Computer Systems Science & Engineering
DOI: 10.32604/csse.2023.022716
Article

Tech Science Press

Hybrid Flow Shop with Setup Times Scheduling Problem

Mahdi Jemmali[1,2,3,*] and Lotfi Hidri[4]

[1]Department of Computer Science and Information, College of Science at Zulfi, Majmaah University, Majmaah, 11952, Saudi Arabia
[2]Mars Laboratory, University of Sousse, Sousse, Tunisia
[3]Department of Computer Science, Higher Institute of Computer Science and Mathematics of Monastir, Monastir University, Monastir, 5000, Tunisia
[4]Industrial Engineering Department, College of Engineering, King Saud University, Riyadh, 11421, Saudi Arabia
*Corresponding Author: Mahdi Jemmali. Email: m.jemmali@mu.edu.sa
Received: 16 August 2021; Accepted: 17 September 2021

Abstract: The two-stage hybrid flow shop problem under setup times is addressed in this paper. This problem is NP-Hard. on the other hand, the studied problem is modeling different real-life applications especially in manufacturing and high performance-computing. Tackling this kind of problem requires the development of adapted algorithms. In this context, a metaheuristic using the genetic algorithm and three heuristics are proposed in this paper. These approximate solutions are using the optimal solution of the parallel machines under release and delivery times. Indeed, these solutions are iterative procedures focusing each time on a particular stage where a parallel machines problem is called to be solved. The general solution is then a concatenation of all the solutions in each stage. In addition, three lower bounds based on the relaxation method are provided. These lower bounds present a means to evaluate the efficiency of the developed algorithms throughout the measurement of the relative gap. An experimental result is discussed to evaluate the performance of the developed algorithms. In total, 8960 instances are implemented and tested to show the results given by the proposed lower bounds and heuristics. Several indicators are given to compare between algorithms. The results illustrated in this paper show the performance of the developed algorithms in terms of gap and running time.

Keywords: Hybrid flow shop; genetic algorithm; setup times; heuristics; lower bound

1 Introduction

The hybrid flow shop problems (HFSP) are commonly utilized in several sectors like the manufacturing area. Indeed, HFSP is modeling several production systems in different industries such as paper, electronics, cars, chemicals, and semiconductors [1–4].

The two-stage HFSP is a particular case, which is well studied in the literature. This is due to its practical applications, and its theoretical challenging aspect. Plenty of results are provided to solve this problem using heuristics, meta-heuristics, and exact methods. Authors in [5] proposed an efficient exact method to solve the two-stage HFSP. The latter research doesn't achieve the setup times constraint and the time provided to solve

the problem is remarkable. The Two-stage HFSP with transportation times is addressed in [6]. In the latter work, several heuristics and exact methods are developed. In the experimental results, it is clear the luck of the big scale class to show the performance of the developed method when the number of jobs is real big. The two-stage HFSP with availability constraint is addressed in [7], and an exact algorithm is proposed. In the latter paper, there is only one machine in stage 1. In addition, only one interval of maintenance for each machine. Authors in [8] solved the problem using a genetic algorithm (GA). Several constructive heuristics are developed in the latter work to apply the genetic algorithm seeking the enhancement of the results. The luck of the development of lower bounds to be used as a measurement of the real performance of the heuristics is remarkable for the latter work.

In [9], the authors proposed an exact solution for the hybrid flow shop under sequence-dependent setup time for stage one. An upper bound utilized Hungarian method is proposed. In addition, the authors proposed three heuristics. The experimental results need to be performed when the author gives more attention to the big scale instances.

The problem with batch processing machines is studied in [10], where a decomposition approach with variable neighborhood search (VNS) is developed. Unequal ready times for jobs are considered a constraint for the latter problem. This problem with parallel dedicated machines is addressed in [11], and several heuristics are proposed. The objective was the minimization of the total completion time. Besides, the number of machines is limited to one for stage 1 and two for stage 2. Only two heuristics are developed in the later work.

In [12] authors proposed an approximate solution based on a Branch and Bound exact procedure to tackle the problem of two-stage hybrid flow with dedicated machines. Besides, the authors proposed four constructive heuristics to solve the problem and three dominance priorities. In the experimental results, the instances are generated with a maximum of 100 jobs.

In [13] a new application of the problem of two-stage HFSP is introduced and a heuristic with a local search method is developed and an ant colony optimization algorithm is developed. The bi objectives in the latter search are the total energy consumption and the makespan.

Authors in [14] addressed the problem and proposed a mutant firefly algorithm. Two objectives functions are proposed in the latter work. One of the objectives is the simultaneous ratio for the coming of several parts of products at the stage of assembly. The second objective is the on-time delivery ratio related to the products delivery assignment.

In [15] authors investigated the hybrid flow shop problem (DHFSP) under sequence-dependent setup times. Three sub-problems are involved. The first sub-problem is the allocation of factories for each job. The next sub-problem is to find the job sequence for every factory. The third sub-problem is the allocation of each job at each stage. To test the performance of the proposed solutions authors generated 780 benchmarks in total. In the same context authors in [16], a novel simulated-annealing (NSA) algorithm was proposed to find a reasonable manufacturing assignment in a good running time. Setup times are needed when switching between jobs. Considering the setup times while preparing the schedules of jobs on the machines allows having accurate production planning. To minimize the gap between theoretical points and real-life situations, the setup times are not neglected in this study. The two-stage HFSP with setup times is treated in literature [17–20].

