IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 25, 2018, accepted April 8, 2018, date of publication April 12, 2018, date of current version May 9, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2826069

Exact and Heuristic Procedures for the Two-Center Hybrid Flow Shop Scheduling Problem With Transportation Times

LOTFI HIDRI, SABEUR ELKOSANTINI, AND MOHAMMED M. MABKHOT
Industrial Engineering Department, King Saud University, Riyadh 11421, Saudi Arabia
Correponding author: Lotfi Hidri (lhidri@ksu.edu.sa)

This work was supported by the Deanship of Scientific Research, King Saud University, through research group under Grant RG-1438-056.

ABSTRACT This paper addresses the two centers hybrid flow shop scheduling problem with transportation times. This problem is faced in several real-world applications. The considered objective function to be minimized in this problem is the maximum completion time. This scheduling problem is NP-hard, and most presented research in this scheduling area ignore the transportation times. In order to solve the considered scheduling problem, several lower bounds are developed. In addition, a two phase's heuristic is presented. This heuristic is based on the optimal solution of the parallel machine scheduling problem with release date and delivery time. Furthermore, a branch and bound exact procedure is developed. Finally, extensive numerical experiments are presented in order to asses the performance and the effectiveness of the proposed procedures. Computational results provide evidence that the proposed procedures are very effective in term of producing optimal solutions in a short computational time for large size problems.

INDEX TERMS Hybrid flow shop, transportation times, lower bound, heuristic, branch and bound.

I. INTRODUCTION

The Hybrid Flow Shop (HFS) scheduling problem has been intensively studied during the last three decades [5], [6], [30]. This scheduling problem models many real industrial systems, especially the production processes. Among these production processes, we find the cable manufacturing [2], the electronic industry [3], and textile industry [4], to quote few. The HFS shops are composed of several production centers in series, where each one includes several identical parallel machines. At least one of these centers should contain more than one machine to be considered as a HFS. These machines are available for treating a set of jobs, successively throughout all the centers in the same order (from the first stage until the last one), in other term the flow of the products within the centers is unidirectional. In each stage a job is processed by only one machine without distinction between them. A machine in a stage can handle at the same time at most one job. Interestingly, the HFS scheduling problem generalizes other important and basic ones: the parallel machine, the single machine, and the flow shop. The makespan (the completion time of the last scheduled job) is considered as the objective to be minimized. In this case the HFS is NP-Hard [1], which makes it a challenging problem from the theoretical point of view. In order to solve the HFS problem optimally or approximately, a plenty of procedures are provided. These procedures are heuristics, or metaheuristics or exact algorithms as branch and bound. For a general overview of such procedures the reader is referred to these papers [5], [6].

In order to reduce the gap between practice and theory, additional characteristics or complex constraints for the HFS have to be considered. In this context, the HFS with multiprocessor tasks, where a job requires at the same time more than one machine in a stage, has been addressed in [16]. The authors studied the complexity of the problem and showed its NP-Hardness. A genetic algorithm based feasible solution was also developed. The HFS with setup times occurs when a machine requires some time to be prepared to process a job. This problem is considered in [17] and solved using a genetic algorithm with random key. In [9] studied a HFS problem with sequence dependent setup times and time lags, where the makespan is the objective to be minimized. A Mixed Integer Program (MIP) and an Immune Algorithm (IA) was proposed to provide a near-optimal solution for the latter scheduling problem. The efficiency of the IA heuristic was assessed throughout an extensive experimental study that showed their

2169-3536 © 2018 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

effectiveness. Furthermore, a finished processed job in a stage might be not available to be treated in the next stage immediately, this is occurring for example when a job requires some time to be transported from one stage to the next one. This is the HFS with transportation time (HFST). This kind of problem has been investigated in [18], and the authors developed a Simulated Annealing (SA) based metaheuristic.

In the past, researchers ignored the transportation time between machines/centers when addressing the scheduling problems and this could be justified by two reasons. The first one is the simplicity of the layout of the production system, where the machines/centers are close to each other's. In this case the transportation time is so small compared to the processing time and therefore neglected. The second reason is that when including the transportation time, the studied scheduling problems get more harder and more complex to be solved. The recent technological developments and the customers' needs require more complex production systems and neglecting the transportation time is no longer accepted.

The transportation time is composed of two types. The first type is the job dependent, where the transportation time depends on the job itself, and the second type is the job independent for which the transportation time depends restrictively of the distance separating the machines/centers [19]. Furthermore, the system ensuring the transportation between machines/centers could be a single transporter as the robot, or multi-transporter [20]. In the single transporter, a processed job may wait for the transporter until its returning back in order to be carried to the next machine/stage. In contrast, for the multi-transporter, several transporters (unlimited) are available and a treated job is transported immediately to the next machine/stage. In this work, the job dependent transportation time with unlimited transporters concept in the HFS environment is considered.

In the sequel, a brief literature review of the HFS containing at least the transportation times as an assumption is presented. Naderi *et al.* [19] studied the flexible flow shop problem with setup times and job dependant transportation time, where the total weighted tardiness is the objective to be minimize. They proposed an electromagnetism algorithm for solving the studied problem. The HFS with setup times and transportation time has been studied in [18]. The authors considered the total completion time and total tardiness as objectives to be minimized, respectively. They presented a SA based algorithm to provide a near-optimal solution for the considered problem.

The flexible flow shop scheduling problem, including the release time and the robotic transportation, with makespan as objective function, has been studied in [21]. For this problem, an ant colony optimization and a genetic algorithms has been proposed and analyzed. In [22], the two centers HFS scheduling problem with transportation consideration has been addressed. The first center contains one machine and the second one has two machines. The transportation between centers is done by a single transporter with one job as a capacity. The authors proposed for solving this problem a heuristic which has been proofed to be a fast and efficient one. The two centers HFS problem with transportation consideration and batching is investigated in [23]. The treated jobs in the first stage are transported by a set of vehicles to the batching stage. Each one of these vehicles can carry only one job. Processing a batch involves a processing cost and the minimization of the makespan and the processing cost is the objective. Authors present an algorithm with polynomial complexity for the particular case where only one vehicle is available. They also showed that the problem is NP-Hard for the general case. An algorithm with pseudo-polynomial complexity is developed to solve the studied problem. In [24], the HFS problem with blocking and robotic cells is studied. The integrated assumptions are the multiple part types, machine eligibility, multiple robots, and unrelated parallel machines. A solution based on the SA is proposed for the makespan minimization. The efficiency of the proposed procedure is assessed over an experimental study. The two centers HFS robotic problem is investigated in [25]. The first center is composed of a two dedicated machines and the second center contains only one machine. The transportation between the centers is done by a robot. A MIP procedure is proposed to minimize the maximum completion time. Furthermore, an analyze of the complexity for special cases of the studied problem is performed. These special cases are showed to be solved within a polynomial time. Because of the NP-Hardness of the considered problem, two heuristics are developed.

In this paper, the two centers HFS problem with job dependant transportation time and unlimited transporters is considered. The objective to be minimized is the maximum completion time. The purpose is to develop a family tight lower bounds, a heuristic and an exact procedure. The exact procedure is based on the Branch and Bound algorithm.

Besides its theoretical challenging nature (NP-Hard in the strong sense), the addressed problem (the two-center hybrid flow shop scheduling problem with transportation times) is accurately modeling several real-world problems. Among these real-world problems, we can present the following example. In [31] and [32] a real-world two-center industrial workshop, with transportation time, is presented. In this manufacturing system, the first center is composed of 12 identical molds subject to a heating process. The second center, contains a set of identical press cutting machines. In addition, transportation times between the two centers are considered.

At the best of our knowledge, this is the first time a branch and bound exact procedure, solving large size instances, in moderate CPU time, for the studied problem, is proposed. This is due to the newly developed lower bounds and the efficient proposed dominance rules. In addition, a new family of lower bounds exploiting the special case of two centers is presented. Finally, a two phase heuristic is developed. This heuristic is based on solving iteratively and exactly parallel machines scheduling problems. The efficiency of these procedures (lower bounds and heuristic) is tested by measuring the average relative gap and encouraging results are obtained.

The remainder of this paper is organized as follows. In Section 2 the studied scheduling problem is formally defined, in addition to some interesting properties. Section 3 is intended to the development of a family of tight lower bounds. In section 4, a two phases efficient heuristic is proposed. In Section 5, the branch and bound exact procedure will be detailed. Section 6 is devoted to an intensive experimental study in order to assess the efficiency of the proposed procedures.

II. PROBLEM DEFINITION AND PROPRIETIES

In this section, the two centers hybrid flow shop scheduling problem with job dependant transportation time is introduced. Firstly, a formal definition of the problem is presented. Then, some interesting proprieties are proposed. These proprieties are useful in the development of tight lower and upper bounds.

A. PROBLEM DEFINITION

The tow centers hybrid flow shop scheduling problem with job dependant transportation time could be defined as follows: A set of two production centers C_1 and C_2 containing m_1 and m_2 identical parallel machines, respectively ($\max(m_1, m_2) > 1$), has to process a set $J = \{1, \ldots, n\}$ of n ($n > \max(m_1, m_2)$) jobs as follows. Each job $j \in J$ is processed by only one machine from C_1, once completed, it has to be transported during t_j unites of time until reaching the second stage C_2, where it will be processed by an available machine. If it is not the case and there is no available machine, then the job has to wait in a buffer. The capacity of this buffer is assumed to be infinite. After that, an available machine from C_2 processes job j during p_{2j} units of time. The machines in stage C_1 (respectively in C_2) are denoted $M_{1,1}, \ldots., M_{1,m_1}$ (respectively, $M_{2,1}, \ldots., M_{2,m_2}$). All the jobs are ready to be processed from zero time. Furthermore, all the machines are available to treat the jobs onwards time zero. All the processing times p_{ij} and the transportation times t_j ($i = 1, 2$ and $j \in J$) are deterministic and integer. A machine treats at the same time only one job and a job is processed at the same time by only one machine. The preemption is not allowed while processing the jobs. The objective is to build a feasible schedule that minimizes the makespan $C_{\max}$ or the completion time of the last treated job on C_2. In other terms, if σ is a feasible schedule, and $c_{ij}(\sigma)$ is the completion time of job j in stage i ($i = 1, 2$ and $j \in J$) then the corresponding makespan is $C_{\max}(\sigma) = \max_{j \in J} c_{2j}(\sigma)$. The optimal makespan $C^*_{\max}$ is defined as $C^*_{\max} = \min_{\sigma \in \Gamma} C_{\max}(\sigma)$ where Γ is the set of all the feasible schedules. Following the three-field notation $\alpha|\beta|\gamma$, ([12]), the consider problem is noted $F2\left(P_{m_1}, P_{m_2}\right)|t_j|\, C_{\max}$. More precisely, $F2$ is indicating the flow shop with two centers, and P_{m_1} (respectively, P_{m_2}) stands for m_1 parallel and identical machines in center 1 (respectively, m_2 parallel and identical machines in center 2).

