KATHOLIEKE UNIVERSITEIT
LEUVEN

Faculty of Business and Economics

Optimal solutions for a dock assignment problem with trailer transportation

Lotte Berghman, Roel Leus and Frits Spieksma

DEPARTMENT OF DECISION SCIENCES AND INFORMATION MANAGEMENT (KBI)

KBI 1010

Optimal solutions for a dock assignment problem with trailer transportation

Lotte Berghman*
Research Group ORSTAT
Katholieke Universiteit Leuven, Belgium
Tel. +32 16 32 69 27; Fax +32 16 32 66 24
Lotte.Berghman@econ.kuleuven.be

Roel Leus
Research Group ORSTAT
Katholieke Universiteit Leuven, Belgium
Tel. +32 16 32 69 67; Fax +32 16 32 66 24
Roel.Leus@econ.kuleuven.be

Frits C.R. Spieksma
Research Group ORSTAT
Katholieke Universiteit Leuven, Belgium
Tel. +32 16 32 69 76; Fax +32 16 32 66 24
Frits.Spieksma@econ.kuleuven.be

— April 2010 —

*Corresponding author

Optimal solutions for a dock assignment problem with trailer transportation

Abstract: This paper presents a model for a dock assignment problem, where trailers need to be assigned to gates for a given period of 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 the second stage consists of a different set of identical parallel machines. We examine multiple integer-programming formulations for the parallel-machine model in stage two and for the three-stage flow shop, we look into the issue of symmetry and we provide extensive computational results. Our goal is to explore the limits of the instance sizes that can be solved to guaranteed optimality within acceptable running times using integer programming.

Keywords: dock assignment, parallel machines, flexible flow shop, integer programming, symmetry.

---

1 Introduction

We examine a warehouse that is used for distribution purposes. There are incoming trailers that need to be unloaded after they arrive at the warehouse, and there are outgoing trailers that need to be loaded before they leave the warehouse. The warehouse features several gates, and each gate can hold at most one trailer at any moment in time. Each gate can be used for loading as well as for unloading a trailer. The site also contains a parking lot, which can be seen as a buffer where trailers are temporarily parked. All transportation activities of trailers between this parking lot and the gates are performed by *terminal tractors*, which are tractors designed for use in ports, terminals and heavy industry. Each incoming trailer, for which the planned arrival time is known (a release date), is dropped off by a trucker at the parking lot and afterwards transported to a gate by a terminal tractor for unloading. Each outgoing trailer, for which a planned departure time is known (a deadline) is available at the parking lot, and also needs to be transported to a gate by a terminal tractor for loading. After unloading or loading at the gate, the trailer is transported back to the parking lot by a terminal tractor, where it will be picked up by a trucker later on.

For each trailer, the activities carried out consist of three stages. The first stage is the transportation of the trailer by a terminal tractor from the parking lot to a gate. Here, we need to decide when this operation starts, and which terminal tractor is used. The second stage is the loading or unloading task; we need to decide at which gate this operation takes place. The third stage is the transportation by a terminal tractor back to the parking lot. Again, the decision needs to be made when this operation starts, and by which terminal tractor it is performed. Notice that the same set of identical machines (the terminal tractors) executes both the first and the third stage. The processing times of the corresponding

operations (i.e., the transportation times) are assumed to be independent of the trailer and the gate. Another set of identical machines executes the second stage (the corresponding processing times depend on the trailer, and do not depend on the gate). The gate assigned to a trailer is considered to be occupied also during the transportation stages one and three, mainly for safety reasons. Consequently, also the 'gate'-resources are not exclusively tied to only one stage.

The dock assignment problem described above is modeled after a situation encountered at a Toyota warehouse in Diest, Belgium. The assumptions we gave follow this practical situation closely. After discussions with the management, it also became clear that the quality of a solution crucially depends on the achievement of two goals: (i) satisfying the deadlines of the outgoing trailers, and (ii) minimizing the waiting times of the incoming trailers. These two objectives will be incorporated in our models.

The contributions of this text are threefold: (1) we propose and compare various integer-programming (IP) formulations for the parallel-machine scheduling problem corresponding to stage two; (2) we study different resolution techniques for the symmetry issue; and (3) based on the comparison of the different (IP) formulations for the parallel-machine scheduling problem, we give a time-indexed formulation for the dock assignment problem that leads to good computational results for medium-size instances.