In this study, the two-stage hybrid flow shop problem is addressed. Novel heuristics are proposed. These heuristics are based on the exact solution of the problem of the parallel machine under release and delivery times constraints. In addition, a lower bound allowing the assessment of the performance of heuristics are proposed. The studied problem is an NP-hard problem. It is very important to search for heuristics that give approximate solutions to the problem with acceptable time. In addition, the scale of the problem can impact the performance of the heuristics on time and gap. In this paper, a total of 8960 instances are

generated and tested to prove the efficiency of the proposed lower bounds and heuristics. In the literature review, there is a lack of treatment of the big scale of instances to solve the problem. Indeed, the number of jobs reaching 200. The proposed heuristics give a solution easily for the big scale of instances with a satisfying time and gap.

The rest of this paper is structured as follows. Problem presentation is presented in Section 2. The lower bounds are developed and displayed in Section 3. The detailed heuristics are given in Section 4. An experimental study and discussion of results are reported in Section 5. Finally, the summarization of the studied work is presented in the conclusion with future lines.

2 Problem Definition and Proprieties

In this section, the presentation of the problem and its relevant proprieties are presented. In addition, the problem of the parallel machine under release and delivery times constraints as well as some of its characteristics are recalled. This is because of the important role of the latter problem in providing algorithms for the current studied problem. The two-stage hybrid flow shop problem with setup times is stated as follows. We are given a shop composed of two stages C_1 and C_2 in series, containing m_1 and m_2 identical parallel machines, respectively. A set $J = \{1, 2, \ldots, n\}$ of n jobs have to be executed on the machines of the two stages in the following way. A machine of C_1 is prepared during a setup time $s_{1,j}$ to be ready for processing a job j during $p_{1,j}$. After completion the stage 1, an available machine in the second stage C_2 is prepared during $s_{2,j}$ to process the job j. All machines and jobs are ready for executing at the first time 0. Preemption is not permitted, and all job characteristics $s_{1,j}$, $s_{2,j}$, $p_{1,j}$, and $p_{2,j}$ are deterministic and integral. The objective of the optimization problem is the minimization of the makespan. According to the three fields notation [21] the studied problem is denoted by $Fm|s_{1,j}, s_{2,j}|C_{max}$.

Example 1:

Let $m_1 = 2$ and $m_2 = 2$. The setup times and processing times for the two stages illustrates in Tab. 1. Fig. 1 presents a feasible solution for example 1, where the makespan is $C_{max} = 142$.

Table 1: Setup time and processing time values of Example 1

j	1	2	3	4	5	6	7	8	9	10
$s_{1,j}$	2	8	15	1	10	5	19	19	3	5
$p_{1,j}$	6	6	2	8	2	12	16	3	8	17
$s_{2,j}$	12	5	3	14	13	3	2	17	19	16
$p_{2,j}$	8	7	12	19	10	13	8	20	16	15

It is remarkable to note that the setup times in stage 1 are connected directly to the processing times, while in stage 2, the setup time might be separated from the processing time, as for job 8. Indeed, for this job $c_{1,8} = 22$ which is a release date in the second stage and the setup time is $s_{2,8} = 17 < 22$.

3 Lower Bounds

In this section, we present three new lower bounds based on the linear relaxation technique. The first one is obtained after applying the relaxation of the second stage using the problem of the parallel machine. The second lower bound is obtained after applying the relaxation of the first stage using the problem of the parallel machine. The third lower bound is an enhancement of the first one.

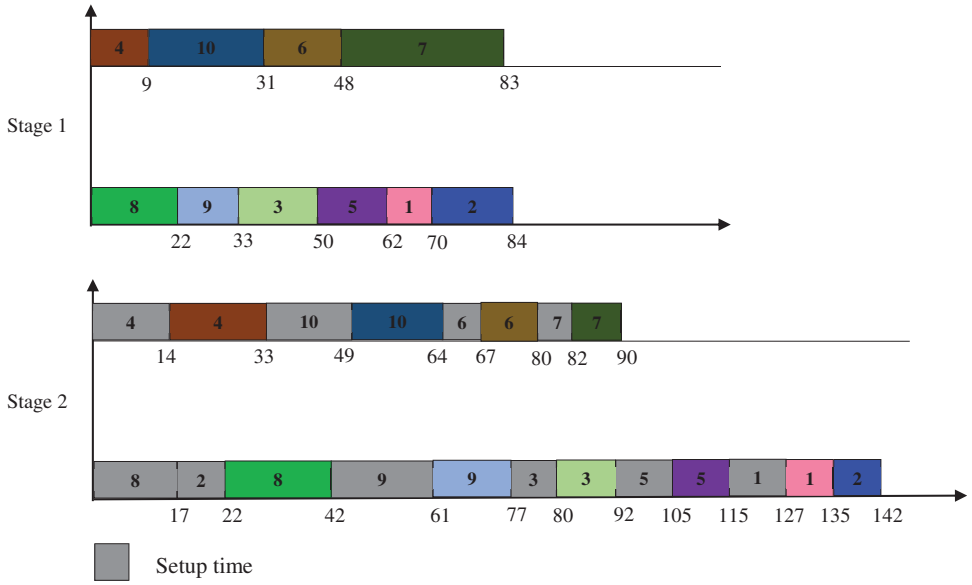

Figure 1: Feasible solution for example 1 $p_{2,j}$

3.1 Relaxed Second Stage Based Parallel Machine Problem (LB_1)

The capacity of the second stage is relaxed and the number of machines m_2 is supposed to be not finite. In this case, an ending processing job in stage 1 has no time to wait for processing in the second stage. Consequently, the obtained scheduling problem in stage 1 is a parallel machines problem under release date r_j and delivery time q_j denoted $P_m|r_j,\ q_j|C_{max}$ where:

- $r_j = s_{1,j}\ \forall j \in J.$
- $p_j = p_{1,j}\ \forall j \in J.$
- $q_j = p_{2,j}\ \forall j \in J.$
- $m = m_1.$

Lemma: Any lower bound of the latter defined problem, is a valid lower bound for $Fm|s_{1,j},\ s_{2,j}|C_{max}$.

