SA, Karimi B (2007) Two hybrid meta-heuristics for the finite horizon ELSP in flexible flow lines with unrelated parallel machines. Appl Math Comput 186:230–245

Jungwattanakit J, Reodecha M, Chaovalitwongse P, Werner F (2008) Algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria. Int J Adv Manuf Technol 37:354–370

Kianfar K, Ghomi SMTF, Jadid AO (2012) Study of stochastic sequence-dependent flexible flow shop via developing a dispatching rule and a hybrid GA. Eng Appl Artif Intell 25:494–506

Lee TS, Loong YT (2019) A review of scheduling problem and resolution methods in flexible flow shop. Int J Ind Eng Comput 10:67–88 . https://doi.org/10.5267/j.ijiec.2018.4.001

Li D, Meng X, Liang Q, Zhao J (2015) A heuristic-search genetic algorithm for multi-stage hybrid flow shop scheduling with single processing machines and batch processing machines. J Intell Manuf 26:873–890

Li S (1997) A hybrid two-stage flowshop with part family, batch production, major and minor set-ups. Eur J Oper Res 102:142–156

Low C (2005) Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines. Comput Oper Res 32:2013–2025

Maleki-Darounkolaei A, Modiri M, Tavakkoli-Moghaddam R, Seyyedi I (2012) A three-stage assembly flow shop scheduling problem with blocking and sequence-dependent set up times. J Ind Eng Int 8:26 . https://doi.org/10.1186/2251-712X-8-26

Meng L, Zhang C, Shao X, Ren Y, Ren C (2019) Mathematical modelling and optimisation of energy-conscious hybrid flow shop scheduling problem with unrelated parallel machines. Int J Prod Res 57:1119–1145

Raissi S, Rooeinfar R, Ghezavati VR (2019) Three Hybrid Metaheuristic Algorithms for Stochastic Flexible Flow Shop Scheduling Problem with Preventive Maintenance and Budget Constraint. J Optim Ind Eng 12:131–147

Ruiz R, Vázquez-Rodríguez JA (2010) The hybrid flow shop scheduling problem. Eur J Oper Res 205:1–18

Sawik T (2006) Hierarchical approach to production scheduling in make-to-order assembly. Int J Prod Res 44:801–830

Schulz S, Buscher U, Shen L (2020) Multi-objective hybrid flow shop scheduling with variable discrete production speed levels and time-of-use energy prices. J Bus Econ 1–29

Shahvari O, Logendran R (2017) An enhanced tabu search algorithm to minimize a bi-criteria objective in batching and scheduling problems on unrelated-parallel machines with desired lower bounds on batch sizes. Comput Oper Res 77:154–176

Ünal AT, Ağralı S, Taşkın ZC (2019) A strong integer programming formulation for hybrid flowshop scheduling. J Oper Res Soc 1–11

Wang H (2005) Flexible flow shop scheduling: optimum, heuristics and artificial intelligence solutions. Expert Syst 22:78–85

Wang I-L, Wang Y-C, Chen C-W (2013) Scheduling unrelated parallel machines in semiconductor manufacturing by problem reduction and local search heuristics. Flex Serv Manuf J 25:343–366.

Hosseini, S., Hassani, A., Behroozi, F. (2021). Minimizing the operational costs in a flexible flow shop scheduling problem with unrelated parallel machines.
Journal of Optimization in Industrial Engineering, 14(1), 169-184.

http://www.qjie.ir/article_679323.html
DOI: 10.22094/joie.2021.1890852.1718