Example 1: Consider the instance with $n = 5$, and $m_1 = m_2 = 2$. The processing times p_{ij} and the transportation times t_j ($i = 1, 2$ and $j \in J$) are presented in Table 1.

TABLE 1. Data of example 1.

j	1	2	3	4	5
p_{1j}	1	2	3	2	3
t_j	3	1	1	2	1
p_{2j}	2	2	1	1	1

A feasible schedule corresponding to the latter instance is presented over Fig. 1.

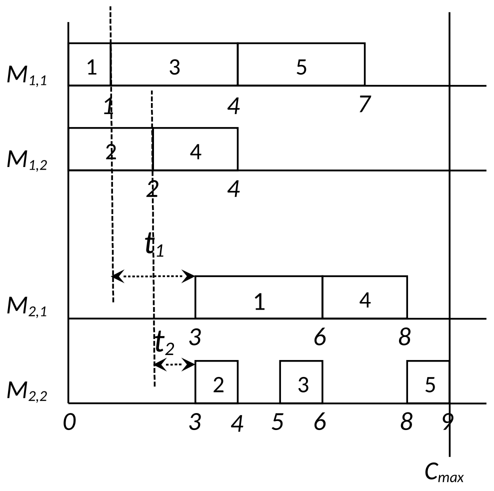

FIGURE 1. Gantt chart of a feasible schedule for example 1.

B. PROPRIETIES

In this section some interesting proprieties are presented, mainly the symmetry of the under consideration scheduling problem.

1) COMPLEXITY

Beside its practical interest, the two centers hybrid flow shop scheduling problem is a challenging one from theoretical point of view. This is due to the following lemma.

Lemma 1: The $F2\left(P_{m_1}, P_{m_2}\right)|t_j|\, C_{\max}$ scheduling problem is strongly NP-Hard

Proof 1: $F2\left(P_{m_1}, P_{m_2}\right)|t_j|\, C_{\max}$ is a generalisation of the two centers hybrid flow shop scheduling problem $F2\left(P_{m_1}, P_{m_2}\right)||\, C_{\max}$, which is strongly NP-Hard ([13]).

2) SYMMETRY

Definition 1: The reverse problem of $F2\left(P_{m_1}, P_{m_2}\right)|t_j|$ $C_{\max}$ is the problem obtained by inverting the roles of the centers. In other terms, starting the scheduling from the second stage C_2 toward the first stage C_1. The scheduling from C_1 to C_2 is said the forward problem and from C_2 to C_1 is said the backward problem. In addition we adopt the following notations. The first and the second centers for the reverse problem are $C^R_1 = C_2$ and $C^R_2 = C_1$, respectively. Furthermore, the processing times in C^R_1 and C^R_2 are $p^R_{1j} = p_{2j}$ and $p^R_{2j} = p_{1j}$, respectively. The inter-stage transportation time(from C^R_1 to C^R_2) is $t^R_j = t_j$ ($j \in J$).

The interest of considering the reverse problem lies in the following results.

Proposition 1: The $F_2\left(P_{m_1}, P_{m_2}\right)\left|t_j\right| C_{\max}$ and its reverse problem are equivalent, i.e. from a schedule S for the forward problem, a schedule S^R for the backward problem is construct, with the same makespan.

Proof 2: Let S be a feasible schedule for $F_2\left(P_{m_1}, P_{m_2}\right)\left|t_j\right|$ $C_{\max}$, the same assignment of the jobs on the first and second centers is conserved as for S. The scale of the time is inverted, such that the $C_{\max}$ time, is the new origin of the new scale. This can be done by considering the linear transformation $t^R = C_{\max} - t$, where t is the time in the old scale and t^R the time in the new one. The result is a feasible schedule S^R for the reverse problem, where the job $j \in J$ is processed first in C_2 within the time interval $\left[C_{\max} - c_{2,j}(S), C_{\max} - c_{2,j}(S) + p_{2,j}\right]$ and in the second stage C_1 during the interval $\left[C_{\max} - c_{1,j}(S), C_{\max} - c_{1,j}(S) + p_{1,j}\right]$, with $c_{i,j}(S)$ $(i = 1, 2.j \in J)$ the completion times for the job j in the center C_i regarding to the schedule S. The two schedules S and S^R have the same makespan, since there is no modification on the affectation of the jobs, and the critical path's length, which gives the makespan value. This value is invariant by the transformation $t' = C_{\max} - t$. Similarly, for a feasible schedule of the reverse problem, a feasible schedule for $F_2\left(P_{m_1}, P_{m_2}\right)\left|t_j\right| C_{\max}$, with the same makespan, can be easily deduced, by applying the same procedure as for the construction of S' from S. Hence, the two problems are equivalents.

The latter proposition yields the following result.

Corollary 1: The Forward and the Backward problems have the same optimal makespan.

Proof 3: Based on the symmetry, a feasible schedule for the *Forward* problem is also a feasible schedule for the *Backward* problem (after the transformation $t^R = C_{\max} - t$), with same makespan. Conversely, a feasible schedule for the *Backward* problem is a feasible schedule for the *Forward* problem, with the same makespan. Particularly, this holds for an optimal schedule.

Remark 1: The reverse problem will be systematically investigated for the proposed procedures: lower bounds, upper bounds, and exact solution, for possible improvements, as a consequence of the latter corollary.

At this stage we define for each job $j \in J$, and each center k $(k = 1, 2)$ a release date r_{kj} and a delivery time q_{kj} that are:

$$\begin{cases} r_{2j} = p_{1j} + t_j \\ r_{1j} = 0, \end{cases} \quad \text{and} \quad \begin{cases} q_{1j} = t_j + p_{2j} \\ q_{2j} = 0, \end{cases}$$

respectively.

III. LOWER BOUNDS

In this section, a set of lower bounds for the two centers hybrid flow shop scheduling problem with transportation times is developed and presented. The first lower bound is based on the relaxation of the centers' capacities. In addition, the SPT rule allows estimating the minimum idle time in a center. This minimum idle time contributes in the development of the second lower bound.

A. A RELAXED CAPACITY BASED LOWER BOUND

Relaxing the capacity constraint in the second center C_2 (assuming that the number of machines in C_2 is unlimited), allows any job to exit C_2 exactly after p_{2j} unites of time. Thus, the obtained relaxed problem in the first stage C_1 is a parallel machine problem $P_{m_1}\left|r_j, q_j\right| C_{\max}$ with:

- release date $r_j = 0$,
- processing time $p_j = p_{1j}$,
- delivery time $q_j = t_j + p_{2j}$.

Therefore, any lower bound for the $P_{m_1}\left|r_j, q_j\right| C_{\max}$ problem is a lower bound for the $F_2\left(P_{m_1}, P_{m_2}\right)\left|t_j\right| C_{\max}$. Particularly, if $C^1_{\max}$ is the value of an optimal solution for the $P_{m_1}\left|r_j, q_j\right| C_{\max}$ problem, then

$$LB_1 = C^1_{\max}$$

is a valid lower bound for the $F_2\left(P_{m_1}, P_{m_2}\right)\left|t_j\right| C_{\max}$ problem.

Similarly, relaxing the capacity constraint of the first stage C_1 leads to a parallel machine problem $P_{m_2}\left|r_j, q_j\right| C_{\max}$ in the second stage C_2, where:

- the release date $r_j = p_{1j} + t_j$,
- the processing time $p_j = p_{2j}$,
- the delivery time $q_j = 0$.

Clearly, if $C^2_{\max}$ is the value of an optimal solution of $P_{m_2}\left|r_j, q_j\right| C_{\max}$ problem, then

$$LB_2 = C^2_{\max}$$

is a valid lower bound for the studied problem.

Remark 2: The presented exact branch and bound algorithm in [26] is used to solve the parallel machine scheduling problem $P_m\left|r_j, q_j\right| C_{\max}$. Since, the $P_m\left|r_j, q_j\right| C_{\max}$ problem is NP-Hard it might happen that within a time limit the latter problem is not solved optimally. In this case, the best obtained lower bound while solving the problem is picked instead of $C^1_{\max}$ or(and) $C^2_{\max}$ in the expressions of LB_1 or (and) LB_2.

B. AN SPT BASED LOWER BOUND

Each job $j \in J$ should be processed on the first center C_1 and then transported to the second center C_2, where it will be processed by an available machine. Consequently, each machine $M_{2,s}$ $(1 \le s \le m_2)$ of the second center C_2 cannot start processing a job at time zero. Therefore, an idle time $I_s(J)$ $(1 \le s \le m_2)$ appears in each machine $M_{2,s}$, which is at least equal to the required duration for the arrival of the first job on it. Clearly, for an optimal schedule if $I_2(J)$ is a lower bound of the total idle time in the second center C_2, then,

Proposition 2:

$$LB^2(J) = \left\lceil \frac{I_2(J) + \sum_{j \in J} p_{2,j}}{m_2} \right\rceil$$

is a valid lower for the $F_2\left(P_{m_1}, P_{m_2}\right)\left|t_j\right| C_{\max}$ problem.

Proof 4: In an optimal solution (with optimal makespan $C^*_{\max}$), and for each machine $M_{2,s}$ $(1 \leq s \leq m_2)$, the window time $\left[0, C^*_{\max}\right]$ is composed of idle times $IM_{2,s}$ and the processing times $PM_{2,s}$ of the scheduled jobs in $M_{2,s}$. Thus, $IM_{2,s}+PM_{2,s} \leq C^*_{\max}$ and $\sum_{1 \leq s \leq m_2} \left(IM_{2,s} + PM_{2,s}\right) \leq m_2 C^*_{\max}$. Since, $I_2(J) \leq \sum_{1 \leq s \leq m_2} IM_{2,s}$ and $\sum_{1 \leq s \leq m_2} PM_{2,s} = \sum_{j \in J} p_{2,j}$ then $\left\lceil \frac{I_2(J) + \sum_{j \in J} p_{2,j}}{m_2} \right\rceil \leq C^*_{\max}$ which ends the proof.