Proof: $P_m|r_j,\ q_j|C_{max}$ is obtained after a relaxation of the current problem. Therefore, the optimal values C^*_{max} is a lower bound of the studied problem. Clearly, if LB is a lower bound of $P_m|r_j,\ q_j|C_{max}$ then it is also a valid lower bound for $Fm|s_{1,j},\ s_{2,j}|C_{max}$.

Remark: The Branch & Bound (B&B) algorithm for $P_m|r_j,\ q_j|C_{max}$ presented in [22] is used to obtained the optimal solution. This exact solution is a valid lower bound for the studied problem. Since the $P_m|r_j,\ q_j|C_{max}$ is NP-Hard, the B&B might fail to obtain the exact solution within a limit time LT (In this study the limit time LT is set to 100 s), in this situation the best returned lower bound value while calling the B&B is a valid lower bound for $Fm|s_{1,j},\ s_{2,j}|C_{max}$. This remark is valid for the subsequent other lower bounds. The obtained lower bound in this subsection is denoted LB_1.

3.2 Relaxed First Stage Based Parallel Machine Problem (LB_2)

The capacity of the first stage is relaxed and the number of machines m_1 is supposed to be not finite. In this case, for any job a machine is set up at zero time and processes without delay in the first stage. Consequently, the obtained scheduling problem in the second stage is a parallel machine under r_j and q_j constraints denoted by $P_m|r_j,\ q_j|C_{max}$ where:

- $r_j = \ \max(s_{1,j} + p_{1,j},\ s_{2,j})\ \forall j \in J.$

- $p_j = p_{2,j}\ \forall j \in J$.
- $q_j = 0 \forall j \in J$.
- $m = m_2$.

Following the same steps as for LB_1, the Branch & Bound (B&B) algorithm of $P_m|r_j,\ q_j|C_{max}$ presented in [22] is used to obtain the optimal solution, and therefore the second lower bound denoted LB_2.

3.3 Enhanced Relaxed Second Stage Based Parallel Machine Problem (LB_3)

The capacity of the second stage is relaxed and the number of machines m_2 is supposed to be infinite. In this case, an ending processing job in stage 1 has no time to wait for processing in the second stage. Consequently, the obtained scheduling problem in stage 1 is parallel machines problem under r_j and q_j constraints denoted by $P_m|r_j,\ q_j|C_{max}$ where:

- $r_j = 0\ \forall j \in J$.
- $p_j = s_{1,j} + p_{1,j}\ \forall j \in J$.
- $q_j = p_{2,j}\ \forall j \in J$.
- $m = m_1$.

In the same way, and applying the B&B algorithm a third lower bound denoted LB_3 is obtained.

Remark: LB_3 dominates LB_1.

Proof: Clearly, an optimal solution of problem (P1) $P_m|r_j,\ q_j|C_{max}$ with $r_j = 0, p_j = s_{1,j} + p_{1,j}$, and $q_j = p_{2,j}$ is a feasible solution for the problem (P2) $P_m|r_j,\ q_j|C_{max}$ with $r_j = s_{1,j}$, $p_j = p_{1,j}$, and $q_j = p_{2,j}$. Therefore, any lower LB_1bound for problem (P2) is dominated by LB_3.

Example 2:

In this example, we give the schedule applying LB_3 with the same instance presented in Example 1. Now, we solve exactly the $P_m|r_j,\ q_j|C_{max}$ with r_j, p_j, and q_j values presented in Tab. 2.

Table 2: The r_j, p_j, and q_j values for LB_3

j	1	2	3	4	5	6	7	8	9	10
r_j	0	0	0	0	0	0	0	0	0	0
p_j	8	14	17	9	12	17	35	22	11	22
q_j	8	7	12	19	10	13	8	20	16	15

The optimal solution of $P_m|r_j,\ q_j|C_{max}$ for the first stage assign on machine 1 the jobs {4, 10, 6, 7} and on machine 2 the jobs { 8, 9, 3, 5, 1, 2}. The optimal solution of $P_m|r_j,\ q_j|C_{max}$ constitute the lower bound of the studied problem and illustrated in Fig. 2. Fig. 2 shows that the value of LB_3 is equal to 91.

4 Heuristics

In this section, a metaheuristic and three heuristics are proposed. The developed metaheuristic is implemented using the genetic algorithm (GA). The three other heuristics are based on solving a parallel machines problem under release and delivery time constraints.

4.1 Genetic Algorithm (H_1)

John Holland in [23] presented for the first time the genetic algorithms (GAs). The GAs are inspired by natural evolution. Indeed, the main fundamentals of natural evolution in a given environment are memetic.