The remainder of this article is structured as follows. Section 2 presents a brief literature survey on the related topics of parallel-machine scheduling with ready times, truck and container scheduling and flexible flow shops. Some definitions and a detailed problem statement are given in Section 3. Various IP formulations for stage two (parallel-machine scheduling) are studied in Section 4. In the subsequent section (Section 5), we investigate several ways in which to remedy the disadvantages caused by symmetry in these formulations, and in Section 6 we study the benefit of adding valid inequalities. A formal statement of the flexible flow-shop problem is given in Section 7, and the best performing formulation for the parallel-machine case is extended towards this setting. We round off the article with some conclusions in Section 8.

2 Literature review

In this section, we briefly review the recent work in a number of relevant fields. First, we survey the literature on mathematical formulations for parallel-machine scheduling with ready times. Secondly, the literature on truck and container scheduling is described and finally, a brief overview of the literature on flexible flow-shop scheduling is given.

2.1 Mathematical programming for parallel-machine scheduling with ready times

A review of the state of the art of parallel-machine scheduling up to 1990 is given by Cheng and Sin (1990) and a survey of mathematical-programming formulations for machine scheduling, including parallel-machine environments, can be found in Blazewicz et al. (1991).

Dessouky (1998); Jain and Grossmann (2001); Sadykov and Wolsey (2006) and Bard and Rojanasoonthon (2006) present formulations for parallel-machine scheduling with ready

times where there is a variable denoting the start time of a job. The non-linear model of Dessouky (1998) assigns jobs to positions on machines and determines a completion time for each job. Jain and Grossmann (2001) search for a minimum-cost assignment of jobs based on a processing cost for each job-machine assignment. Their mixed-integer linear model assigns jobs to machines and uses separate decision variables for sequencing the set of jobs assigned to each machine. Some logical cuts are added to the formulation in order to reduce the computation time. The objective of Bard and Rojanasoonthon (2006) is to maximize the weighted number of jobs scheduled, where a job in a higher priority class has infinitely more weight than a job in a lower priority class. Their IP formulation uses binary variables to assign jobs to machines and to sequence the jobs.

Time-indexed formulations have recently also received a great deal of attention; one of the reasons for their good performance is the fact that the linear-programming relaxations provide strong lower bounds. The binary decision variables associate one starting period with each job. Sousa and Wolsey (1992); Crama and Spieksma (1996); van den Akker et al. (2000); Bigras et al. (2008) and Kedad-Sidhoum et al. (2008) all present time-indexed formulations for a single-machine problem based on the one presented by Dyer and Wolsey (1990), which can easily be extended to parallel machines.

2.2 Scheduling problems with transporters

Two other areas in which trailers are scheduled are cross docking and container terminals. The truck-dock assignment problem examines the scheduling of a set of trailers at docks over time (Miao et al. 2009). A number of area-specific constraints are added in order to link the inbound and outbound shipments (see Boysen et al. 2010) or to model the operations within the cross dock (see Miao et al. 2009). Heuristics are often used to solve realistic instances.

Böse et al. (2000) describe the main logistic processes in seaport container terminals and propose evolutionary algorithms for optimization. Bish et al. (2001) and Bish et al. (2005) concentrate on the transportation of containers from a ship to a yard using a fleet of vehicles. Since the authors focus on the performance for large instances, heuristics are put forward.

2.3 Flexible flow-shop scheduling

The dock assignment problem can be seen as a flexible flow shop. In a flexible flow shop, at least one stage consists of parallel machines. The terminal tractors in this paper can be modeled as machines rather than transporters, especially since the time it takes the tractors to convey a trailer between the gates and the parking lot is essentially independent of the distance. In this way, the transportation activities become stages in a flexible flow shop.

Linn and Zhang (1999); Vignier et al. (1999) and Ribas et al. (2010) all provide a survey of the flexible flow-shop literature (also called hybrid flow shop or multi-processor flow shop). Most studies deal with two-stage flow shops with parallel machines either in the first or in the second stage, but not in both. There are many research articles related to flexible flow-shop scheduling, but most of these do not deal with ready times. Both approximation (see, e.g., Tang and Xuan 2006; Nichi et al. 2010) and optimal approaches (see, e.g., Kis and Pesch 2005; Haouari et al. 2006) appear in literature.