An explicit expression of a lower bound of the total idle time in the second center is developed over the following lemma.

Lemma 2: Let $\bar{p}_{1j}, \bar{t}_j$be the sorted lists in the increasing order of p_{1j} and t_j. In addition, let $SPT\ (m_2)$ denotes the sum of the completion times of the m_2 jobs having the smallest $p_{1,j}$ scheduled in the first center according to the Shortest Processing Time(SPT) rule. Then $SPT\ (m_2) + \sum_{j=1}^{m_2} \bar{t}_j$ is a lower bound for the total idle time in the second center C_2.

Proof 5: For an optimal schedule σ^*, the completion time of the job $j \in J$ in the first center C_1 is denoted $c_{1j}\left(\sigma^*\right)$. Observing that the m_2 first processed jobs $j_1, j_2, \ldots, j_{m_2}$ in the second center C_2 induce an idle time equal to $\sum_{s=1}^{m_2} \left(c_{1j_s}\left(\sigma^*\right) + t_{j_s}\right) \leq I_2(J)$. Indeed, the processing of j_s begins at $c_{1j_s}\left(\sigma^*\right)+t_{j_s}$ $(1 \leq s \leq m_2)$. Based on [14] and [15], $\sum_{s=1}^{m_2} c_{1j_s}\left(\sigma^*\right) \geq SPT\ (m_2)$. Moreover, $\sum_{s=1}^{m_2} t_{j_s} \geq \sum_{s=1}^{m_2} \bar{t}_j$,then $SPT\ (m_2) + \sum_{j=1}^{m_2} \bar{t}_j \leq \sum_{s=1}^{m_2} \left(c_{1j_s}\left(\sigma^*\right) + t_{j_s}\right) \leq I_2(J)$.

Based on the latter proposition and previous lemma an immediate result is:

Corollary 2: A valid lower bound for the studied problem is

$$LB^2_{SPT} = \left\lceil \frac{SPT\ (m_2)\ + \sum_{j=1}^{m_2} \bar{t}_j + \sum_{j \in J} p_{2,j}}{m_2} \right\rceil.$$

According to the symmetry of the problem, one can deduce easily that

$$LB^1_{SPT} = \left\lceil \frac{SPT\ (m_1)\ + \sum_{j=1}^{m_1} \bar{t}_j + \sum_{j \in J} p_{1,j}}{m_1} \right\rceil$$

is a lower bound for the considered problem.

Clearly we have:

Corollary 3: A valid lower bound for the considered problem is given by: $LB = \left(LB_1, LB_2, LB^2_{SPT}, LB^1_{SPT}\right)$.

IV. HEURISTIC

The proposed heuristic is composed of two phases. The first one consists in developing an initial feasible schedule, while the second phase is intended to the improvement of this initial solution. The pillar of this heuristic is the parallel machine scheduling problem with release date and delivery time $P_m \left|r_j, q_j\right| C_{\max}$, as well as its equivalent problem $P_m \left|r_j, d_j\right| L_{\max}$, which considers the maximum lateness as objective to be minimized. Consequently, this part will start by presenting the two latter problems and the equivalence relationship between them.

Proposition 3: The $P_m \left|r_j, q_j\right| C_{\max}$ problem is equivalent to $P_m \left|r_j, d_j\right| L_{\max}$ problem. In other terms, solving one of them is equivalent to solving the other one.

Proof 6: For an instance of the $P_m \left|r_j, q_j\right| C_{\max}$ problem an equivalent instance for the $P_m \left|r_j, d_j\right| L_{\max}$ problem could be deduced by setting for each job $j \in J$ a due date $d_j = C - q_j$, where C an upper bound of the optimal makespan. Conversely, for an instance of the $P_m \left|r_j, d_j\right| L_{\max}$ problem, let $D = \max_{j \in J} d_j$, $q_j = D - d_j$ $(j \in J)$, and s_j the starting time of the job j in a feasible schedule. Then, $L_{\max} = \max_{j \in J} \left(s_j + p_j - d_j\right) = \max_{j \in J} \left(s_j + p_j - D + q_j\right) = \max_{j \in J} \left(s_j + p_j + q_j\right) - D = C_{\max} - D$.

At this stage, the developed heuristic is introduced over the following algorithm.

A. PHASE 1: INITIAL FEASIBLE SOLUTION

1.1 Set $m = m_1$, $r_j = 0$, $p_j = p_{1j}$, and $q_j = p_{2,j} + t_j$ $(j \in J)$.
1.2 Solve exactly the corresponding $P_m \left|r_j, q_j\right| C_{\max}$ problem.
1.3 For each $j \in J$, Set $c_{\mathbf{1j}}$ as the completion time for the obtained solution in **STEP 1.2**.
1.4 Set $m = m_2$, $r_j = c_{\mathbf{1j}} + t_j$, $p_j = p_{2,j}$, and $q_j = 0$ $(j \in J)$.
1.5 Solve exactly the corresponding $P_m \left|r_j, q_j\right| C_{\max}$ problem.
1.6 For each $j \in J$, Set t_{2j} as the starting time for the obtained solution in **STEP 1.4**.
1.7 Set $UB = \max_{j \in J} \left(t_{2j} + p_{2,j}\right)$.

B. PHASE 2: IMPROVEMENT PHASE

2.1 Set $m = m_1$, $r_j = 0$, $p_j = p_{1,j}$, and $d_j = t_{2j}$ $(j \in J)$.
2.2 Solve exactly the corresponding $P_m \left|r_j, d_j\right| L_{\max}$ problem.
2.3 For each $j \in J$, Set $c_{\mathbf{1j}}$ as the completion time for the obtained solution in **2.2**.
2.4 **IF** $L^*_{\max} = 0$ then **STOP**, **ELSE** Set $UB := UB + L^*_{\max}$.
2.5 Set $m = m_2$, $r_j = c_{\mathbf{1j}} + t_j$, $p_j = p_{2,j}$, and $q_j = 0$ $(j \in J)$.
2.6 Solve exactly the corresponding $P_m \left|r_j, q_j\right| C_{\max}$ problem.
2.7 For each $j \in J$, Set t_{2j} as the starting time for the obtained solution in **STEP 2.6**.
2.8 **IF** $C^*_{\max} < UB$ then Set $UB = C^*_{\max}$.
2.9 Go to **STEP 2.1.**

During the first phase, two consecutive parallel machine scheduling problems $P_m \left| r_j, q_j \right| C_{\max}$ are solved optimally. The first one is presented over **STEP 1.1** and **STEP 1.2,** while the second one is described in **STEP 1.3, STEP 1.4,** and **STEP 1.5.** Since the jobs are available for processing in center C_2 at time $r_j = c_{\mathbf{1j}} + t_j$ (**STEP 1.4),** then the unidirectional flow constraint is satisfied (i.e. a job is processed firstly in center C_1, and after that transported to the second center C_2 to be treated). Consequently, a feasible schedule for the studied scheduling problem is obtained by the concatenation of the two feasible schedules from **STEP 1.2** and **STEP 1.4.** Obviously, the value of the makespan of the latter feasible schedule is UB. The improvement phase, is an iterative procedure between the two centers. Each iteration consists in fixing the schedule in a center and trying to reschedule in the other one without deteriorating the existing solution. More precisely, the feasible schedule in the second center C_2 is fixed and the due dates of the jobs, in the first center are set as $d_j = t_{\mathbf{2j}} - t_j$ $(j \in J)$. Furthermore, in **STEP 2.2** the obtained $P_{m_1} \left| r_j, d_j \right| L_{\max}$ problem is solved and the resulted optimal value is denoted $L^*_{\max}$. Because of the choice of d_j, the optimal value satisfies $L^*_{\max} \leq 0$ (the existing solution is not deteriorated). Therefore, if $L^*_{\max} < 0$ an improvement is detected and the new makespan is updated to $UB + L_{\max}$. After that, the obtained schedule in the first center is fixed by setting $r_j = c_{\mathbf{1j}} + t_j$ for the second center (**STEP 2.5**).The obtained $P_{m_2} \left| r_j, q_j \right| C_{\max}$ problem is solved (**STEP 2.6**) and the optimal value is denoted $C^*_{\max}$.If $C^*_{\max} < UB$, then once again the makespan is improved and updated to the value $UB := C^*_{\max}$. The improvement phase is stopped the first time reaching $L^*_{\max} = 0$. Thanks to the symmetry propriety of the studied problem, the latter heuristic is performed for the reverse problem, where the obtained makespan is denoted UB^R. Thus, the selected feasible schedule is the one with value $UB := \min\left(UB, UB^R\right)$.

Remark 3: The main components of the latter heuristic is the resolution of the two NP-Hard problems $P_m \left| r_j, q_j \right| C_{\max}$ and $P_m \left| r_j, d_j \right| L_{\max}$. This is done by the exact algorithm from [26]. In the case where an optimal solution is not reached within a limit time, the best upper bound (best lateness) and the corresponding feasible schedule substitute $C^*_{\max}$ ($L^*_{\max}$).

V. BRANCH AND BOUND EXACT PROCEDURE

In order to solve a NP-Hard problem, generally we start out by developing a heuristic and a lower bound for the addressed problem. The advantage of using a heuristic is the low consumed time. If the heuristics value is equal to the lower bound, then the obtained solution is optimal. If the heuristic fails to reach the optimal solution and the latter condition is not satisfied, then two choices have to be considered. The first one is to stop with only a feasible solution even with mediocre value (high value of the relative gap) and this is the disadvantage of using heuristic. The second choice is to use a branch and bound algorithm to find and optimal solution, bounded by the already developed heuristic and the lower bound. Therefore, the heuristic is an input for the branch and bound algorithm. The major disadvantage of the branch and bound algorithm is the high consumed time while solving the problem. However, using the branch and bound carefully by embedding efficient lower bounds and dominance rules could speed up the procedure as it has been done in this paper.

In this section, a Branch and Bound (B&B) exact algorithm is developed. Recall that a B&B procedure consists in decomposing the original treated problem into a partition of small and easy solving sub-problems. This procedure results in a searching tree, where each node represents a partial solution and then a solution representation is required. In addition, moving from a node to another one corresponds to the so called branching strategy. Since the size of the searching tree is important, then eliminating dominated nodes (that will not conduct the search to optimal solution) is one of the main keys to speed up the convergence of the B&B. The nodes elimination could be done by computing a lower bound at each node and compare it with an upper bound.