During this process, the chromosomes are the main actors in which all the messages and mutations are performed. The different components are:

1. Chromosome Representation: in this study, a chromosome represents a feasible solution. A chromosome is constituted by a permutation of jobs. The jobs in this permutation are scheduled on the machines of stage 1 according to the most available machine rule. That is, the first remaining job in the permutation is allocated to the most available machine. Once all the jobs are scheduled to stage 1, the jobs are scheduled on the second stage according to the non-decreasing order of their completion time in the first stage.
2. Initial population: All the individuals in the initial population are arbitrarily generated, except one solution which is generated according to the LPT rule in the first stage. The jobs are sorted according to the non-increasing order of their processing time in the first stage. Once these jobs are assigned to stage 1 according to the LPT rule, they are scheduled in the second stage according to the increasing order of their completion time in the first stage.
3. Selection Operator: During the selection phase an operator acts to pick the most suitable solutions in order to accelerate the convergence toward the global optimum. Several operators are utilized and the most used one is the roulette wheel selection rule, which is an adopted selection operator in this study.
4. Reproduction: In this step, an individual passes to the next generation without modification, it is a cloning phase. The main objective of the reproduction phase is to preserve the individuals with high fitness from the actual generation to the next one.
5. The Operator of Crossover: Enriching the diversity of the population is the main objective of the crossover operation. Indeed, in this step, the structure of the individuals (chromosomes) is manipulated. Conventionally, the crossover operator considers two parents and generates two children. To guarantee the innovation (even partially) of the children a crossbreeding is performed. The basic idea is that two successful parents will yield better children. The crossover rate $p_c (p_c \in [0,\ 1])$ constitutes the proportion of parents on which a crossover operator will act. The position-based crossover operator (POX) is adopted in this strategy.
6. Mutation: The main goal of the mutation is to create a small randomness element in the population of individuals. The mutation operator could increase or decrease the space of the possible feasible solutions, in addition to the variability of the individuals. p_m determines the probability of mutating each element (gene) of representation. In this work, the adopted p_m is to mutate a percentage of all genes.
7. Replacement Strategies: In this step, individuals are withdrawn following a given selection strategy. Indeed, the best individuals from parents and offspring are kept. This approach allows a rapid convergence and some time to a moderate solution. In order to avoid such a situation, sometimes, bad (fitness) individuals are selected. Those replacement strategies can be deterministic or stochastic.

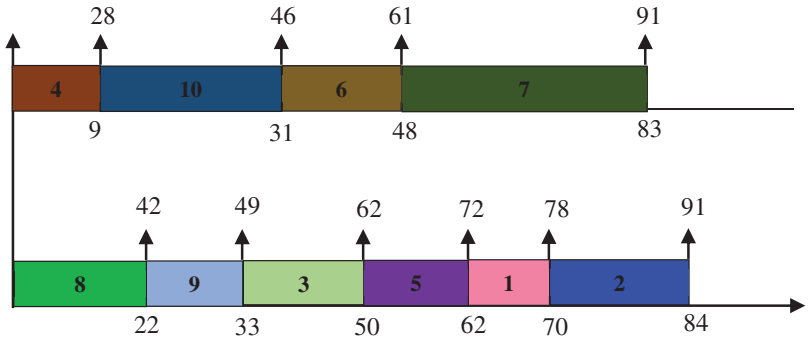

Figure 2: Schedule applying LB_3

In practice, the tuple (Population, crossover rate, probability of mutating) considered in the genetic algorithm are chosen as following: (200;0.9;0.9), (200;0.8;0.9), (200;0.9;0.8), (200;0.7;0.9), (200;0.8;0.8), (200;0.6;0.9), (200;0.7;0.8), (200;0.6; 0.8), (200;0.5;0.9), and (200;0.4;0.9).

4.2 Heuristic H_2

The heuristic H_2 is based on solving optimally a sequence of two parallel machines under release and delivery time constraints. First, a parallel machines problem $P_m|r_j,\ q_j|C_{max}$ is solved in the first stage. The parameters of this parallel machine are:

- $r_j = 0\ \forall j \in J$.
- $p_j = s_{1,j} + p_{1,j}\ \forall j \in J$.
- $q_j = p_{2,j}\ \forall j \in J$.
- $m = m_1$.

Once the above scheduling problem ($P_m|r_j,\ q_j|C_{max}$) is solved and an optimal solution is obtained, the completion time of $j \in J$ is denoted $c_{1,j}$. Second, in stage 2, a parallel machines problem $P_m|r_j,\ q_j|C_{max}$ is optimally solved. The characteristics of the latter problem are:

- $r_j = \max(c_{1,j},\ s_{2,j})\ \forall j \in J$.
- $p_j = p_{2,j}\ \forall j \in J$.
- $q_j = 0\ \forall j \in J$.
- $m = m_2$.

In the third step, we remark that the obtained solution is not necessarily a feasible one since it may occur that a job has no setup time in the second stage. To overcome this drawback, an adjusted solution is introduced. The adjustment is as follows. If the first scheduled job in a machine of the second stage is the job j, then we have to check two cases:

1) if $s_{2,j} \geq c_{1,j}$ then the completion time of j is $c_{2,j} = s_{2,j} + p_{2,j}$.
2) if $s_{2,j} < c_{1,j}$ then the completion time of j is $c_{2,j} = c_{1,j} + p_{2,j}$.

If a job j is preceded by a job i in the same machine then:

1) if $c_{2,i} + s_{2,j} \geq c_{1,j}$ then the completion time of j is $c_{2,j} = c_{2,i} + s_{2,j} + p_{2,j}$.
2) if $c_{2,i} + s_{2,j} < c_{1,j}$ then the completion time of j is $c_{2,j} = c_{1,j} + p_{2,j}$.