A limited number of articles propose solution procedures for flow-shop scheduling problems with release times. Moursli and Pochet (2000) introduce a branch-and-bound algorithm for makespan minimization that produces high-quality results even when it is truncated after a few minutes of computation time. Gupta et al. (2002) generalize well-known heuristic approaches and present constructive algorithms based on job insertion techniques and iterative algorithms based on local search. Paternina-Arboleda et al. (2008) propose a heuristic for makespan minimization based on the identification and exploitation of the bottleneck stage.

3 Definitions and detailed problem statement

In this text, the dock assignment problem is modeled as a three-stage flexible flow-shop problem. Each job is composed of three tasks, one for each stage. The first stage is the transportation of the trailer to the gate by a terminal tractor, the second stage is the loading or unloading task, and the third stage is the transportation of the trailer back to the parking lot by a terminal tractor. Each task of stage two has to be scheduled on exactly one gate, and each task of stage one and three has to be scheduled on exactly one terminal tractor.

The set J contains all jobs (or trailers), with $|J| = n$, while T is the set of all the tasks to be performed (also referred to as activities). Each job $j \in J$ is a vector (t_1, t_2, t_3) of three tasks, one at each stage (the first component is the task in the first stage, etc.). T can be partitioned as follows: $T = T^1 \cup T^2 \cup T^3$ with T^i the set of all tasks of stage i $(i = 1, 2, 3)$. A second partition is $T = T_U \cup T_L$, where the set T_U contains all tasks related to a trailer that has to be unloaded, while T_L gathers all the tasks pertaining to a trailer to be loaded. Each task $t \in T^1$ has a ready time r_t. For the unloading tasks, this ready time equals the planned arrival time of the trailer; for the loading tasks we have $r_t = 0$ because we assume that the empty trailer is already available at the parking lot. For each task $t_2 \in T^2$ there is a processing time p_t, denoting the time to load or unload the trailer. Further, in this second stage $m < n$ identical gates constitute the resources; the set G contains all these machines $(|G| = m)$. Each machine (either a gate or a tractor) can process at most one task at a time. Each third-stage loading task $t \in T_L \cap T^3$ has a deadline $\overline{d}_t$, which is based on the agreed arrival time at the customer. All transportation activities between the parking lot and the gates have a constant duration of one time unit. These transportation times are modeled as being independent of the driving distance because the actual driving time of the terminal tractor is low compared to the time it takes the driver to follow the safety instructions and attach the trailer to the tractor. There are τ identical terminal tractors available for executing the transportation activities of both the first and the third stage. Preemption of a task is not allowed.

Informally, the goal is to unload all incoming shipments (stage two) as quickly as possible, and to have all outgoing trailers ready for transport (stage three) by their deadline. In our formulations we will ensure the latter requirement as a hard restriction. As our objective we choose the weighted sum of completion times, where for unloading jobs, the completion time of stage two is important, while for loading jobs we focus on the completion time of stage three. Each of the tasks in these two sets also has a weight w_t, representing the importance of the job.

The gate assigned to a trailer is considered to be occupied also during the transportation

job	weight	ready time	processing time	deadline	type
1	3	0	11		U
2	1	0	14		U
3	1	1	15		U
4	3	2	13		U
5	2	5	12		U
6	1	0	12	30	L
7	3	0	10	20	L
8	2	0	13	36	L
9	1	0	12	19	L
10	3	0	13	16	L

Table 1: Data for the example instance. Type 'U' are unload jobs, type 'L' are load jobs. All parameters (weight, ready time, ...) pertain to the appropriate tasks of each job.

stages one and three, mainly for safety reasons. Additionally, after loading or unloading, a trailer cannot immediately be transported to the parking lot if all tractors are busy. The trailer remains at the gate until a terminal tractor becomes available, which may prevent other trailers from being loaded or unloaded there. In line with Kise et al. (1991) and following the literature on manufacturing flow lines (see, e.g., Dallery and Gershwin 1992), we refer to this phenomenon as *blocking*. The blocking time is the difference between the ending time of the loading or unloading task and the starting time of the transportation to the parking lot.

Finding the optimal schedule for a set of tasks with release times is NP-hard, even on a single processor (see Lenstra and Rinnooy Kan 1978). Consequently, finding the optimal schedule for the considered flexible flow-shop problem is also NP-hard. An example of a problem instance is provided in Table 1, where for ease of notation each parameter (ready time, weight, ...) pertaining to one particular task of a job is specified as a parameter of the job. A feasible schedule for this instance with $\tau = 1$ tractor is described in Figure 1. The green blocks represent the transportation tasks done by the terminal tractors and the blue blocks represent the blocking time between stages two and three.

