Solving a dock assignment problem as a three-stage flexible flow-shop problem

L. Berghman, R. Leus
ORSTAT, K.U.Leuven, Leuven, Belgium

Abstract - **This paper presents a model for a dock assignment problem based on the situation encountered in a practical case. Trailers are assigned to gates during a specific period in time for loading or unloading activities. The parking lot is used as a buffer zone. Transportation between the parking lot and the gates is performed by additional resources called terminal tractors. The problem is modeled as a three-stage flexible flow shop, where the first and the third stage share the same identical parallel machines and besides that, all stages share a different set of identical parallel machines. Different mathematical formulations are given and a Lagrangian relaxation approach is examined to solve this flexible flow-shop problem.**

Keywords - **dock assignment, flexible flow shop, additional resources, Lagrangian relaxation**

I. Introduction

Toyota is one of the world's largest automobile manufacturers, selling over 7.5 million models (including Hino and Daihatsu) annually on all five continents, and generating almost 130 billion euro in net revenues. Since 1999, the total warehouse space floor of the European Distribution Centre TPCE (Toyota Parts Centre Europe) located in Diest (Belgium) has been expanded to 67.700 m^2. Toyota's Distribution Centre delivers to 28 European distributors on a daily basis.

At TPCE, the warehouse has some fifty gates, each with a capacity of one trailer, where goods can either be loaded on an empty trailer or be unloaded from a loaded trailer. Besides the warehouse with the gates, the site also contains two parking lots, which can be seen as a buffer where trailers can be temporarily parked, both before and after the loading and unloading. All transportation activities of trailers between these parking lots and the gates are done by two terminal tractors, which are tractors designed for use in ports, terminals and heavy industry. Because of the numerous loading, unloading and transportation activities, TPCE needs a schedule specifying the starting time and the assigned gate or terminal tractor for each activity.

The remainder of this paper is structured as follows: the situation at TPCE will be modeled as a three-stage flexible flow shop. This is described in detail in Section II. A brief review of the relevant literature is given in Section III. In Section IV, some mathematical formulations are given. And finally, a Lagrangian relaxation approach is discussed in Section V.

II. Flexible flow-shop scheduling

Each job is composed of three tasks, one for each stage. The fist stage consists of one of the terminal tractors moving the trailer to the gate, while the second stage consists of the loading and unloading activities. The different gates at TPCE are considered to be identical parallel machines and every task of stage two has to be assigned to one of the parallel machines. The third stage is one of the terminal tractors that moves the trailer back to the parking lot. Two identical parallel machines execute both the first and the third stage. For safety reasons, we consider the gate on which the corresponding loading or unloading activity of stage two is planned as being unavailable during the execution of the transportation activity. In these stages, the processing times are not dependent on the distance because the actual driving time of the terminal tractors is small compared to the time it takes the driver to follow the safety instructions and attach the trailer to the tractor. Some fifty identical parallel machines execute the second stage, so the processing time depends only on the job and is independent of the machine.

For a given job, stage two always starts immediately after stage one finishes; the same does not hold for stage three. After loading or unloading, a trailer cannot be transported to the parking lot by a terminal tractor if both tractors are busy. The trailer must remain on the gate until a terminal tractor becomes available, which makes it impossible for another trailer to be loaded or unloaded there.

The facilities are disjunctive, in the sense that each machine may process only one task at a time. Preemption of a task is not allowed and no machine has buffer storage for work-in-process.

A. Definition and detailed problem statement

T, the set of all tasks t with $|T| = 3n$, can be partitioned into different subsets. A first partition of T is T_1, T_2, T_3. $T_1 = \{t_1^{(1)}, t_2^{(1)}, \ldots, t_n^{(1)}\} \subset T$ contains all transportation activities from the parking lot to a gate, $T_2 = \{t_1^{(2)}, t_2^{(2)}, \ldots, t_n^{(2)}\} \subset T$ contains all loading and unloading activities and $T_3 = \{t_1^{(3)}, t_2^{(3)}, \ldots, t_n^{(3)}\} \subset T$ contains all transportation activities from a gate to the parking lot. T_i^U, T_i^L is a partition of T_i for $i = 1, 2, 3$. $T^U = T_1^U \cup T_2^U \cup T_3^U$ is the set of tasks related to an unloading activity, while $T^L = T_1^L \cup T_2^L \cup T_3^L$ is the set of tasks related to a loading activity.