The concatenation of the two feasible solutions respectively in the first and second stages provides a feasible solution for t $Fm|s_{1,j},\ s_{2,j}|C_{max}$.

Example 3:

We adopt the same instance represented in Tab. 1 in Example 1. In the first stage, we apply $P_m|r_j,\ q_j|C_{max}$, and the exact solution is the same that described in Fig. 2. From this optimal solution, we determine all $c_{1,j}$ values and we represent r_j, q_j and p_j to solve the second optimal problem applied on the second stage in Tab. 3.

Table 3: Determination of $c_{1,j}$ and definition of r_j, q_j and p_j

j	1	2	3	4	5	6	7	8	9	10
$c_{1,j}$	70	84	50	9	62	48	83	22	33	31
r_j	70	84	50	14	62	48	83	22	33	31
p_j	8	7	12	19	10	13	8	20	16	15
q_j	0	0	0	0	0	0	0	0	0	0

The optimal solution of $P_m|r_j,\ q_j|C_{max}$ for the second stage with instance given in Tab. 3 assign jobs {4, 10, 6, 5, 7} on machine 1 and {8, 9, 3, 1, 2} on machine 2. Fig. 3 represents the related schedule. Fig. 3 displays a feasible schedule in the second stage as a result of the heuristic H_2.

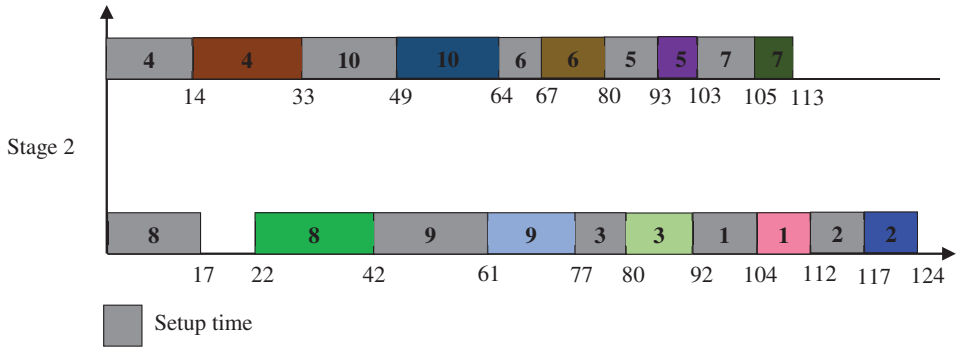

Figure 3: Feasible schedule for H_2

4.3 Heuristic H_3

The heuristic H_3 is based on solving optimally a sequence of two parallel machines under release date and delivery time. First, a parallel machine scheduling problem $P_m|r_j,\ q_j|C_{max}$ is solved in the first stage. The parameters of this parallel machine are:

- $r_j = 0\ \forall j \in J$.
- $p_j = s_{1,j} + p_{1,j}\ \forall j \in J$.
- $q_j = p_{2,j}\ \forall j \in J$.
- $m = m_1$.

Once the above scheduling problem ($P_m|r_j,\ q_j|C_{max}$) is solved and an exact solution is returned, the completion time of a job $j \in J$ is denoted $c_{1,j}$. Second, in the second stage, a parallel machine scheduling problem $P_m|r_j,\ q_j|C_{max}$ is optimally solved. The characteristics of the latter problem are:

- $r_j = c_{1,j}\ \forall j \in J$.
- $p_j = p_{2,j}\ \forall j \in J$.
- $q_j = 0\ \forall j \in J$.
- $m = m_2$.

The obtained solution in the second stage does not include setup times. In this step, the setup times $s_{2,j}$ are embedded before the processing of jobs. The concatenation of the two feasible solutions respectively in the first and second stages provides a feasible solution for the studied problem.

4.4 Heuristic H_4

The heuristic H_4 is based on solving optimally a sequence of two parallel machines under release and delivery time constraints. First, a parallel machine scheduling problem $P_m|r_j,\ q_j|C_{max}$ is solved in the first stage. The parameters of this parallel machine are:

- $r_j = 0\ \forall j \in J$.
- $p_j = s_{1,j} + p_{1,j}\ \forall j \in J$.
- $q_j = p_{2,j}\ \forall j \in J$.
- $m = m_1$.

Once the above scheduling problem $(P_m|r_j,\ q_j|C_{max})$ is solved and an optimal solution is obtained, the completion time of a job $j \in J$ is denoted $c_{1,j}$. Second, in the second stage, a parallel machine scheduling problem $P_m|r_j,\ q_j|C_{max}$ is optimally solved. The characteristics of the latter problem are:

- $r_j = c_{1,\,j}\ \forall j \in J$.
- $p_j = s_{2,\,j} + p_{2,\,j}\ \forall j \in J$.
- $q_j = 0\ \forall j \in J$.
- $m = m_2$.

In the expression of the processing times $p_j = s_{2,j} + p_{2,j}$ we consider the setup time and the processing time as one entity. This will give a feasible solution in the second stage which may largely exceed an optimal solution. To readjust the obtained solution, we split setup time from processing time, and we apply the following recursive rule in the jobs scheduled to the same machine. $c_{2,j} = \max(c_{1,j},\ c_{2,j-1} + s_{2,j})$ where $c_{2,j-1}$ is the completion time of the predecessor of job j in the same machine. The concatenation of the two feasible solutions respectively in the first and second stages provides a feasible solution for $Fm|s_{1,\,j},\ s_{2,\,j}|C_{max}$.