A. SOLUTIONS REPRESENTATION

Firstly, observing that a feasible schedule σ in the first center induces a natural parallel machine scheduling problem $P_{m_2} \left| r_j \right| C_{\max}$ in the second center. This scheduling problem is obtained by setting the release dates as $r_j = c_{1j}(\sigma) + t_j$ and the processing times $p_j = p_{2j}$ for each job $j \in J$, where $c_{1j}(\sigma)$ stands for the completion time of job j in the feasible schedule σ. Clearly, solving the studied scheduling problem $\left(F_2\left(P_{m_1}, P_{m_2}\right) \left| t_j \right| C_{\max}\right)$ is equivalent to finding a feasible schedule σ^* in the first center satisfying $\overline{C}(\sigma^*) = \min\limits_{\sigma \in \Gamma} \overline{C}(\sigma)$, where Γ is the set of all feasible schedules in the first center, and $\overline{C}(\sigma)$ the optimal makespan of the $P_{m_2} \left| r_j \right| C_{\max}$ problem quoted above. In addition, a feasible schedule $\sigma \in \Gamma$ is represented throughout a permutation $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ of n jobs, where the respective starting times s_{σ_i} and s_{σ_j} of the jobs σ_i and σ_j, respectively, satisfy $s_{\sigma_i} \leq s_{\sigma_j}$ for $1 \leq i < j \leq n$. Thus, a feasible solution of the studied problem is represented by a permutation of the n jobs.

B. BRANCHING STRATEGY

Any node N_k at the level k of the search tree is assumed to represent a partial feasible schedule $\sigma(N_k)$, where k jobs are scheduled in the first center. This partial feasible schedule $\sigma(N_k)$ could be represented over a partial permutation $(\sigma_1, \sigma_2, \ldots, \sigma_k)$ of k jobs $(1 \leq k \leq n)$. This is done according to the latter subsection, where a feasible schedule is represented over a permutation of the n jobs. The set of unscheduled jobs J_{US} has $n - k$ members, where each one of them is considered as a descendant. Consequently, branching from node N_k corresponds to the selection of an unscheduled job $j_0 \in J_{US}$ and building a node N_{k+1} with $(\sigma_1, \sigma_2, \ldots, \sigma_k, j_0)$ the corresponding partial permutation. When reaching the level $n - 1$, a feasible schedule in the first center C_1 is obtained. Therefore, a feasible schedule for the studied problem is obtained by solving the $P_{m_2} \left| r_j \right| C_{\max}$ problem as described in the latter subsection

(Solutions representation). The depth first strategy is adopted in this B&B.

C. BOUNDING STRATEGY

This part is devoted to the development and the presentation of two lower bounds at the nodes, in addition to a faster upper bound. Prior to that, some definitions and notations are given. Let N be a node in the search tree such that $N \neq N_0$ (N_0 is the root node: node at level 0), and denoting the set of already scheduled jobs by J_S, then the unscheduled jobs $J_{US} = J \setminus J_S$. Each job $j \in J_S$ is already scheduled and then has a completion time denoted c_{1j}. Moreover, each machine $M_{1,i}$ $(1 \leq i \leq m_1)$ of the first center has an availability time u_i. The availabilities of these machines are assumed to be sorted in the increasing order : $u_1 \leq u_2 \leq \ldots \leq u_{m_1}$(without loss of generality).

1) RELAXING CAPACITY BASED LOWER BOUND

Relaxing the capacity of the first center implies that an unscheduled job j is ready to be processed in the second center at time $u_1 + p_{1,j} + t_j$. In addition, a scheduled job j is available for processing in the second center at time $r_j = c_{1j} + t_j$. Thus, a parallel machine scheduling problem $P_{m_2}\left|r_j\right| C_{\max}$ is obtained by setting the release dates r_j and the processing times p_j as follows.

$$\begin{cases} r_j = c_{1j} + t_j & \text{if } j \in J_S \\ r_j = u_1 + p_{1j} + t_j & \text{if } j \in J_{US} \\ p_j = p_{2j} & j \in J \end{cases} \tag{1}$$

A first lower bound in node N is depicted throughout the following proposition.

Proposition 4: Let T be a subset of jobs ($T \subseteq J$), satisfying $|T| \geq m_2$. Denoting $\bar{r}_j(T)$ the j^{th} release date sorted in the increasing order in T, then

$$LB_N^1(T) = \left\lceil \frac{\sum\limits_{1 \leq j \leq m_2} \bar{r}_j(T) + \sum\limits_{j \in T} p_{2,j}}{m_2} \right\rceil$$

is valid lower bound for the obtained sub-problem at node N.

Proof 7: Let $C^*_{\max}$ be the optimal solution for the already defined problem $Pm_2\left|r_j\right| C_{\max}$. In each machine $M_{2,i}$ $(i = 1, 2, \ldots., m_2)$, of the second center C_2, the first processed job starts processing at $t_{2,i}$. Moreover, each machine presents an idle time $I_i(T)$ and a load time $P_i(T)$. Clearly we have $t_{2,i}(T) + I_i(T) + P_i(T) \leq C^*_{\max}$ for $i = 1 \ldots. m_2$. Thus, $\sum\limits_{1 \leq i \leq m_2} t_{2,i}(T) + \sum\limits_{1 \leq i \leq m_2} I_i(T) + \sum\limits_{1 \leq i \leq m_2} P_i(T) \leq m_2 C^*_{\max}$. Remarking that $\sum\limits_{1 \leq k \leq m_2} \bar{r}_k(T) \leq \sum\limits_{1 \leq i \leq m_2} t_{2,i}(T)$, and $\sum\limits_{1 \leq i \leq m_2} P_i(T) = \sum\limits_{j \in T} p_{2,j}$. One can deduce easily that $\sum\limits_{k=1,m_2} \bar{r}_k(T) + \sum\limits_{j \in T} p_{2,j} \leq m_2 C^*_{\max}$, which ends the proof.

$LB_N^1(T)$ might be enhanced by exploring all the subsets ($T \subseteq J$), such that $|T| \geq m_2$, in this case $LB_N^1 = \max\limits_{T \subseteq J; |T| \geq m_2} LB_N^1(T)$ is a valid lower bound. Unfortunately such an exploration requires to check up an exponential number subsets, which is out of reach. In the following proposition, it will be shown that a few number of subsets is sufficient.

Proposition 5: $LB_N^1 = \max\limits_{T \subseteq J; |T| \geq m_2} LB_N^1(T)$ is valid lower bound for the obtained sub-problem at node N,that is computed in $O(n \log m_2)$.

Proof 8: Clearly, $\bar{r}_{m_2}(T) \in \{r_{(m_2)}, r_{(m_2+1)}, \ldots, r_{(n)}\}$ where $r_{(i)}$ is the i^{th} release date sorted in the increasing order in J. Let T_k $(k = m_2, \ldots, n)$ be a subset of J verifying $LB_N^1(T_k) = \max\limits_{T \subseteq J: \bar{r}_{m_2}(T) = r_{(k)}} LB_N^1(T)$. Consequently, $LB_N^1 = \max\limits_{m_2 \leq k \leq n} LB_N^1(T_k)$. Define $f(T) = \sum\limits_{j=1,m_2} \bar{r}_j(T) + \sum\limits_{j \in T} p_{2,j}$ for $T \subseteq J$. We remark that $f(T_{k+1}) = f(T_k) + r_{k+1} - r_{j_k} - p_{2,j_k}$ for $k = m_2, \ldots, n-1$ with $j_k = \arg\min\limits_{j \in T_k}(r_j + p_{2,j})$.The computation of $f(T_{m_2})$ requires $O(n)$ time. Given $f(T_k)$, the determination of the job j_k is the main task in the computation of $f(T_{k+1})$. This could be done in $O(\log m_2)$ time. Therefore, the computation of $f(T_k)$ for $k = m_2, \ldots, n$ can be done in $O(n \log m_2)$ time.

2) AN AVAILABILITY MACHINES BASED LOWER BOUND

In the first center a parallel machine scheduling problem, with availabilities of machines and delivery times $P, NC_{inc}\left|q_j\right| C_{\max}$ is defined as follows. The availability v_i of each machine $M_{1,i}$ $(i = 1, 2, \ldots., m_2)$ is the completion time of the last scheduled job on it. Let u_i be the i^{th} availability sorted in the increasing order in $\{v_1, v_2, \ldots, v_{m_1}\}$. The set of the jobs to be considered is J_{US}, the delivery time and the processing time of a job $j \in J_{US}$ are $q_j = t_j + p_{1,j}$ and $p_j = p_{1,j}$, respectively. A lower bound of the considered $P, NC_{inc}\left|q_j\right| C_{\max}$, which is NP-Hard, is a lower bound for the sub-problem associated with the explored node N. Due to the availabilities constraint, the number of the utilized machines in an optimal schedule could be less than m_1. In this context, Gharbi and Haouari [28] presented the following proposition.

Proposition 6: If UB is an upper bound on the optimal makespan of $P, NC_{inc}\left|q_j\right| C_{\max}$ problem, then the number of machines m to be utilized satisfies $m_l(J_{US}) \leq m \leq m_u(J_{US})$, where :

- $m_l(J_{US}) = \left\lceil \dfrac{\sum\limits_{j \in J_{US}} p_{1,j}}{UB - u_1 - \bar{q}_1(J_{US})} \right\rceil$ *where* $\bar{q}_1(J_{US}) = \min\limits_{j \in J_{US}} q_j$
- $m_u(J_{US})$ *as the smallest* k $(k = 1, \ldots, m_1 - 1)$ *satisfying* $u_{k+1} + \min\limits_{j \in J_{US}}(p_{1,j} + q_j) > UB$*. If no* k *satisfies this condition, then* $m_u(J_{US}) = m_1$.

At this stage, the second lower bound is presented in the following proposition.

Proposition 7: Assume that the jobs of J_{US} are assigned to exactly m machines of the first center C_1, then a valid lower bound which is defined for a given subset $T \subseteq J_{US}$ is:

$$LB_N^2(T,m) = \left\lceil \frac{\sum\limits_{i=1,m} u_i + \sum\limits_{j\in T} p_{1,j} + \sum\limits_{k=1,m} \bar{q}_k(T)}{m} \right\rceil,$$

where $\bar{q}_k(T)$ is defined as the k^{th} smallest delivery time in the subset T.