All tasks $t \in T$ have a ready time r_t. For the unloading tasks $t \in T^U$, this ready time equals the planned arrival time of the trailer. For the loading tasks $t \in T^L$, $r_t = 0$ because we assume that all the goods to be loaded on the trailers are available in the warehouse and the empty trailer is standing at the parking lot. All tasks $t \in T$ have a weight w_t. All

978-1-4244-4870-8/09/$26.00 ©2009 IEEE

unloading tasks $t \in T^U$ have a due date d_t equal to the ready time because the aim is to avoid tardiness. All loading tasks $t \in T^L$ have a deadline $\overline{d}_t$ based on the driving time to the customer and the agreed arrival time at the customer. All transportation activities between the parking lot and the gates have a constant duration C, independent of the distance. There are two machines available for both the first and the third stage. G is the set of all identical parallel machines g of stage two, with $|G| = m < n$. Each machine can process at most one job at a time. The processing time p_t of a job $t \in T$ is the time needed to load or unload the trailer at the gate.

Our problem consists of scheduling the tasks in such a way that the total weighted lateness (or tardiness, since $d_t = r_t$) of the unloading tasks is minimized and the total weighted earliness of the transportation activities back to the gate that are related to a loading task is maximized.

In Table I, data for a problem with ten jobs, four gates and one terminal tractor are given. A feasible schedule for this problem is presented in Fig. 1. The colored blocks represent the transportation activities by the terminal tractor and the white blocks represent the loading or unloading activities at the gates.

Task	w_t	r_t	p_t	d_t	$\overline{d}_t$	type
t_1	3	0	110	0		U
t_2	1	0	140	0		U
t_3	1	10	150	10		U
t_4	3	20	130	20		U
t_5	2	50	120	50		U
t_6	1	0	120		250	L
t_7	3	0	100		200	L
t_8	2	0	130		320	L
t_9	1	0	120		130	L
t_{10}	3	0	130		130	L

TABLE I
A PROBLEM WITH TEN TASKS, FOUR GATES AND ONE TERMINAL TRACTOR

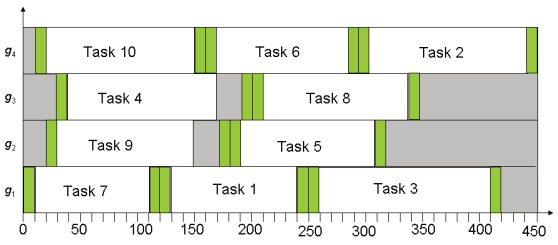

Fig. 1. A feasible schedule

III. LITERATURE REVIEW

The goal of this article is to examine mathematical programming formulations for the practical problem and to implement a Lagrangian relaxation approach to obtain high-quality solutions in reasonable computation time. We briefly survey related work on mathematical formulations for machine scheduling problems. Subsequently, we review the application of Lagrangian relaxation in machine scheduling problems.

A. Mathematical formulations

A review of parallel machine scheduling literature is given in Cheng and Sin [1] and a survey of mathematical programming formulations for machine scheduling, including parallel machine scheduling, can be found in Blazewicz et al. [2]. Different examples of continuous-time formulations can be found. For uniform parallel machine scheduling, Dessouky [3] examines an assignment-based integer non-linear programming formulation where jobs are assigned to positions on machines and Jain and Grossmann [4] propose an assignment-based integer linear programming formulation where jobs are assigned to machines and additional variables are used to determine the sequence on each machine. Bard and Rojanasoonthon [5] present a flow formulation for identical parallel machine scheduling while Mellouli et al. [6] study different models including a flow formulation, a sequence-based formulation and an assignment-based formulation. Time-indexed formulations have also received a great deal of attention because the linear programming relaxations provide strong lower bounds. Luh et al. [7], Sousa and Wolsey [8] and Kedad-Sidhoum et al. [9] examine formulations for a single machine problem that can easily be extended to a parallel machine scheduling problem. The binary decision variables assign each job to a certain starting period.