5 Computational Results

This section focuses on the illustration and the discussion of the obtained results. All proposed algorithms in this paper were coded in C++. All coded programs were executed by a workstation that has the following characteristics: Intel(R) Xeon(R) CPU E5-2687W v4 @ 3.00 GHz and 64 GB RAM.

5.1 Test Problems

The generation of the setup times and processing times are applied according to the uniform distribution denoted by $U[st, fi]$. In this paper the choice of $st = 1$ and $fi = \{20, 40\}$. The class of instance is depending on the choice of the generation of $s_{1,\,j}, p_{1,\,j}, s_{2,\,j}$ and $p_{2,\,j}$. So, we generate $s_{1,\,j}$ in [1, 20] or [1, 40], and same thing for $p_{1,j}$, $s_{2,j}$ and $p_{2,j}$. We denoted by C1, C2, C3, and C4 the choice of fi to generate $s_{1,\,j}$, $p_{1,\,j}$, $s_{2,\,j}$ and $p_{2,\,j}$, respectively. The value of each variable C1, C2, C3, and C4 represent the manner that the setup times and processing times are generated. For example, if $(C1 = 20,\ C2 = 20,\ C3 = 40,\ C4 = 20)$ meaning that we generate $s_{1,j}$ in [1, 20], $p_{1,j}$ in [1, 20], $s_{2,j}$ in [1, 40] and $p_{2,j}$ in [1, 20]. The number of permutations to generate classes is $4^2 = 16$. The number of machines for each stage is chosen in {2, 3, 4, 5}. The number of permutation to constitute a pair (m_1, m_2) is 16.

Now, for the number of jobs, we adopt $n = \{10, 20, 30, 50, 100, 150, 200\}$. For each type of class (C1, C2, C3, C4), each (m_1, m_2) and each n we generate 5 instances. Finally, the total number of instances is $16 \times 16 \times 7 \times 5 = 8960$.

5.2 Performance Measures

To assess the performance of the heuristics and lower bounds, the following performance measures are used:

- *Time*: average CPU time (s)
- $Gap = \frac{UB^* - LB^*}{LB^*}$, where UB^* the best heuristic value and LB^* the best obtained lower bound value.
- $Gap_L = \frac{LB^* - L}{LB^*}$, where L is the studied lower bound value.
- $Gap_U = \frac{U - UB^*}{UB^*}$, where U is the studied heuristic value.

5.3 Results and Discussion

Tab. 4 represents the variation of the *Gap* according to n. As shown in this table when the number of jobs increases, the *Gap* increase. The maximum *Gap* of 0.27 is obtained when $n = 200$.

Table 4: *Gap* according to n

n	*Gap*
10	0.19
20	0.24
30	0.25
50	0.26
100	0.26
150	0.26
200	0.27

Tab. 5 shows the variation of *Gap* according to (m_1) on stage 1 and to m_2 on stage 2.

Table 5: *Gap* according to m_1 and m_2

m_1	*Gap*	m_2	*Gap*
2	0.05	2	0.54
3	0.17	3	0.26
4	0.31	4	0.12
5	0.45	5	0.06

Tab. 6 presents the *Gap* values according to C1, C2, C3, and C4. The minimum gap value of 0.13 is obtained when C3= 20.

Table 6: *Gap* according to C1, C2, C3, and C4

	C1	C2	C3	C4
20	0.33	0.33	0.13	0.21
40	0.16	0.16	0.31	0.29

Now, we present the results obtained for the developed lower bounds. Tab. 7 shows that the best lower bound is LB_3. Indeed, the percentage of this is lower when the number of instances that LB_3 is the best one is 84.6% in an average time 16.37 and an average gap of 0.04.

Tab. 8, represents the variation of Gap_L according to n. For LB_3 the maximum gap is obtained when $n = \{10, 20, 30\}$.

Tab. 9 presents the Gap_L and *Time* variation according to m_1 and m_2 for all lower bounds. The most time-consuming lower bound is LB_3 when $m_1 = 5$. In this condition, the running time is reached to 34.56 s.

Table 7: Overall Gap_L

	LB_1	LB_2	LB_3
Gap_L	0.47	0.40	0.04
Time	4.26	0.63	16.37
Perc	0.7%	16.3%	84.6%

Table 8: Gap_L and *Time* according to n

n	LB_1		LB_2		LB_3	
	Gap_L	*Time*	Gap_L	*Time*	Gap_L	*Time*
10	0.28	0.05	0.24	0.03	0.04	0.86
20	0.46	3.50	0.38	2.39	0.04	21.51
30	0.50	6.38	0.41	0.88	0.04	20.36
50	0.51	4.00	0.44	0.32	0.03	17.07
100	0.51	4.08	0.45	0.17	0.03	17.26
150	0.52	5.17	0.45	0.27	0.03	18.86
200	0.52	6.63	0.45	0.38	0.03	18.89

Table 9: Gap_L and *Time* according to m_1 and m_2 for all lower bounds

	LB_1		LB_2		LB_3	
	Gap_L	Time	Gap_L	Time	Gap_L	Time
m_1						
2	0.49	0.73	0.62	0.61	0.00	1.88
3	0.47	2.29	0.45	0.65	0.01	8.02
4	0.46	5.59	0.32	0.66	0.04	21.03
5	0.46	8.42	0.23	0.62	0.08	34.56
m_2						
2	0.51	4.17	0.22	0.15	0.10	15.61
3	0.47	4.41	0.37	0.25	0.03	16.55
4	0.45	4.42	0.48	0.67	0.01	17.14
5	0.45	4.03	0.55	1.46	0.00	16.18

The variation of Gap_L and *Time* according to C1, C2, C3, and C4 for all lower bounds is illustrated in Tab. 10.