Proof 9: Indeed, if $C^*_{\max}$ is the optimal makespan of the $Pm, NC_{inc} \left| q_j \right| C_{\max}$. For each machine $M_{1,i}$ $(i = 1, 2, ..., m_1)$, the first job starts processing at time $t_{1,i}(T)$, the last scheduled job jobs waits at least for $w_{1,i}(T)$ to exit the system. Further, let $P_i(T)$ be the total load in $M_{1,i}$, then $t_{1,i}(T) + P_i(T) + w_{1,i}(T) \leq C^*_{\max}$. Hence, $\sum\limits_{i=1,m} t_{1,i}(T) + \sum\limits_{i=1,m} P_i(T) + \sum\limits_{i=1,m} w_{1,i}(T) \leq mC^*_{\max}$. Since $\sum\limits_{i=1,m} u_i \leq \sum\limits_{i=1,m} t_{1,i}(T)$, $\sum\limits_{i=1,m} P_i(T) = \sum\limits_{j\in T} p_{1,j}$, and $\sum\limits_{k=1,m} \bar{q}_k(T) \leq \sum\limits_{i=1,m} w_{1,i}(T)$, then $\sum\limits_{i=1,m} u_i + \sum\limits_{j\in T} p_{1,j} + \sum\limits_{k=1,m} \bar{q}_k(T) \leq mC^*_{\max}$. This proofs the latter proposition.

Clearly, $LB_N^2(m) = \max\limits_{T\subseteq J_{US}} LB_N^2(T,m)$ is a valid lower bound. Moreover,

Proposition 8: $LB_N^2(m)$ is calculated in $O(n \log m)$.

Proof 10: The proof is similar to the one done for proposition 5.

Consequently,

Corollary 4: A valid lower bound for the sub-problem at node N is

$$LB_N^2 = \min_{m_l(J_{US}) \leq m \leq m_u(J_{US})} LB_N^2(m),$$

which can be computed in $O(m_1 n \log m_1)$.

Proof 11: Immediate.

Clearly, we have:

Corollary 5: a valid lower bound at each node is $LB_N = \max\left(LB_N^1, LB_N^2\right)$.

3) UPPER BOUNDING STRATEGY

At each non-root node N a priority list based heuristic **H** is presented. This heuristic schedules the job with largest $p_{2,j}$ in J_{US}, in the first available machine in the first center C_1. Once all the jobs are scheduled in the first center, at any time and among the available jobs, the one with largest $p_{2,j}$ is scheduled in the first available machine in center two. In the sequel the **H** corresponding algorithm is presented.

```
Heuristic H
Phase 1 - Scheduling in $C_1$
1.1. Sort the jobs in $J_{US}$ according to the decreasing order of $p_{2,j}$. Set $V = J_{US} \cdot J$
1.2. Schedule a job $j \in V$ with largest $p_{2,j}$. Set $V = V \setminus \{j\}$
1.3. If $V \neq \emptyset$ then go to Step 1.2
Phase 2 - Scheduling in $C_2$
2.1. Set a release date $r_j = c_{1,j}$ for each job $j \in J$ (completion time of $j$ on center $C_1$ ). Set $V = J$
2.2. Schedule an already released job $j \in V$ with largest $p_{2,j}$. Set $V = V \setminus \{j\}$
2.3. If $V = \emptyset$ then Stop, Else go to Step 2.2
```

Sorting the jobs in step **1.1** is the main effort to be provide in H, which is done in $O(n \log n)$ time. This sorting is done only one time, thus the computation effort of **H** at each node is $O(n)$.

D. ADDITIONAL IMPROVEMENTS

In addition to the main components of a B&B procedure, some further enhancements could be performed. These enhancements reduce the search tree size and therefore decrease the total consumed time while running the B&B procedure. In this context, two main improvements will be presented in this section.

1) DOMINANCE RULES

The dominance rules are intended to eliminate some nodes from the search tree. These nodes are dominated by others and consequently there is no need to be explored. These dominance rules contribute in reducing the consumed time while performing the exploration of the search tree. The majority of the proposed dominance rules are retrieved from [29] and adapted for the $F_2\left(P_{m_1}, P_{m_2}\right)\left|t_j\right| C_{\max}$. At node N, the set of unscheduled jobs J_{US} is sorted according to the priority to be scheduled and added to the partial schedule σ. In this case J_{US} will be denoted as $J_{US} = \{j_1, j_2, \ldots, j_K\}$. Given that, the following rules hold:

- D_1 If two jobs j_k and j_{k+1} in J_{US} have the same data: $p_{1,j_k} = p_{1,j_{k+1}}$, $t_{j_k} = t_{j_{k+1}}$, and $p_{2,j_k} = p_{2,j_{k+1}}$, then exploring the node corresponding to the partial schedule σj_{k+1} is not required, since the same partial schedule σj_k is already explored.
- D_2 If $u_1 = u_2$ then the partial schedules $\sigma\,(j_k)\,(j_{k+1})$ and $\sigma\,(j_{k+1})\,(j_k)$ are equivalent and the node corresponding to $\sigma\,(j_{k+1})\,(j_k)$ is eliminated. This dominance rule is considering the symmetry of the available machines.

An important propriety of the considered problem is its symmetry, which has been established in the proprieties section. Because of this propriety the reverse problem (backward) is explored systematically in the development of the lower bounds, the upper bound in order to improve their qualities. In the development of the B&B procedure the reverse problem will be explored. In this context, the developed B&B procedure will be solved iteratively for the original (*Forward*) problem and its reverse version (*Backward*). A given time limit is set up. If the B&B is not able to find an optimal solution within this time limit for the *Forward* problem (*Backward* problem) the *Backward* problem (*Forward* problem) is investigated. This cyclic procedure is halted if the optimal solution is not detected or the existing lower and upper bounds are not improved.

2) AN ITERATIVE FORWARD-BACKWARD IMPLEMENTATION

An important propriety of the considered problem is its symmetry, which has been established in the proprieties section. Because of this propriety the reverse problem (backward) is explored systematically in the development of the lower bounds, the upper bound in order to improve their qualities. In the development of the B&B procedure the reverse problem will be explored. In this context, the developed B&B procedure will be solved iteratively for the original (*Forward*) problem and its reverse version (*Backward*). A given time limit is set up. If the B&B is not able to find an optimal solution within this time limit for the *Forward* problem (*Backward* problem) the *Backward* problem (*Forward* problem) is investigated. This cyclic procedure is halted if the optimal solution is not detected or the existing lower and upper bounds are not improved.

VI. COMPUTATIONAL RESULTS

This section is intended to the evaluation of the proposed algorithms. This evaluation is based on an intensive experimental computation study throughout a set of instances. The details of these instances are presented in the sequel. All the algorithms are coded in C and implemented in a Pentium IV 2.8GHz Personal Computer with 1GBRAM.

A. TEST PROBLEMS

The used test problems are a generalization of the ones proposed in [27], more specifically:

- The number of jobs $n \in \{10, 20, 30, 40, 50, 100, 150, 200\}$.
- The couple of the number of machines at each center $(m_1, m_2) \in \big\{(2, 2), (2, 4), (4, 2), (4, 4), (3, 3), (3, 5), (5, 3), (5, 5)\big\}$.
- The transportation times and the processing times t_j, $p_{1,j}$, and $p_{2,j}$ $(j \in J)$ are generated using the uniform distribution in the following intervals.
  - $p_{1,j}$ in $[1, a]$,
  - t_j in $[1, b]$,
  - $p_{2,j}$ in $[1, c]$.

Where $a, b, c \in \{20, 40\}$. A number of 10 instances is generated, for each combination of n, m_1, m_2, a, b, and c. Consequently, a set of 5120 instances is obtained.

B. PERFORMANCE OF THE PROPOSED PROCEDURES

The experimental results are reported in Tables 2-13. The parameters allowing the evaluation of the performance of the addressed problem in these tables are presented below.

- $[a : b : c]$: means that the processing times in C_1, the processing times in C_2, and the transportation times are generated uniformly from $[1, a]$, $[1, b]$,and $[1, c]$, respectively.
- SR : number of solved instances at the root node $(UB = LB)$.
- SB : number of solved instances by the B&B.
- $\%S$: percent of solved instances.
- MT : average CPU time (in seconds) for the solved instances.
- MG : is the average relative gap of unsolved instances where the relative gap is $rg = \dfrac{UB - LB}{LB} \times 100$ for each instance.
- $MaxG$: is the maximum relative gap of the unsolved instances.

The overall results are summarized in Table 2.

TABLE 2. Performance of the proposed procedures.

SR	SBB	UN	$\%S$	MT	MG	$MaxG$
2786	1488	846	83.48	79.27	2.51	18.42

According to Table 2, we observe that 55% (2786 out of 5120) of the test problems are solved at the root node, without need to run the B&B algorithm. Consequently, the developed lower bounds and the heuristic are efficient. For the remaining unsolved problems, the B&B procedure provides the optimal solution for 29.1% (1488 out of 5120) of the instances. The average required CPU time to solve the instances is 79s, which is a moderate CPU time. Furthermore, the average relative gap for the unsolved instances is 2.5. All these arguments proof that the proposed procedures are able to solve large size instances(up to 200 jobs) in an acceptable CPU time.

The impact of the number of jobs on the obtained results is exhibited over Table 3.

TABLE 3. Performance of the proposed procedures according to the number of jobs.

n	SR	SBB	UN	$\%S$	MT	MG	$MaxG$
10	426	214	0	100.00	0.03	0.00	0.00
20	293	231	116	81.88	137.12	3.37	17.11
30	316	190	134	79.06	92.65	4.93	18.42
40	322	191	127	80.16	80.82	3.77	12.82
50	347	162	131	79.53	72.17	2.53	13.79
100	337	201	102	84.06	92.04	1.21	8.30
150	364	163	113	82.34	95.76	0.70	4.38
200	381	136	123	80.78	80.98	0.50	3.55

Based on Table 3, one can observe that the easiest instances to be solved are those for $n = 10$ where all of them are solved. Surprisingly, the percentage of the solved problems is almost not varying with respect to n and it ranges between 80% and 84%. Remarkably, the average consumed CPU time to solve problems with $n = 20$ is the highest one (137.11s). Further, the average relative gap for the unsolved problems is almost decreasing as the number of jobs increases, and it reaches its lowest value for $n = 200$. Thus the developed procedures are performing well especially for the large size problems (with respect to n).