A large part of this article (in particular, Sections 4, 5 and 6) will investigate the specific setting in which only the loading and unloading activities are taken into account and the terminal tractors are left aside – in other words, we only schedule the tasks in T^2 on the gates. In doing so, our goal is to identify a formulation that can satisfactorily deal with instances of realistic size. Although the resulting problem is a simplification, it is not an unrealistic approximation of reality when processing times at the gates are significantly larger than the tractor movement times, and there is a sufficiently high number of tractors. Notice that the problem in stage two can be denoted by $Pm|r_j, \overline{d}_j| \sum w_j C_j$ in the standard three-field notation. Our findings for this special case will be useful for producing an appropriate IP formulation for the flexible flow shop in Section 7. In the parallel-machine setting, each job's ready time, weight and due date or deadline are associated with its stage-two task. An example of a feasible parallel-machine schedule for the problem instance introduced in Table 1 is given in Figure 2.

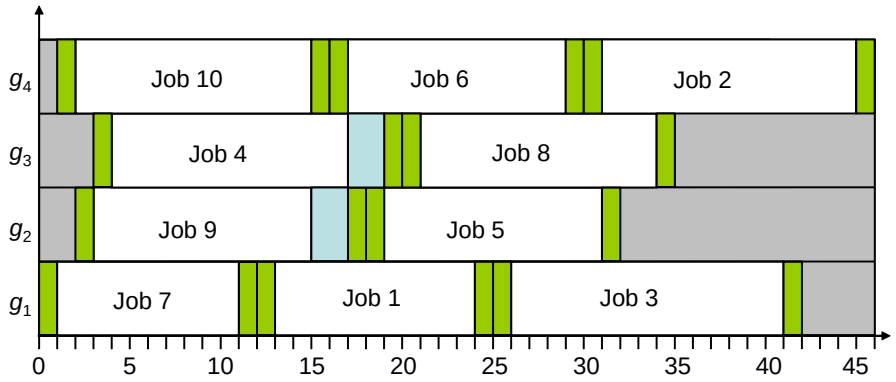

Figure 1: A feasible schedule for the example instance. Each rectangle labeled 'Job i' represents the stage-two task of the particular job; g_k is the k^{th} gate.

4 Parallel-machine scheduling

In this section, we schedule only the stage-two load and unload activities and we neglect the work of the terminal tractors. We examine two assignment-based formulations (in Sections 4.1 and 4.2), a flow formulation (Section 4.3) and a time-indexed formulation (Section 4.4). We conclude the section by a comparison of the formulations from a theoretical point of view and by means of numerical experiments (Section 4.5).

4.1 Assignment-based formulation 1

The first formulation AB1 (assignment-based formulation 1) is based on Dessouky (1998). Below, we introduce additional variables z_{tu} to linearize the formulation. The decision variables of AB1 are the following. For every task $t \in T^2$ and for every gate $g \in G$,

$$x_{ti}^{g} = \begin{cases} 1 & \text{if task } t \text{ is the } i^{th} \text{ task at gate } g, \\ 0 & \text{otherwise.} \end{cases}$$

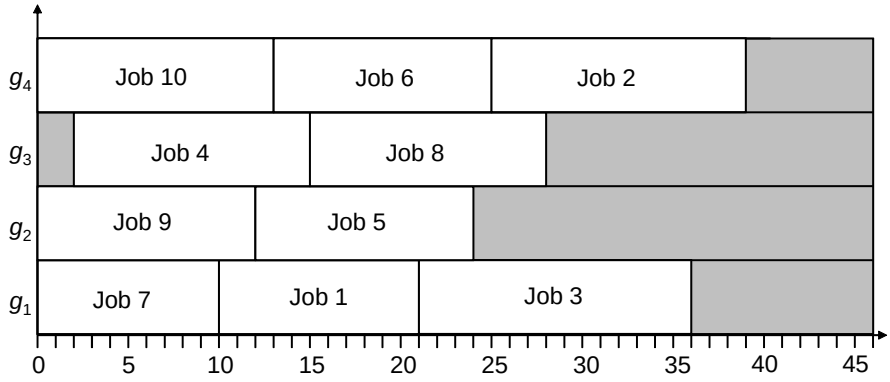

Figure 2: A feasible parallel-machine schedule for the example instance.