The assessment of the proposed heuristics is presented in Tabs. 11 and 12.

The behavior of Gap_U and *Time* according to m_1 and m_2 for all heuristics is presented in Tab. 13.

Table 10: Gap_L and *Time* according to C1, C2, C3, and C4 for all lower bounds

		C1		C2		C3		C4	
		Gap_L	Time	Gap_L	Time	Gap_L	Time	Gap_L	Time
20	LB1	0.41	4.63	0.55	1.49	0.48	4.67	0.47	4.49
	LB2	0.34	0.77	0.34	0.71	0.41	0.75	0.54	0.20
	LB3	0.05	11.89	0.05	11.60	0.03	17.22	0.01	16.69
40	LB1	0.53	3.98	0.40	7.01	0.48	4.18	0.49	4.61
	LB2	0.47	0.50	0.47	0.56	0.41	0.49	0.28	1.01
	LB3	0.02	21.08	0.02	21.15	0.03	16.09	0.06	16.90

Table 11: Overall results for all heuristics

	H_1	H_2	H_3	H_4
Gap_U	0.00	0.06	0.06	0.03
Time	15.63	16.55	14.59	16.56
Perc	93.2%	30.1%	30.3%	30.3%

Table 12: Gap_U and *Time* variation according to n for all heuristics

n	H_1		H_2		H_3		H_4	
	Gap_U	*Time*	Gap_U	*Time*	Gap_U	*Time*	Gap_U	*Time*
10	0.00	0.16	0.12	0.94	0.13	0.59	0.08	0.65
20	0.00	0.42	0.09	22.16	0.09	17.58	0.05	20.30
30	0.00	0.86	0.07	21.17	0.07	16.58	0.04	19.24
50	0.00	2.37	0.05	17.19	0.05	16.06	0.02	18.89
100	0.00	11.17	0.03	17.41	0.03	16.32	0.01	18.56
150	0.00	30.49	0.03	19.11	0.03	18.21	0.01	19.58
200	0.00	63.93	0.02	18.54	0.02	16.79	0.01	19.68

Tab. 14 presents the variation of Gap_U and *Time* for C1, C2, C3, and C4 for all heuristics. The maximum reaching time is 21.43 s. This running time is obtained for heuristic H_1 when C4 = 20. However, the minimum reached time 10.18 s is obtained for H_3 when C2 = 20.

Table 13: Gap_u and *Time* variation according to m_1 and m_2 for all heuristics

	H_1		H_2		H_3		H_4	
	Gap_U	*Time*	Gap_U	*Time*	Gap_U	*Time*	Gap_U	*Time*
m_1								
2	0.00	15.18	0.02	1.95	0.02	1.54	0.01	1.96
3	0.00	15.55	0.05	8.44	0.05	7.02	0.01	7.77
4	0.00	15.77	0.08	21.26	0.08	18.51	0.02	21.11
5	0.00	16.00	0.09	34.94	0.10	31.30	0.02	35.39
m_2								
2	0.00	14.94	0.06	15.93	0.06	13.94	0.03	14.43
3	0.00	15.24	0.07	16.94	0.07	14.86	0.04	16.53
4	0.00	15.83	0.06	17.35	0.06	15.32	0.03	18.13
5	0.00	16.50	0.05	16.37	0.05	14.25	0.03	17.14

Table 14: Gap_U and *Time* variation according to C1, C2, C3, and C4

		C1		C2		C3		C4	
		Gap_U	*Time*	Gap_U	*Time*	Gap_U	*Time*	Gap_U	*Time*
20	H_1	0.00	15.76	0.00	15.50	0.00	18.61	0.00	21.43
	H_2	0.07	12.14	0.07	11.91	0.04	17.86	0.05	16.87
	H_3	0.07	10.40	0.07	10.18	0.04	15.60	0.05	14.96
	H_4	0.03	13.01	0.03	12.46	0.05	14.61	0.03	16.37
40	H_1	0.00	15.75	0.00	15.68	0.00	17.79	0.00	21.41
	H_2	0.05	21.38	0.05	21.40	0.06	16.36	0.06	17.16
	H_3	0.05	18.98	0.05	19.02	0.06	14.47	0.06	15.04
	H_4	0.03	20.35	0.03	20.67	0.05	14.69	0.03	17.53

6 Conclusion

This paper focuses on the two-stage hybrid flow shop problem under setup times. Several lower bounds and heuristics were developed. The exact resolution of the problem of the parallel machine is utilized in this paper to elaborate on lower bounds and heuristics. A genetic heuristic is developed for the metaheuristic. A comparison between lower bounds and heuristics is detailed to show the proposed algorithms' performance in terms of gap and time. The genetic algorithm developed in this paper is the best heuristic comparing with the proposed others ones.

In future research work, the elaboration of exact solutions based on B&B algorithms could be considered. Indeed, the proposed heuristics and lower bounds in the paper can be used to elaborate such an exact algorithm. In addition, some other metaheuristics could be explored to detect the most appropriate ones to this scheduling problem.