A survey of flexible flow-shop literature can be found in Linn and Zhang [10]. Most studies deal with two-stage flow shops with parallel machines either in the first or in the second stage, but not in both. To the best of our knowledge, very few examples of tight continuous-time formulations can be found in literature. Riane et al. [11] propose a sequence-based formulation, while Paternina-Arboleda et al. [12] present an assignment-based formulation and Jungwattanakit et al. [13] describe a flow formulation. Concerning the time-indexed formulations, Chen and Luh [14] consider a job shop scheduling problem consisting of scheduling different parts on different types of machines where each machine type has different identical machines and the completion of each part requires a series of different operations. Tang et al. [15] and Tang and Xuan [16] study mathematical formulations for the scheduling problem with multiple identical parallel machines available for processing jobs at each stage.

B. Lagrangian relaxation for machine scheduling problems

Most articles in the machine-scheduling literature that make use of Lagrangian relaxation, relax the capacity constraints on machines of their time-indexed integer programming formulation by using Lagrange multipliers and decompose the relaxed problem into independent job level subproblems (parallel machine scheduling: Luh et al. [7]. Kedad-Sidhoum et al. [9]; job-shop or flow-shop scheduling: Chen et al. [17] and Tang and Xuan [16]), although in flow-shop or job-shop scheduling it might be that the precedence constraints are relaxed (Chen and Luh [14], Tang et al. [15]). Kedad-Sidhoum et al. [9] also relax the number of occurrences, such that a job can be processed several times. The Lagrange multipliers are updated by a (surrogate) subgradient method. When the dual solution

at the end of the algorithm is infeasible, a feasible schedule is most of the time constructed by a heuristic list-scheduling approach.

IV. Mathematical formulations

Since the available literature does no contain much information dedicated to mathematical formulations for flexible flow-shop problems, we start by testing formulations for only part of our problem. When we only look at the gate assignment part of the problem and do not take the terminal tractors into account, the problem at hand is close to $P_m|r_j, d_j|\sum w_j T_j$ in the standard three-field notation, although some tasks have deadlines rather than due dates.

Based on preliminary experiments, we have found that out of different formulations, the time-indexed formulation described in Subsection C gives the best results. Therefore, this formulation is extended to the flexible flow-shop scheduling problem in Subsection D.

A. Assignment-based formulation for the parallel machine scheduling problem

The decision variables of this first formulation are the following. For all tasks $t \in T$ and for every gate $g \in G$,

$$x_{tg}^{(2)} = \begin{cases} 1 & \text{if task } t \text{ is scheduled at gate } g, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

For all tasks $t, u \in T$,

$$z_{tu} = \begin{cases} 1 & \text{if task } t \text{ is scheduled before task } u \text{ when both} \\ & \text{tasks are executed at the same gate,} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

For every task $t \in T$ we also have a starting time s_t.
A mathematical formulation of our problem is the following:

$$\min \sum_{t \in T^U} w_t(s_t + p_t - d_t) + \sum_{t \in T^L} w_t(s_t + p_t - \overline{d}_t) \tag{3}$$

subject to

$$\sum_{g \in G} x_{tg}^{(2)} = 1 \qquad \forall t \in T \tag{4}$$

$$x_{tg}^{(2)} + x_{ug}^{(2)} - z_{tu} - z_{ut} \leq 1 \qquad \forall \{t, u\} \in T; \forall g \in G \tag{5}$$