Additionally, for all tasks $t, u \in T^2$, we define

$$z_{tu} = \begin{cases} 1 & \text{if task } t \text{ precedes task } u \text{ and both tasks are executed at the same gate,} \\ 0 & \text{otherwise.} \end{cases}$$

Finally, for every task $t \in T^2$ we also have a completion time C_t. Formulation AB1 for the considered parallel-machine problem is the following:

$$\min \sum_{t \in T^2} w_t C_t \tag{4.1}$$

subject to

$$\sum_{g \in G} \sum_{i=1}^{n} x_{ti}^g = 1 \qquad \forall t \in T^2 \tag{4.2}$$

$$\sum_{t \in T^2} x_{ti}^g \leq 1 \qquad \forall g \in G; i = 1, \ldots, n \tag{4.3}$$

$$\sum_{t \in T^2} x_{ti}^g \geq \sum_{t \in T^2} x_{t,i+1}^g \qquad \forall g \in G; i = 1, \ldots, n-1 \tag{4.4}$$

$$x_{ti}^g + \sum_{j=i+1}^{n} x_{uj}^g \leq 1 + z_{tu} \qquad \forall \{t, u\} \subset T^2; \forall g \in G; i = 1, \ldots, n-1 \tag{4.5}$$

$$C_t - (1 - z_{tu})M \leq C_u - p_u \qquad \forall \{t, u\} \subset T^2 \tag{4.6}$$

$$r_t \leq C_t - p_t \qquad \forall t \in T^2 \tag{4.7}$$

$$C_t \leq \overline{d}_t \qquad \forall t \in T_L \cap T^2 \tag{4.8}$$

$$z_{tu} \in \{0, 1\} \qquad \forall \{t, u\} \subset T^2 \tag{4.9}$$

$$x_{ti}^g \in \{0, 1\} \qquad \forall t \in T^2; \forall g \in G; i = 1, \ldots, n \tag{4.10}$$

$$C_t \geq 0 \qquad \forall t \in T^2 \tag{4.11}$$

The objective function (4.1) minimizes for all tasks $t \in T^2$ the weighted completion time C_t of the task. Constraint (4.2) limits each task to be processed exactly once. Constraint (4.3) specifies that each gate can process at most one task at a time. Constraint (4.4) enforces the dominant strategy not to have an i^{th} task at a gate when there is no $(i-1)^{th}$ task. Constraints (4.5) and (4.6) ensure that at each gate, the ending time of each task is not larger than the starting time of the following task. The parameter M is a large number; a possible value for M is $\max_{t \in T^2}\{r_t\} + \sum_{t \in T^2} p_t$ (which is the value used in our implementation). Constraint (4.7) imposes that a task cannot start before its ready time and constraint (4.8) demands that a task be finished by its deadline. Finally, constraints (4.9) and (4.10) state that the decision variables z_{tu} and x_{ti}^g are binary and constraint (4.11) requires all completion times to be non-negative.

4.2 Assignment-based formulation 2

In retrospect, in the previous formulation AB1 the index i for the position at the gate in the decision variable x_{ti}^g seems to be redundant, since sequencing decisions are also implicit in

the additional variables z_{tu}. In our second assignment-based formulation AB2, this position index is left out. More specifically, for all tasks $t \in T^2$ and for every gate $g \in G$, we have

$$x_t^g = \begin{cases} 1 & \text{if task } t \text{ is scheduled at gate } g, \\ 0 & \text{otherwise.} \end{cases}$$

The decision variables z_{tu} and C_t remain unchanged. The new choice of decision variables leads to the following linear formulation AB2 of the parallel-machine problem:

$$\min \sum_{t \in T^2} w_t C_t$$

subject to

$$\sum_{g \in G} x_t^g = 1 \qquad \forall t \in T^2 \tag{4.12}$$

$$x_t^g + x_u^g - z_{tu} - z_{ut} \leq 1 \qquad \forall \{t,u\} \subset T^2; \forall g \in G \tag{4.13}$$

$$C_t - (1 - z_{tu})M \leq C_u - p_u \qquad \forall \{t,u\} \subset T^2$$

$$r_t \leq C_t - p_t \qquad \forall t \in T^2$$

$$C_t \leq \overline{d}_t \qquad \forall t \in T_L \cap T^2$$

$$x_t^g \in \{0,1\} \qquad \forall t \in T^2; \forall g \in G$$

$$z_{tu} \in \{0,1\} \qquad \forall \{t,u\} \subset T^2$$

$$C_t \geq 0 \qquad \forall t \in T^2$$

Constraint (4.12) demands that each task be assigned to exactly one gate. Constraint (4.13) ensures that if tasks t and u are assigned to the same gate g, then one must be processed before the other. The remainder of the model is similar to AB1. This formulation is close to the ones presented in Jain and Grossmann (2001), although in that reference the parallel machines are not identical in that the processing times depend on the machine and the objective is to minimize the sum of the processing costs of the job-machine combinations.