Acknowledgement: The authors would like to thank the Deanship of Scientific Research at Majmaah University for supporting this work under Project Number No. 1439-19.

Funding Statement: The authors would like to thank the Deanship of Scientific Research at Majmaah University for supporting this work under Project Number No. 1439-19.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] J. Alcaraz and C. Maroto, "A robust genetic algorithm for resource allocation in project scheduling," *Annals of Operations Research*, vol. 102, pp. 83–109, 2001.

[2] L. Shi, G. Guo and X. Song, "Multi-agent based dynamic scheduling optimization of the sustainable hybrid flow shop in a ubiquitous environment," *International Journal of Production Research*, vol. 59, pp. 576–597, 2021.

[3] W. Han, Q. Deng, G. Gong, L. Zhang and Q. Luo, "Multi-objective evolutionary algorithms with heuristic decoding for hybrid flow shop scheduling problem with worker constraint," *Expert Systems with Applications*, vol. 168, pp. 114282, 2021.

[4] W. Yankai, W. Shilong, L. Dong, S. Chunfeng and Y. Bo, "An improved multi-objective whale optimization algorithm for the hybrid flow shop scheduling problem considering device dynamic reconfiguration processes," *Expert Systems with Applications*, vol. 174, pp. 114793, 2021.

[5] M. Haouari, L. Hidri and A. Gharbi, "Optimal scheduling of a two-stage hybrid flow shop," *Mathematical Methods of Operations Research*, vol. 64, pp. 107–124, 2006.

[6] L. Hidri, S. Elkosantini and M. M. Mabkhot, "Exact and heuristic procedures for the two-center hybrid flow shop scheduling problem with transportation times," *IEEE Access*, vol. 6, pp. 21788–21801, 2018.

[7] H. Allaoui and A. Artiba, "Scheduling two-stage hybrid flow shop with availability constraints," *Computers & Operations Research*, vol. 33, pp. 1399–1419, 2006.

[8] S. Wang and M. Liu, "A genetic algorithm for two-stage no-wait hybrid flow shop scheduling problem," *Computers & Operations Research*, vol. 40, pp. 1064–1075, 2013.

[9] S. Wang, X. Wang and L. Yu, "Two-stage no-wait hybrid flow-shop scheduling with sequence-dependent setup times," *International Journal of Systems Science: Operations & Logistics*, vol. 7, pp. 291–307, 2020.

[10] Y. Tan, L. Mönch and J. W. Fowler, "A hybrid scheduling approach for a two-stage flexible flow shop with batch processing machines," *Journal of Scheduling*, vol. 21, pp. 209–226, 2018.

[11] J. Yang, "Minimizing total completion time in two-stage hybrid flow shop with dedicated machines," *Computers & Operations Research*, vol. 38, pp. 1045–1053, 2011.

[12] S. Wang and M. Liu, "A heuristic method for two-stage hybrid flow shop with dedicated machines," *Computers & Operations Research*, vol. 40, pp. 438–450, 2013.

[13] S. Wang, X. Wang, F. Chu and J. Yu, "An energy-efficient two-stage hybrid flow shop scheduling problem in a glass production," *International Journal of Production Research*, vol. 58, pp. 2283–2314, 2020.

[14] B. Fan, W. Yang and Z. Zhang, "Solving the two-stage hybrid flow shop scheduling problem based on mutant firefly algorithm," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, pp. 979–990, 2019.

[15] Y. Li, X. Li, L. Gao, B. Zhang, Q. K. Pan *et al.,* "A discrete artificial bee colony algorithm for distributed hybrid flowshop scheduling problem with sequence-dependent setup times," *International Journal of Production Research*, vol. 59, pp. 3880–3899, 2021.

[16] H. Mirsanei, M. Zandieh, M. J. Moayed and M. R. Khabbazi, "A simulated annealing algorithm approach to hybrid flow shop scheduling with sequence-dependent setup times," *Journal of Intelligent Manufacturing*, vol. 22, pp. 965–978, 2011.

[17] H. T. Lin and C. J. Liao, "A case study in a two-stage hybrid flow shop with setup time and dedicated machines," *International Journal of Production Economics*, vol. 86, pp. 133–143, 2003.

[18] M. Hekmatfar, S. F. Ghomi and B. Karimi, "Two stage reentrant hybrid flow shop with setup times and the criterion of minimizing makespan," *Applied Soft Computing*, vol. 11, pp. 4530–4539, 2011.

[19] G. C. Lee, J. M. Hong and S. H. Choi, "Efficient heuristic algorithm for scheduling two-stage hybrid flowshop with sequence-dependent setup times," *Mathematical Problems in Engineering*, vol. 2015, pp. 1–10, 2015.

[20] S. Wang and M. Liu, "Two-stage hybrid flow shop scheduling with preventive maintenance using multi-objective tabu search method," *International Journal of Production Research*, vol. 52, pp. 1495–1508, 2014.

[21] R. L. Graham, E. L. Lawler, J. K. Lenstra and A. R. Kan, "Optimization and approximation in deterministic sequencing and scheduling: A survey," *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979.

[22] A. Gharbi and M. Haouari, "Minimizing makespan on parallel machines subject to release dates and delivery times," *Journal of Scheduling*, vol. 5, pp. 329–355, 2002.

[23] J. H. Holland, "Genetic algorithms and adaptation," in *Adaptive Control of Ill-Defined Systems*, Springer: Boston, MA, vol. 16, pp. 317–333, 1984.