A more specific analysis is carried out to study the effect of the number of the machines in the two centers and the results are reported in Table 4.

The provided results in Table 4 show that the percentage of solved problems is impacted by number of machines. More precisely, the lowest percentage 65.63% is reached for $m_1, m_2 = (5, 5)$. Moreover, the three smallest percentage of solved problems are 65.63%, 73.59%, 82.97% reached for

TABLE 4. Performance of the proposed procedures with respect to the number of machines.

m_1	m_2	SR	SBB	UN	$\%S$	MT	MG	$MaxG$
2	2	300	287	53	91.72	88.10	2.34	11.32
2	4	426	165	49	92.34	26.01	1.97	11.72
3	3	281	250	109	82.97	108.02	2.58	17.11
3	5	376	160	104	83.75	40.50	2.01	12.41
4	2	424	160	56	91.25	71.17	2.19	11.93
4	4	288	183	169	73.59	102.71	2.52	14.00
5	3	382	172	86	86.56	117.40	2.09	15.08
5	5	309	111	220	65.63	89.69	3.12	18.42

TABLE 5. Performance of the proposed procedures according to m_1, m_1 and $n \in \{10, 20, 30, 40\}$.

n	(m_1, m_2)	SR	SBB	$\%S$	MT	MG	$MaxG$
10	$(2,2)$	31	49	100.00	0.07	0.00	0.00
10	$(2,4)$	53	27	100.00	0.03	0.00	0.00
10	$(4,2)$	31	49	100.00	0.11	0.00	0.00
10	$(4,4)$	60	20	100.00	0.01	0.00	0.00
10	$(3,3)$	56	24	100.00	0.01	0.00	0.00
10	$(3,5)$	57	23	100.00	0.01	0.00	0.00
10	$(5,3)$	65	15	100.00	0.01	0.00	0.00
10	$(5,5)$	73	7	100.00	0.00	0.00	0.00
20	$(2,2)$	35	35	87.50	103.77	2.31	4.63
20	$(2,4)$	50	23	91.25	50.46	1.93	3.82
20	$(4,2)$	29	36	81.25	187.90	3.25	17.11
20	$(4,4)$	41	24	81.25	104.65	3.74	7.84
20	$(3,3)$	50	24	92.50	118.76	3.00	7.08
20	$(3,5)$	22	34	70.00	201.87	4.02	10.45
20	$(5,3)$	43	29	90.00	158.80	3.38	7.61
20	$(5,5)$	23	26	61.25	211.40	3.47	8.96
30	$(2,2)$	31	38	86.25	120.27	5.58	11.32
30	$(2,4)$	56	17	91.25	14.24	6.26	11.72
30	$(4,2)$	29	30	73.75	115.42	5.08	16.24
30	$(4,4)$	46	20	82.50	38.88	4.75	12.41
30	$(3,3)$	54	16	87.50	70.18	4.02	7.65
30	$(3,5)$	23	29	65.00	145.15	3.55	14.00
30	$(5,3)$	48	20	85.00	121.44	5.79	15.08
30	$(5,5)$	29	20	61.25	152.07	5.59	18.42
40	$(2,2)$	34	35	86.25	100.42	2.82	7.93
40	$(2,4)$	52	23	93.75	29.95	1.32	2.78
40	$(4,2)$	33	30	78.75	117.55	3.79	12.82
40	$(4,4)$	42	25	83.75	21.08	3.29	11.63
40	$(3,3)$	53	17	87.50	78.99	3.30	7.94
40	$(3,5)$	31	21	65.00	83.76	3.69	12.82
40	$(5,3)$	45	24	86.25	154.59	3.73	11.95
40	$(5,5)$	32	16	60.00	60.80	4.90	11.34

$(5,5), (4,4), and (3,3)$, respectively, where the number of machines in the two centers is equal. Furthermore, the average consumed time, while solving the problems, decreases for $m_1 = m_2$ as m_1 increases (except $m_1 = m_2 = 2$). In the cases where m_1 is different from m_2, the problems become more easier to be solved. Consequently, the developed procedures perform satisfactorily when there is different number of machines in the two centers.

TABLE 6. Performance of the proposed procedures according to m_1, m_1 and $n \in \{50, 100, 150, 200\}$.

n	(m_1, m_2)	SR	SBB	$\%S$	MT	MG	$MaxG$
50	$(2,2)$	37	39	95.00	105.34	0.54	1.01
50	$(2,4)$	53	15	85.00	28.33	1.81	7.30
50	$(4,2)$	34	29	78.75	107.14	2.32	7.98
50	$(4,4)$	48	14	77.50	16.75	1.71	10.09
50	$(3,3)$	54	16	87.50	58.72	2.28	11.93
50	$(3,5)$	37	19	70.00	84.21	2.74	11.27
50	$(5,3)$	48	20	85.00	113.07	2.11	5.10
50	$(5,5)$	36	10	57.50	54.35	3.63	13.79
100	$(2,2)$	40	34	92.50	102.23	0.69	1.86
100	$(2,4)$	51	23	92.50	17.70	1.10	4.40
100	$(4,2)$	34	34	85.00	129.31	0.72	2.34
100	$(4,4)$	44	26	87.50	34.10	0.37	0.58
100	$(3,3)$	51	19	87.50	87.33	0.62	1.20
100	$(3,5)$	40	24	80.00	121.22	1.52	5.73
100	$(5,3)$	42	26	85.00	159.04	0.71	2.78
100	$(5,5)$	35	15	62.50	95.54	2.04	8.30
150	$(2,2)$	43	31	92.50	110.88	0.19	0.47
150	$(2,4)$	56	19	93.75	39.36	0.33	1.03
150	$(4,2)$	48	19	83.75	97.00	0.51	1.69
150	$(4,4)$	45	20	81.25	79.20	0.28	0.85
150	$(3,3)$	53	25	97.50	82.44	0.37	0.38
150	$(3,5)$	38	17	68.75	113.68	0.84	2.72
150	$(5,3)$	44	23	83.75	156.40	0.39	1.42
150	$(5,5)$	37	9	57.50	97.77	1.15	4.38
200	$(2,2)$	49	26	93.75	72.58	0.27	0.59
200	$(2,4)$	55	18	91.25	30.29	0.39	1.31
200	$(4,2)$	43	23	82.50	134.52	0.48	1.77
200	$(4,4)$	50	11	76.25	38.57	0.29	1.57
200	$(3,3)$	53	19	90.00	78.83	0.19	0.38
200	$(3,5)$	40	16	70.00	115.01	0.62	2.48
200	$(5,3)$	47	15	77.50	91.89	0.19	0.48
200	$(5,5)$	44	8	65.00	99.38	0.90	3.55

TABLE 7. Performance of the proposed procedures according to the balance rate r.

r	SR	SBB	UN	$\%S$	MT	MG	$MaxG$
0.25	127	28	5	96.88	46.08	1.00	3.37
0.30	112	29	19	88.13	78.32	1.68	6.29
0.50	450	240	270	71.88	78.68	3.66	18.42
0.60	200	75	45	85.94	90.13	2.63	15.08
0.83	49	50	61	61.88	85.81	2.86	12.41
1.00	723	570	307	80.81	99.06	2.08	16.24
1.20	70	68	22	86.25	211.68	1.34	9.02
1.67	181	97	42	86.88	40.04	0.82	5.05
2.00	584	302	74	92.29	70.42	1.38	8.96
3.33	146	13	1	99.38	13.09	0.62	0.62
4.00	144	16	0	100.00	0.88	0.00	0.00

In order to have a precise idea about the behavior of the proposed procedures, the Tables 5 and 6 are presented. These tables combines the effect of n, m_1, and m_2 simultaneously.

The provided results in Tables 5 and 6 confirm the drawn conclusions from Tables 4 and 3.

TABLE 8. Performance of the proposed procedures for transportation time distributions.

b	SR	SBB	UN	$\%S$	MT	MG	$MaxG$
20	1289	791	480	81.25	103.08	3.01	18.42
40	1497	697	366	85.70	56.70	1.86	12.41

TABLE 9. Performance of the proposed procedures for rates [*a* : *b* : *c*].

$[a:b:c]$	SR	SBB	UN	$\%S$	MT	MG	$MaxG$
[20 : 20 : 20]	280	256	104	83.75	91.39	2.51	16.24
[20 : 20 : 40]	228	191	221	65.47	117.28	4.04	18.42
[20 : 40 : 20]	424	157	59	90.78	32.57	0.88	5.05
[20 : 40 : 40]	355	156	129	79.84	41.00	2.29	12.41
[40 : 20 : 20]	414	183	43	93.28	102.53	1.22	9.02
[40 : 20 : 40]	367	161	112	82.50	104.30	2.13	15.08
[40 : 40 : 20]	390	199	51	92.03	69.49	1.72	8.96
[40 : 40 : 40]	328	185	127	80.16	84.97	1.94	11.76

TABLE 10. Performance of the proposed procedures for rate [20 : 20 : 20] and *n* ∈ {10, 20, 30, 40}.

		[20 : 20 : 20]					
n	(m_1, m_2)	SR	SBB	$\%S$	MT	MG	$MaxG$
10	(2, 2)	1	9	100	0.06	0.00	0.00
10	(2, 4)	9	1	100	0.00	0.00	0.00
10	(4, 2)	8	2	100	0.01	0.00	0.00
10	(4, 4)	6	4	100	0.00	0.00	0.00
10	(3, 3)	1	9	100	0.03	0.00	0.00
10	(3, 5)	6	4	100	0.00	0.00	0.00
10	(5, 3)	9	1	100	0.00	0.00	0.00
10	(5, 5)	10	0	100	0.00	0.00	0.00
20	(2, 2)	1	6	70	85.73	3.46	4.07
20	(2, 4)	6	4	100	0.00	0.00	0.00
20	(4, 2)	8	2	100	60.01	0.00	0.00
20	(4, 4)	2	6	80	228.85	6.07	6.25
20	(3, 3)	1	8	90	285.20	6.33	6.33
20	(3, 5)	6	3	90	94.40	1.15	1.15
20	(5, 3)	4	6	100	61.93	0.00	0.00
20	(5, 5)	1	7	80	437.38	6.97	7.27
30	(2, 2)	0	8	80	75.03	2.92	5.26
30	(2, 4)	9	1	100	0.01	0.00	0.00
30	(4, 2)	9	1	100	0.30	0.00	0.00
30	(4, 4)	1	5	60	102.25	4.15	4.71
30	(3, 3)	1	4	50	20.06	6.45	16.24
30	(3, 5)	5	5	100	60.23	0.00	0.00
30	(5, 3)	7	3	100	70.04	0.00	0.00
30	(5, 5)	1	2	30	202.25	4.27	7.04
40	(2, 2)	3	5	80	76.71	0.45	0.47
40	(2, 4)	8	2	100	0.01	0.00	0.00
40	(4, 2)	9	1	100	0.03	0.00	0.00
40	(4, 4)	2	3	50	125.43	4.05	9.60
40	(3, 3)	1	7	80	301.03	5.01	9.26
40	(3, 5)	5	5	100	0.09	0.00	0.00
40	(5, 3)	5	5	100	10.23	0.00	0.00
40	(5, 5)	1	2	30	1.40	4.51	7.77

In the sequel, the effect of the average load of the centers on the results will be presented. To that aim the following definitions and notations will be introduced.