$$s_t + p_t - (1 - z_{tu})M \leq s_u \qquad \forall \{t, u\} \in T \tag{6}$$

$$r_t \leq s_t \qquad \forall t \in T \tag{7}$$

$$s_t + p_t \leq \overline{d}_t \qquad \forall t \in T^L \tag{8}$$

$$x_{tg}^{(2)} \in \{0, 1\} \qquad \forall t \in T; \forall g \in G \tag{9}$$

$$z_{tu} \in \{0, 1\} \qquad \forall t, u \in T \tag{10}$$

$$s_t \geq 0 \qquad \forall t \in T \tag{11}$$

The objective function (3) minimizes for all unloading tasks $t \in T_2^U$ the total time between the finishing time $s_t + p_t$ of the unloading task and the due date d_t and for all loading tasks $t \in T_2^L$ the total time between the the finishing time $s_t + p_t$ of the loading task and the deadline $\overline{d}_t$. Constraint (4) limits each loading or unloading task to be assigned to exactly one gate, while constraint (5) ensures that if task t and task u are assigned to the same gate g, then one must be processed before the other. Constraint (6) ensures that the ending time of a certain task is smaller than the starting time of the following task on that same gate, where M stands for a large value. A possible value for M, which will be used in the implementations, is $\max_t r_t + \sum_{t \in T_2} p_t$. Constraints (7) and (8) enforce the time windows: no tasks can start before its ready time and all loading tasks have to be finished before their deadline. Finally, constraints (9) and (10) state that our decision variables $x_{tg}^{(2)}$ and z_{tu} are binary variables and constraint (11) states that all starting times have to be positive.

For a scheduling problem on identical parallel machines, permuting the machine indices does not affect the structure of the problem. For example, switching the assignment of all jobs assigned to machine 1 and those assigned to machine 2 will result in essentially the same solution. Due to this structure, branching becomes ineffective because many assignments of values to variables represent the same solution. For this reason, different symmetry-breaking constraints have been implemented and compared. Moreover, different valid inequalities have been tested to tighten and speed-up the formulation. Preliminary experiments show that adding the following constraints results in less computational effort.

$$\sum_{g=1}^{t} x_{tg}^{(2)} = 1 \qquad \forall t \in \{1, ..., m\} \tag{12}$$

$$z_{tu} + z_{ut} \leq 1 \qquad \forall \{t, u\} \in T \tag{13}$$

$$x_{tg}^{(2)} + x_{uh}^{(2)} + z_{tu} + z_{ut} \leq 2 \qquad \forall \{t, u\} \in T; \forall \{g, h\} \in G \tag{14}$$

Constraint (12) is a symmetry-breaking constraint that states that for $t \leq m$, task t is scheduled on one of the machines $1, \ldots, t$. Constraints (13) and (14) are valid inequalities. The first one states that for every pair of tasks, either they are not scheduled on the same gate, or one task comes before the other or the other way around. The second one guarantees that the sequencing variables z_{tu} and z_{ut} are both zero if tasks t and u are assigned to different gates.

This formulation has much in common with the ones presented in Jain and Grossman [4] although in those formulations the parallel machines are not identical such that the processing times depend on the machine and the objective is to minimize the sum of the processing costs c_j^m of assigning task j to machine m. Constraint (14) stems from Zhu and Heady [18].

B. Flow-based formulation for the parallel machine scheduling problem

The second mathematical formulation is based on Bard and Rojanasoonthon [5]. The main differences with our setting are the non-identical parallel machines, the two time windows per task, the setup times, the priority classes containing the tasks and the corresponding contributions. The decision variables of

this formulation are the following. For all tasks $t, u \in T$,

$$x_{tu}^{(2)} = \begin{cases} 1 & \text{if task } t \text{ is scheduled immediately before task } u, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

A mathematical formulation of our problem is the following:

$$\min \sum_{t \in T^U} w_t(s_t + p_t - d_t) + \sum_{t \in T^L} w_t(s_t + p_t - \overline{d}_t) \tag{16}$$

subject to

$$\sum_{u \in T^0 \setminus \{t\}} x_{tu}^{(2)} = 1 \qquad \forall t \in T \tag{17}$$

$$\sum_{t \in T} x_{0t}^{(2)} = m \tag{18}$$

$$\sum_{u \in T^0 \setminus \{t\}} x_{ut}^{(2)} - \sum_{u \in T^0 \setminus \{t\}} x_{tu}^{(2)} = 0 \qquad \forall t \in T^0 \tag{19}$$

$$s_t + p_t - (1 - x_{tu}^{(2)})M \leq s_u \qquad \forall \{t, u\} \in T \tag{20}$$