4.3 Flow formulation

In the following formulation, subsequently referred to as formulation F (for 'flow-based'), a dummy task t_0 that acts both as the first and as the last task in the activity sequence at each gate is added to the model: $T_0^2 = T^2 \cup \{t_0\}$. The decision variables are the following: for all tasks $\{t,u\} \subset T_0^2$ and for every gate $g \in G$,

$$x_{tu}^g = \begin{cases} 1 & \text{if task } t \text{ is the immediate predecessor of task } u \text{ at gate } g, \\ 0 & \text{otherwise.} \end{cases}$$

Similar to the previous formulations, every task $t \in T^2$ has a completion time C_t. We propose the following formulation F:

$$\min \sum_{t \in T^2} w_t C_t$$

subject to

$$\sum_{u \in T_0^2 \setminus \{t\}} \sum_{g \in G} x_{tu}^g = 1 \qquad \forall t \in T^2 \tag{4.14}$$

$$\sum_{t \in T_0^2} x_{t_0 t}^g = 1 \qquad \forall g \in G \tag{4.15}$$

$$\sum_{u \in T_0^2 \setminus \{t\}} x_{ut}^g - \sum_{u \in T_0^2 \setminus \{t\}} x_{tu}^g = 0 \qquad \forall t \in T^2; \forall g \in G \tag{4.16}$$

$$C_t - (1 - x_{tu}^g)M \leq C_u - p_u \qquad \forall \{t, u\} \subset T^2; \forall g \in G \tag{4.17}$$

$$r_t \leq C_t - p_t \qquad \forall t \in T^2$$

$$C_t \leq \overline{d}_t \qquad \forall t \in T_L \cap T^2$$

$$x_{tu}^g \in \{0, 1\} \qquad \forall \{t, u\} \subset T_0^2; \forall g \in G \tag{4.18}$$

$$C_t \geq 0 \qquad \forall t \in T^2$$

Constraint (4.14) restricts each task to be processed exactly once and ensures that when a task is scheduled, it has exactly one successor, which can be the dummy task t_0. Constraint (4.15) limits the number of initial tasks. These constraints (4.14) and (4.15) indirectly specify that each machine can process at most one task at a time. Constraint (4.16) entails the conservation of flow: if task t is assigned to gate g, then both its predecessor and successor must also be processed by gate g. This formulation is based on Bard and Rojanasoonthon (2006). The main differences with their setting are the non-identical parallel machines, the setup times, the priority classes containing the tasks and the corresponding contributions to the objective function.

4.4 Time-indexed formulation

The time-indexed formulation TI relies on a discretization of the planning horizon, for the length of which we use the practical upper bound $H_{\max}$. The formulation is based on Dyer and Wolsey (1990). For all tasks $t \in T^2$, for all time periods $u \in H_t$ and for every gate $g \in G$,

$$x_{tu}^g = \begin{cases} 1 & \text{if processing of task } t \text{ starts in time period } u \text{ at gate } g, \\ 0 & \text{otherwise,} \end{cases}$$

where H_t is defined as follows: $H_t = \{r_t + 1, \ldots, H_{\max} - p_t + 1\}$ if $t \in T_U \cap T^2$ and $H_t = \{r_t + 1, \ldots, \overline{d}_t - p_t + 1\}$ if $t \in T_L \cap T^2$. We call this set of time periods the *time window* of a task. A time-indexed linear formulation TI of the parallel-machine problem is then the following:

$$\min \sum_{t \in T^2} w_t \left(\left(\sum_{u \in H_t} u \sum_{g \in G} x_{tu}^g \right) - 1 + p_t \right) \tag{4.19}$$

subject to

$$\sum_{g\in G}\sum_{u\in H_t} x_{tu}^g = 1 \qquad \forall t \in T^2 \tag{4.20}$$

$$\sum_{t\in T^2}\sum_{v=u-p_t+1}^{u} x_{tv}^g \leq 1 \qquad \forall g \in G; \forall u \in \{1,\ldots,H_{\max}\} \tag{4.21}$$

$$x_{tu}^g \in \{0,1\} \qquad \forall t \in T^2; \forall u \in H_t; \forall g \in G \tag{4.22}$$