TABLE 11. Performance of the proposed procedures for rate [20 : 20 : 20] and *n* ∈ {50, 100, 150, 200}.

		[20 : 20 : 20]					
n	(m_1, m_2)	SR	SBB	$\%S$	MT	MG	$MaxG$
50	(2, 2)	3	7	100	130.24	0.00	0.00
50	(2, 4)	8	2	100	60.03	0.00	0.00
50	(4, 2)	5	5	100	0.11	0.00	0.00
50	(4, 4)	0	3	30	427.59	2.67	5.07
50	(3, 3)	2	8	100	240.40	0.00	0.00
50	(3, 5)	7	2	90	12.24	0.52	0.52
50	(5, 3)	7	3	100	0.24	0.00	0.00
50	(5, 5)	0	1	10	224.49	2.84	7.96
100	(2, 2)	3	6	90	145.04	0.39	0.39
100	(2, 4)	9	1	100	0.06	0.00	0.00
100	(4, 2)	7	3	100	120.25	0.00	0.00
100	(4, 4)	0	7	70	423.73	0.60	1.08
100	(3, 3)	3	4	70	100.56	0.28	0.29
100	(3, 5)	6	3	90	3.63	0.27	0.27
100	(5, 3)	3	6	90	218.64	0.34	0.34
100	(5, 5)	0	5	50	399.40	1.55	3.32
150	(2, 2)	5	4	90	145.85	0.13	0.13
150	(2, 4)	9	1	100	76.00	0.00	0.00
150	(4, 2)	6	4	100	70.64	0.00	0.00
150	(4, 4)	3	2	50	144.34	0.35	0.51
150	(3, 3)	4	4	80	181.95	0.29	0.39
150	(3, 5)	7	2	90	1.49	0.20	0.20
150	(5, 3)	5	5	100	233.05	0.00	0.00
150	(5, 5)	4	1	50	218.29	1.15	2.55
200	(2, 2)	3	5	80	49.82	0.24	0.38
200	(2, 4)	6	4	100	61.33	0.00	0.00
200	(4, 2)	6	4	100	61.64	0.00	0.00
200	(4, 4)	1	5	60	230.67	0.32	0.72
200	(3, 3)	1	6	70	188.76	0.38	0.71
200	(3, 5)	5	5	100	107.80	0.00	0.00
200	(5, 3)	6	4	100	88.93	0.00	0.00
200	(5, 5)	0	3	30	213.53	0.37	0.70

- the average load A_i in center C_i $(i = 1, 2)$ is defined as: $A_i = \frac{E\left(\sum_{j=1}^{n} p_{i,j}\right)}{m_i} = \frac{na_i}{2m_i}$ with $p_{i,j}$ uniformly distributed in $[1, a_i]$.
- the balance rate between centers C_1 and C_2 is $r = \frac{A_1}{A_2} = \frac{am_2}{cm_1}$ (a and c are stated in the section "Test problems").
- In the case $r = 1$ the two centers are balanced.

The obtained values of r are 0.25, 0.30, 0.5, 0.6, 0.83, 1, 1.2, 1.67, 23.33, 4, and Table 7 exhibits the obtained results with respect to r.

Based on Table 7, the proposed procedures performs well in terms of percentage of solved problems, mean CPU time, and the relative gap, the average load in the first center is more important than the one in the second center: $r = 2$, $r = 3.33$, $r = 4$. In addition, from $r = 0.83$, the percentage of solved problems increases as $r =$ increases.

The effect of the transportation time on the performance of the proposed procedures is reported in Table 8

TABLE 12. Performance of the proposed procedures for rate [20 : 20 : 40] and $n \in \{10, 20, 30, 40\}$.

		[20 : 20 : 40]					
n	(m_1, m_2)	SR	SBB	$\%S$	MT	MG	$MaxG$
10	(2, 2)	1	9	100	0.04	0.00	0.00
10	(2, 4)	4	6	100	0.01	0.00	0.00
10	(4, 2)	8	2	100	0.00	0.00	0.00
10	(4, 4)	9	1	100	0.00	0.00	0.00
10	(3, 3)	4	6	100	0.01	0.00	0.00
10	(3, 5)	8	2	100	0.00	0.00	0.00
10	(5, 3)	8	2	100	0.00	0.00	0.00
10	(5, 5)	10	0	100	0.00	0.00	0.00
20	(2, 2)	0	7	70	233.37	3.16	4.63
20	(2, 4)	6	3	90	6.05	3.23	3.23
20	(4, 2)	5	4	90	77.79	3.37	3.37
20	(4, 4)	0	4	40	517.38	7.48	10.45
20	(3, 3)	0	5	50	505.79	5.70	17.11
20	(3, 5)	3	6	90	217.29	5.63	5.63
20	(5, 3)	3	3	60	121.39	2.59	4.05
20	(5, 5)	2	4	60	143.81	5.46	8.96
30	(2, 2)	0	4	40	223.76	7.79	11.32
30	(2, 4)	7	2	90	11.13	1.28	1.28
30	(4, 2)	6	3	90	133.46	0.63	0.63
30	(4, 4)	0	1	10	1.31	5.45	14.00
30	(3, 3)	0	2	20	302.26	7.34	15.57
30	(3, 5)	4	3	70	91.69	1.50	2.68
30	(5, 3)	4	3	70	272.12	3.35	4.50
30	(5, 5)	0	1	10	309.82	10.44	18.42
40	(2, 2)	2	4	60	116.91	5.32	7.93
40	(2, 4)	4	6	100	120.22	0.00	0.00
40	(4, 2)	6	2	80	87.55	0.46	0.48
40	(4, 4)	0	0	0	0.00	5.96	12.82
40	(3, 3)	1	2	30	2.92	6.46	12.82
40	(3, 5)	4	5	90	33.06	0.67	0.67
40	(5, 3)	4	4	80	323.35	3.48	6.29
40	(5, 5)	0	0	0	0.00	8.35	11.34

TABLE 13. Performance of the proposed procedures for rate [20 : 20 : 40] and $n \in \{50, 100, 150, 200\}$.

		[20 : 20 : 40]					
n	(m_1, m_2)	SR	SBB	$\%S$	MT	MG	$MaxG$
50	(2, 2)	1	6	70	292.96	0.66	1.01
50	(2, 4)	6	3	90	66.76	0.43	0.43
50	(4, 2)	10	0	100	0.02	0.00	0.00
50	(4, 4)	1	2	30	237.05	4.55	11.27
50	(3, 3)	0	1	10	171.99	3.55	7.98
50	(3, 5)	3	4	70	63.90	1.34	1.57
50	(5, 3)	6	3	90	140.22	1.99	1.99
50	(5, 5)	1	0	10	0.05	7.68	13.79
100	(2, 2)	1	5	60	317.56	0.91	1.86
100	(2, 4)	5	4	90	71.88	0.19	0.19
100	(4, 2)	7	3	100	120.23	0.00	0.00
100	(4, 4)	0	2	20	366.10	2.19	5.73
100	(3, 3)	1	4	50	418.33	1.17	2.34
100	(3, 5)	6	3	90	0.44	0.58	0.58
100	(5, 3)	4	4	80	180.28	0.43	0.55
100	(5, 5)	0	1	10	367.72	4.43	8.30
150	(2, 2)	1	7	80	318.52	0.42	0.47
150	(2, 4)	7	3	100	16.79	0.00	0.00
150	(4, 2)	5	5	100	133.81	0.00	0.00
150	(4, 4)	1	0	10	0.13	1.65	2.72
150	(3, 3)	1	3	40	456.86	0.88	1.69
150	(3, 5)	7	3	100	62.29	0.00	0.00
150	(5, 3)	3	2	50	23.94	0.27	0.38
150	(5, 5)	0	0	0	0.00	2.04	4.38
200	(2, 2)	3	4	70	219.00	0.29	0.59
200	(2, 4)	9	1	100	0.48	0.00	0.00
200	(4, 2)	6	3	90	134.92	0.09	0.09
200	(4, 4)	3	1	40	63.42	1.46	2.48
200	(3, 3)	1	3	40	404.37	0.71	1.77
200	(3, 5)	8	2	100	63.66	0.00	0.00
200	(5, 3)	5	3	80	241.11	0.22	0.30
200	(5, 5)	3	0	30	0.83	2.09	3.55

The Table 8 indicates that for $b = 40$ there is a slight advantage in terms of percentage of solved problems with 85.70% against 81.25% for $b = 20$. The difference becomes more clear when we come to the mean time solving the problems and the average gap, where for $b = 40$, $MT = 56.70s, MG = 3.01$ and for $b = 20$, $MT = 103.08s, MG = 1.86$. Therefore, the proposed procedures perform satisfactorily when the transportation time is more important.

The results reflecting the effects of the rates [a: b: c] on the performance of the proposed procedures is presented throughout the Table 9.

According to Table 9, the least performance of the proposed procedures is reached for the rates [20 : 20 : 40], where the percentage of solved problems is 65.47% (smallest one), the mean time solving problems $MT = 117.28s$ (greatest one) and $MG = 4.04$ (greatest one).

In the sequel, more detailed results for rates [20 : 20 : 20] and [20 : 20 : 40] are presented over Tables 10-13, respectively. These results show the effect of different parameters(n,m_1, m_2) on the performance of the proposed procedures for these particular rates([20 : 20 : 20] and [20 : 20 : 40]).