$$r_t \leq s_t \qquad \forall t \in T \tag{21}$$

$$s_t + p_t \leq \overline{d}_t \qquad \forall t \in T^L \tag{22}$$

$$x_{tu}^{(2)} \in \{0, 1\} \qquad \forall t, u \in T^0 \tag{23}$$

$$s_t \geq 0 \qquad \forall t \in T \tag{24}$$

Constraint (17) limits each task to be processed exactly once. It also ensures that, if a task is scheduled, it has no more than one successor, which might be the dummy task t_0. Constraint (18) limits the number of initial tasks. These constraints (17) and (18) indirectly specify that each machine can process at most one task at a time. Constraint (19) ensures the conservation of flow. If task t is assigned to gate g, both its predecessor and successor must be processed by gate g. Constraint (20) ensures that for a given gate, the ending time of a certain task is not longer than the starting time of the following task. As before, M stands for a large value, and a possible value for M equals $\max_t r_t + \sum_{t \in T} p_t$. This is the value we will use in the implementation. Constraints (21) and (22) enforce the time windows. Finally, constraint (23) states that our decision variables $x_{tu}^{(2)}$ are binary variables and constraint (24) states that all starting times have to be positive.

The symmetry problem is solved by producing a set of m schedules instead of assigning particular schedules to particular gates.

C. Time-indexed formulation for the parallel machine scheduling problem

This type of formulation is based on time-discretization where time is divided into periods. Let H denote the scheduling horizon, thus we consider the time periods $0, \ldots, H_{\max}$ with $H_{\max} = \max_t r_t + \sum_{t \in T} p_t$. This mathematical formulation is based on Sousa and Wolsey [8] and Kedad-Sidhoum et al. [9]. For all tasks $t \in T$ and for all periods $u \in H_t^{(2)}$,

$$x_{tu}^{(2)} = \begin{cases} 1 & \text{if task } t \text{ starts in time period } u, \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

with $H_t^{(2)} = r_t, \ldots, H_{\max}$ if $t \in T^U$ and $H_t^{(2)} = r_t, \ldots, \overline{d}_t - p_t$ if $t \in T^L$.

A mathematical formulation of our problem is the following:

$$\begin{aligned} \min \sum_{t \in T^U} w_t\Big(\sum_{u \in H_t} (u x_{tu}^{(2)}) + p_t - d_t\Big) \\ + \sum_{t \in T^L} w_t\Big(\sum_{u \in H_t} (u x_{tu}^{(2)}) + p_t - \overline{d}_t\Big) \end{aligned} \tag{26}$$

subject to

$$\sum_{u \in [r_t, H_{\max} - p_t]} x_{tu}^{(2)} = 1 \qquad \forall t \in T \tag{27}$$

$$\sum_{t \in T_2} \sum_{v \in [\max\{r_t, u - p_t\}, u]} x_{tv}^{(2)} \leq m \qquad \forall u \in H_t \tag{28}$$

$$x_{tu}^{(2)} \in \{0, 1\} \qquad \forall t \in T; \forall u \in H_t \tag{29}$$

The objective function (26) minimizes for all unloading tasks $t \in T^U$ the total time between the finishing time $\sum_{u \in H_t^{(2)}} u x_{tu}^{(2)} + p_t$ of the unloading task and the due date d_t and for all loading tasks $t \in T^L$ the total time between the finishing time $\sum_{u \in H_t^{(2)}} u x_{tu}^{(2)} + p_t$ of the loading task and the deadline $\overline{d}_t$. Constraint (27) limits each task to be processed exactly once. Constraint (28) ensures that for a given time interval u, only m unloading or loading activities can be executed. Finally, constraint (29) states that our decision variables $x_{tu}^{(2)}$ are binary variables.

As in Section IV-B, the symmetry problem is solved by producing a set of m schedules.