The objective function (4.19) has a similar interpretation as before. Constraint (4.20) requires each task to be started exactly once. Constraint (4.21) ensures that at a given time period u, only one tasks can be executed on each gate. Here and below, decision variables that are undefined because of the time windows do not appear in the model. Finally, constraint (4.22) states that the decision variables x_{tu} are binary variables. In this formulation no set of big-M constraints is needed, but a major disadvantage is obviously the pseudo-polynomial number of variables.

For the determination of a tight value for $H_{\max}$, we proceed as follows. Let $r_{\max} = \max_{t\in T^2}\{r_t\}$ and $F = (T_U \cap T^2) \cup \{v \in (T_L \cap T^2) : \overline{d}_v > r^*\}$. Let $l_{\max} = \arg\max_{t\in F}\{p_t\}$. An upper bound on the schedule length of at least one optimal schedule is $r_{\max} + \left\lfloor \frac{\sum_{t\in F\setminus\{l_{\max}\}} p_t}{m} \right\rfloor + p_{l_{\max}}$, so throughout Sections 4, 5 and 6, we set $H_{\max} = r_{\max} + \left\lfloor \frac{\sum_{t\in F\setminus\{l_{\max}\}} p_t}{m} \right\rfloor + p_{l_{\max}}$.

4.5 Comparison of the formulations

In order to compare two formulations that are stated in terms of different variables, one should compare the projection of the polyhedra of the linear relaxations of both formulations in the same space (see, e.g., Oncan et al. 2009). In particular, the binary variables of AB2 can be written as a function of the variables of AB1 in the following way:

$$x_t^g = \sum_{i=1}^{n} x_{ti}^g \qquad \forall t \in T^2; \forall g \in G \tag{4.23}$$

When the dominant decision (4.4) is neglected, the only difference between the two formulations is that the first formulation contains constraint (4.5) while constraint (4.13) is part of the second formulation; the latter constraint can be rewritten as

$$\sum_{i=1}^{n} x_{ti}^g + \sum_{i=1}^{n} x_{ui}^g - z_{tu} - z_{ut} \leq 1 \qquad \forall \{t,u\} \subset T^2; \forall g \in G \tag{4.24}$$

For two tasks t and u and a gate g, the following combination of fractional values for the decision variables is admissible for the LP relaxation of AB1 but not for (4.24):

$$\begin{array}{lll} x_{t2}^g = 0.5 & x_{u2}^g = 0.5 & z_{tu} = 0 \\ x_{t3}^g = 0.5 & x_{u3}^g = 0.5 & z_{ut} = 0 \end{array}$$

On the other hand, the following values satisfy all constraints of the linear relaxation of AB2, while constraint (4.5) is not respected.

$$\begin{array}{ll} x_{t1}^g = 1 & z_{tu} = 0.5 \\ x_{u2}^g = 1 & z_{ut} = 0.5 \end{array}$$

Consequently, neither of these two formulations is stronger than the other.

We use the name AB1_ext to refer to formulation AB1 extended with constraint (4.24) as valid inequalities, and AB2_1 is the formulation AB2 expressed in the variables of formulation AB1 according to (4.23). On comparing these two formulations, we observe that AB1_ext is tighter. Constraints (4.3), (4.4) and (4.5) can be considered to constitute valid inequalities for formulation AB1_ext. From computational experiments, we learn that there is no considerable difference between the computation times for formulation AB1_ext without these three constraint sets and formulation AB2_1 (the results for this comparison are not reported in this text). We find that including (4.3), (4.4) and (4.5) into formulation AB1_ext only increases the computation times. Consequently, we see no considerable empirical advantage of using the variables x_{ti}^g rather than x_t^g (despite the tightened formulation).

Comparing assignment-based, flow and time-indexed formulations with one another is difficult; the problem lies in the establishment of a direct relation between the decision variables. From a theoretical point of view, it is not predictable in a straightforward manner which formulation will perform best. Dyer and Wolsey (1990) conclude that the relaxations of formulations based on time discretization give stronger bounds than formulations using decision variables representing starting times and sequencing choices for their 1-machine scheduling problem with ready times. Based on experimental running times, Mellouli et al. (2009) find that an assignment formulation performs better than a flow formulation for parallel-machine scheduling without ready times.