The presented results in Tables 10 and 13 confirm the conclusions already drawn from previous tables.

VII. CONCLUSION

In this paper, the two-center hybrid flow shop scheduling problem, with transportation times is considered. A set of efficient lower bounds is developed, in addition to a two phases heuristic. These lower and upper bounds rely on the exact solution of the parallel machine scheduling problem with release dates and delivery times. Furthermore, a branch and bound exact procedure is presented. This branch and bound algorithm includes fast and efficient lower bounds, performant heuristic, and dominance rules. In addition, within the exact procedure, the solutions are represented as a permutation of the jobs. Moreover, at the last level of the search tree a parallel machine scheduling problem is solved exactly. We observe that the obtained results are not impacted by the NP-Hardness of this embedded problem in the branch and bound procedure. Extensive experimental results are presented. These results provide strong evidence of the performance of the proposed procedures.

More researches are required to tackle the hybrid flow shop scheduling problem with more than two centers and with transportation times. Moreover, other constraints might be included as the setup times, release dates, delivery times in the future researches. These researches could use the presented procedures in this work as the solution of relaxed problems for the studied ones. In addition, to tackle the existing scheduling problem other technics could be used. Among these technics the well-known hybrid switching control or switching LPV control approach that can be employed in the future to deal with the scheduling issue on basis of the latest work, such as in [33] and [34].

REFERENCES

[1] J. N. Gupta, "Two-stage, hybrid flowshop scheduling problem," *J. Oper. Res. Soc.*, vol. 39, no. 4, pp. 359–364, Apr. 1988.

[2] S. L. Narasimhan and S. S. Panwalker, "Scheduling in a two-stage manufacturing process," *Int. J. Prod. Res.*, vol. 22, no. 4, pp. 555–564, Mar. 1984.

[3] Z. H. Jin, K. Ohno, T. Ito, and S. E. Elmaghraby, "Scheduling hybrid flowshops in printed circuit board assembly lines," *Prod. Oper. Manage.*, vol. 11, no. 2, pp. 216–230, Jun. 2002.

[4] S. E. Elmaghraby and R. E. Karnoub, "Production control in hybrid flowshops: An example from textile manufacturing," in *The Planning and Scheduling of Production Systems: Methodologies and Applications*, A. Artiba and S. E. Emaghraby, Eds. London, U.K.: Chapman & Hall, 1997, pp. 163–198.

[5] R. Ruiz and J. A. Vázquez-Rodríguez, "The hybrid flow shop scheduling problem," *Eur. J. Oper. Res.*, vol. 205, no. 1, pp. 1–18, Aug. 2010.

[6] I. Ribas, R. Leisten, and J. M. Framinan, "Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective," *Comput. Oper. Res.*, vol. 37, no. 8, pp. 1439–1454, Aug. 2010.

[7] D. A. Wismer, "Solution of the flowshop-scheduling problem with no intermediate queues," *Oper. Res.*, vol. 20, no. 3, pp. 689–697, Jun. 1972.

[8] C. Rajendran, "A no-wait flowshop scheduling heuristic to minimize makespan," *J. Oper. Res. Soc.*, vol. 45, no. 4, pp. 472–478, Apr. 1994.

[9] N. Javadian, P. Fattahi, M. Farahmand-Mehr, M. Amiri-Aref, and M. Kazemi, "An immune algorithm for hybrid flow shop scheduling problem with time lags and sequence-dependent setup times," *Int. J. Adv. Manuf. Technol.*, vol. 63, nos. 1–4, pp. 337–348, Nov. 2012.

[10] V. Botta-Genoulaz, "Hybrid flow shop scheduling with precedence constraints and time lags to minimize maximum lateness," *Int. J. Prod. Econ.*, vol. 64, nos. 1–3, pp. 101–111, Mar. 2000.

[11] R. Ruiz, F. S. Şerifoğlu, and T. Urlings, "Modeling realistic hybrid flexible flowshop scheduling problems," *Comput. Oper. Res.*, vol. 35, no. 4, pp. 1151–1175, Apr. 2008.

[12] P. Brucker, *Scheduling Algorithms*. Berlin, Germany: Springer-Verlag, 1998.

[13] J. N. D. Gupta and E. A. Tunc, "Scheduling a two-stage hybrid flowshop with separable setup and removal times," *Eur. J. Oper. Res.*, vol. 77, no. 3, pp. 415–428, Sep. 1994.

[14] M. Haouari and R. M'Hallah, "Heuristic algorithms for the two-stage hybrid flowshop problem," *Oper. Res. Lett.*, vol. 21, no. 1, pp. 43–53, Aug. 1997.

[15] M. Haouari, L. Hidri, and A. Gharbi, "Optimal scheduling of a two-stage hybrid flow shop," *Math. Methods Oper. Res.*, vol. 64, no. 1, pp. 107–124, Aug. 2006.

[16] O. Engin, G. Ceran, and M. K. Yilmaz, "An efficient genetic algorithm for hybrid flow shop scheduling with multiprocessor task problems," *Appl. Soft Comput.*, vol. 11, no. 3, pp. 3056–3065, Apr. 2011.

[17] M. E. Kurz and R. G. Askin, "Scheduling flexible flow lines with sequence-dependent setup times," *Eur. J. Oper. Res.*, vol. 159, no. 1, pp. 66–82, Nov. 2004.

[18] B. Naderi, M. Zandieh, A. K. G. Balagh, and V. Roshanaei, "An improved simulated annealing for hybrid flowshops with sequence-dependent setup and transportation times to minimize total completion time and total tardiness," *Expert Syst. Appl.*, vol. 36, no. 6, pp. 9625–9633, Aug. 2009.

[19] B. Naderi, M. Zandieh, and M. A. H. A. Shirazi, "Modeling and scheduling a case of flexible flowshops: Total weighted tardiness minimization," *Comput. Ind. Eng.*, vol. 57, no. 4, pp. 1258–1267, Nov. 2009.

[20] D. R. Sule, *Industrial Scheduling*. Boston, MA, USA: PWS Publishing Company, 1996.

[21] S. S. Zabihzadeh and J. Rezaeian, "Two meta-heuristic algorithms for flexible flow shop scheduling problem with robotic transportation and release time," *Appl. Soft Comput.*, vol. 40, pp. 319–330, Mar. 2016.

[22] W.-Y. Zhong and L.-H. Lv, "Hybrid flowshop scheduling with interstage job transportation," *J. Oper. Res. Soc. China*, vol. 2, no. 1, pp. 109–121, Mar. 2014.

[23] H. Zhu, "A two stage scheduling with transportation and batching," *Inf. Process. Lett.*, vol. 112, no. 19, pp. 728–731, Oct. 2012.

[24] A. Elmi and S. Topaloglu, "A scheduling problem in blocking hybrid flow shop robotic cells with multiple robots," *Comput. Oper. Res.*, vol. 40, no. 10, pp. 2543–2555, Oct. 2013.

[25] N. Chikhi, M. Abbas, R. Benmansour, A. Bekrar, and S. Hanafi, "A two-stage flow shop scheduling problem with transportation considerations," *4OR*, vol. 13, no. 4, pp. 381–402, Dec. 2015.

[26] A. Gharbi and M. Haouari, "Minimizing makespan on parallel machines subject to release dates and delivery times," *J. Scheduling*, vol. 5, no. 4, pp. 329–355, Jul. 2002.

[27] C.-Y. Lee and G. L. Vairaktarakis, "Minimizing makespan in hybrid flowshops," *Oper. Res. Lett.*, vol. 16, no. 3, pp. 149–158, Oct. 1994.

[28] A. Gharbi and M. Haouari, "Optimal parallel machines scheduling with availability constraints," *Discrete Appl. Math.*, vol. 148, no. 1, pp. 63–87, Apr. 2005.

[29] A. Gharbi and M. Haouari, "Optimal parallel machines scheduling with initial and final availability constraints," in *Proc. 9th Int. Workshop Project Manage. Scheduling PMS*, 2004, pp. 218–221.

[30] L. Hidri and A. Gharbi, "New efficient lower bound for the hybrid flow shop scheduling problem with multiprocessor tasks," *IEEE Access*, vol. 5, pp. 6121–6133, Apr. 2017.

[31] M. L. R. Varela, J. Trojanowska, S. Carmo-Silva, N. M. L. Costa, and J. Machado, "Comparative simulation study of production scheduling in the hybrid and the parallel flow," *Manage. Prod. Eng. Rev.*, vol. 8, no. 2, pp. 69–80, Jun. 2017.

[32] J. Costa and M. L. R. Varela, "Decision system for supporting the implementation of a manufacturing section on an automotive factory in Portugal," in *Proc. Online Trends Innov. Comput. (ICT)*, 2012, pp. 90–95.

[33] Y. Zhu, Z. Zhong, W. X. Zheng, and D. Zhou, "HMM-based H∞ filtering for discrete-time Markov jump LPV systems over unreliable communication channels," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, doi: 10.1109/TSMC.2017.2723038.

[34] L. Zhang, Y. Zhu, and W. X. Zheng, "Synchronization and state estimation of a class of hierarchical hybrid neural networks with time-varying delays," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 2, pp. 459–470, Feb. 2016.

LOTFI HIDRI received the bachelor's degree in mathematics from the Tunisian College of Science in 1993, the master's degree in energetic engineering from the National Engineering School in 1999, and the Ph.D. degree in operations research from the Tunisian High Institute of Management in 2007. He is currently a Faculty Member with the Industrial Engineering Department, King Saud University. His main research interests include scheduling and transportation.

SABEUR ELKOSANTINI received the M.Sc. degree in computer science from Lumière University Lyon 2, France, in 2004, and the Ph.D. degree in computer science from Blaise Pascal University, France, in 2008. He is also involved in many research projects. His main research interests include the development of new innovative models and intelligent approaches for reactive decision-making for logistics and transportation systems.

MOHAMMED M. MABKHOT received the B.Sc. degree in industrial engineering from King Khalid University, Abha, Saudi Arabia, and the M.Sc. degree in industrial engineering from King Saud University, Riyadh, Saudi Arabia, where he is currently pursuing the Ph.D. degree in industrial engineering. His master's thesis was on the development of knowledge-based decision support systems for disruption and risks management in reconfigurable manufacturing system. His research interests include the management of smart manufacturing systems to response to unpredictable change in customer demands and to failures of production resources using artificial intelligence.