D. Time-indexed formulation for the flexible flow-shop problem

When extending the above formulation to our flexible flow-shop problem, $H_{max} = \max_t r_t + \sum_{t \in T} p_t + 2nC$. The decision variables are the following. For all task $t \in T_i$ (for $i = 1, 2, 3$) and for all moments in time $u \in H_t^{(i)}$,

$$x_{tu}^{(i)} = \begin{cases} 1 & \text{if task } t \text{ starts in time period } u, \\ 0 & \text{otherwise.} \end{cases} \tag{30}$$

with $H_t^{(1)} = \{r_t, \ldots, T - p_t - 2C\}$ (with $T = \max_t r_t + \sum_{t \in T_2} p_t + 2nC$), $H_t^{(2)} = \{r_t + C, \ldots, T - p_t - C\}$ and $H_t^{(3)} = \{r_t + C + p_t, \ldots, T - C\}$ if $t \in T^U$ and $H_t^{(1)} = \{r_t, \ldots, \overline{d}_t - p_t - 2C\}$, $H_t^{(2)} = \{r_t + C, \ldots, \overline{d}_t - p_t - C\}$ and $H_t^{(3)} = \{r_t + C + p_t, \ldots, \overline{d}_t - C\}$ if $t \in T^L$. A mathematical formulation of our problem is the following:

$$\min \sum_{t\in T^U} w_t \left(\left(\sum_{u\in H_t^{(2)}} u x_{tu}^{(2)} \right) + p_t - d_t \right) + \sum_{t\in T^L} w_t \left(\left(\sum_{u\in H_t^{(3)}} u x_{tu}^{(3)} \right) + C - \overline{d}_t \right) \tag{31}$$

subject to

$$\sum_{u\in H_t^{(k)}} x_{tu}^{(k)} = 1 \qquad \forall t \in T_k; k = 1,2,3 \tag{32}$$

$$\sum_{t\in T} \left(x_{tu}^{(1)} + x_{tu}^{(3)} + \sum_{v\le u} \left(x_{tv}^{(2)} - x_{tv}^{(3)} \right) \right) \le m \qquad \forall u \in H \tag{33}$$

$$\sum_{t\in T} \left(x_{tu}^{(1)} + x_{tu}^{(3)} \right) \le 2 \qquad \forall u \in H \tag{34}$$

$$x_{tu}^{(1)} = x_{t,u+C}^{(2)} \qquad \forall t; \forall u \in H \tag{35}$$

$$x_{tu}^{(2)} \le \sum_{v=u+p_t}^{T} x_{t,v}^{(3)} \qquad \forall t; \forall u \in H \tag{36}$$

$$x_{tu}^{(i)} \in \{0,1\} \qquad \text{for } i = 1,2,3; \forall t \in T_i; \forall u \in H_t^{(i)} \tag{37}$$

The objective function (31) minimizes for all unloading tasks $t \in T^U$ the total time between the finishing time $\sum_{u\in H_t} u x_{tu}^{(2)} + p_t$ of the unloading task and the due date d_t and for all loading tasks $t \in T^L$ the total time between the the finishing time $\sum_{u\in H_t} u x_{tu}^{(3)} + C$ of the transportation task back to the parking lot and the deadline $\overline{d}_t$. In this way, we achieve that all import goods are as early as possible in the warehouse and all export trailers are ready as early as possible. Constraint (32) limits each task to be processed exactly once either on a gate or by a terminal tractor. Constraint (33) ensures that for a given time interval, only m unloading, loading or transportation activities can be executed. Constraint (34) ensures the capacity of the terminal tractors. Constraints (35) and (36) ensure that for each job the first task has to be finished before the second task can start and the second task has to be finished before the third task can start. Finally, constraint (37) states that our decision variables $x_{tu}^{(i)}$ (for $i = 1,2,3$) are binary variables.

V. Lagrangian relaxation

For some instances with 34 tasks, nine gates and one terminal tractor, the computation times for the time-indexed formulation exceed two hours. Note that a higher number of terminal tractors (even two) significantly decreases the required computational effort. In order to produce near-optimal solutions within reasonable running time for practical-size instances, we resort to Lagrangian relaxation.

Both the resource constraints of the time-indexed formulation (constraints (33) and (34)) are relaxed using Lagrange multipliers, such that easily solvable, independent job-level subproblems are obtained. The Lagrange multipliers act as prices to regulate the use of the machines. For each task, the starting time to strike the best balance between these machine prices and the tardiness / earliness of the job is selected. The Lagrange multipliers are updated according to the demand of the machines, by a subgradient method.

When the dual solution at the end of the algorithm is infeasible, a feasible schedule is constructed by a two-phase heuristic approach based on list scheduling, which uses the dual solution, and a heuristic to take the deadlines into account (see [16], for example).

References

[1] T. Cheng and C. Sin, "A state-of-the-art review of parallel-machine scheduling research," *European Journal of Operational Research*, vol. 47, pp. 271–292, 1990.

[2] J. Blazewicz, M. Dror, and J. Weglarz, "Mathematical programming formulations for machinery scheduling: A survey," *European Journal of Operational Research*, vol. 51, pp. 283–300, 1991.

[3] M. Dessouky, "Scheduling identical jobs with unequal ready times on uniform parallel machines to minimize the maximum lateness," *Computers and Industrial Engineering*, vol. 34 (4), pp. 793–806, 1998.

[4] V. Jain and I. Grossmann, "Algorithms for hybrid MILP/CP models for a class of optimization problems," *INFORMS Journal on Computing*, vol. 13 (4), pp. 258–276, 2001.

[5] J. Bard and S. Rojanasoonthon, "A branch-and-price algorithm for parallel machine scheduling with time windows and job priorities," *Naval Research Logistics*, vol. 53, pp. 24–44, 2006.

[6] R. Mellouli, C. Sadfi, C. Chu, and I. Kacem, "Identical parallel-machine scheduling under availability constraints to minimize the sum of completion times," *European Journal of Operational Research*, vol. 197, pp. 1150–1165, 2009.

[7] P. Luh, D. Hoitomt, E. Max, and K. Pattipati, "Schedule generation and reconfiguration for parallel machines," *IEEE Transactions on Robotics and Automation*, vol. 6, pp. 687–696, 1990.

[8] J. Sousa and L. Wolsey, "A time indexed formulation of non-preemptive single machine scheduling problems," *Mathematical Programming*, vol. 54, pp. 353–367, 1992.

[9] S. Kedad-Sidhoum, Y. R. Solis, and F. Sourd, "Lower bounds for the earliness-tardiness scheduling problem on parallel machines with distinct due dates," *European Jounal of Operational Research*, vol. 189, pp. 1305–1316, 2008.

[10] R. Linn and W. Zhang, "Hybrid flow shop scheduling: a survey," *Computers and Industrial Engineering*, vol. 37, pp. 57–61, 1999.

[11] F. Riane, A. Artiba, and S. Elmaghraby, "A hybrid three-stage flexible flowshop problem: efficient heuristics to minimize makespan," *European Journal of Operational Research*, vol. 109, pp. 321–329, 1998.

[12] C. Paternina-Arboleda, J. Montoya-Torres, M. Acero-Dominguez, and M. Herrera-Hernandez, "Scheduling jobs on a k-stage flexible flow-shop," *Annals of Operations Research*, vol. 164, pp. 29–40, 2008.

[13] J. Jungwattanakit, M. Reodecha, P. Chaovalitwongse, and F. Werner, "A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria," *Computers and Operations Research*, vol. 36, pp. 358–378, 2009.

[14] H. Chen and P. Luh, "An alternative framework to Lagrangian relaxation approach for job shop scheduling," *European Journal of Operational Research*, vol. 149, pp. 499–512, 2003.

[15] L. Tang, H. Xuan, and J. Liu, "A new Lagrangian relaxation algorithm for hybrid flowshop scheduling to minimize total weighted completion time," *Computers and Operations Research*, vol. 33, pp. 3344–3359, 2006.

[16] L. Tang and H. Xuan, "Lagrangian relaxation algorithms for real-time hybrid flowshop scheduling with finite intermediate buffers," *Journal of the Operational Research Society*, vol. 57, pp. 316–324, 2006.

[17] H. Chen, C. Chu, and J.-M. Proth, "An improvement of the Lagrangian relaxation approach for job shop scheduling: a dynamic programming method," *IEEE Transactions on Robotics and Automation*, vol. 14 (5), pp. 786–795, 1998.

[18] Z. Zhu and R. Heady, "Minimizing the sum of earliness/tardiness in multi-machine scheduling: a mixed integer programming approach," *Computers and Industrial Engineering*, vol. 38, pp. 297–305, 2000.