We compare the performance (especially the computation times) of our proposed formulations empirically for a set of test instances[1]. All experiments were executed with ILOG OPL Development Studio on a Dell Latitude D630 with an Intel Pentium-4 2.2-GHz processor and 2 GB RAM, equipped with Windows 7. Most of the test instances are based on those generated by Jain and Grossmann (2001) and Sadykov and Wolsey (2006) for non-identical parallel-machine scheduling with "freedom" parameter $\theta = 0.6$ (some additional small instances are created in a similar way). The first half of the tasks are unloading tasks while the second half are loading tasks. The ready times of the loading tasks are set to 0. The weights are randomly selected out of $\{1, 2, 3\}$ (each value has equal probability). To obtain a single processing time, for each task a random number between 1 and the number of machines is generated and the processing time of the task on that machine is used.

Table 2 contains the results of our formulations on the small-size test instances; here and below, a time limit of one hour is imposed on the CPU time. The computational results of the first assignment-based formulation AB1 are significantly worse than the results of the other three formulations; as mentioned before, there is no empirical advantage of using the variables x_{ti}^g and for this reason AB1 will not be studied further in the remainder of this article. We find that the formulation TI performs best, probably because of the tight LP bound.

[1]All instances can be found at the website
http://www.econ.kuleuven.be/public/NDBAC96/gate_assignment.htm.

# jobs	# gates	AB1	AB2	F	TI
3	2	1.60 s	1.29 s	1.55 s	1.04 s
5	2	1.02 s	1.80 s	1.56 s	0.79 s
7	2	5.74 s	2.33 s	2.08 s	1.02 s
7	3	2.59 s	1.54 s	1.53 s	0.79 s
9	3	2768.97 s	3.61 s	7.83 s	1.29 s
10	3	> 1 h	55.37 s	299.22 s	0.78 s
12	3	> 1 h	> 1 h	> 1 h	0.76 s

Table 2: Computation times for the different formulations.

5 Symmetry

When symmetry is inherent in the problem, multiple combinations of values for the variables may represent the same solution. This poses a problem for IP solvers, because many subproblems in the enumeration tree can be isomorphic, resulting in a wasteful duplication of computational effort. Even for relatively modestly sized problems, integer linear programs with large symmetry groups are difficult to solve using traditional branch-and-bound or branch-and-cut algorithms (see, e.g., Sherali and Smith 2001; Margot 2008; Jans 2008). As we deal with identical machines, many alternative optimal solutions can be created by simply renumbering the machines (see Figure 3). This section presents symmetry-breaking constraints (SBCs) and adapted formulations to work around this symmetry.

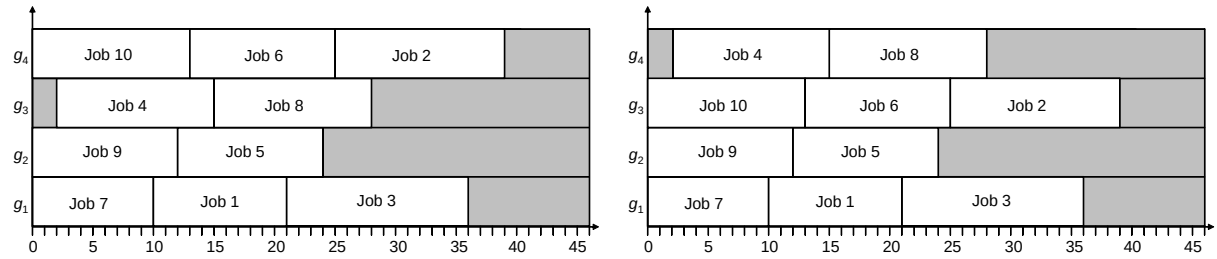

Figure 3: These two figures represent essentially the same solution, but machines 3 and 4 are interchanged.

5.1 Symmetry-breaking constraints

A first and very straightforward set of constraints that eliminates (part of) the symmetry requires that the number of tasks that are scheduled on machine g be at least as high as the number on machine $g + 1$. The following constraints apply to models AB2, F and TI,

respectively.

$$\sum_{t\in T^2} x_t^g \geq \sum_{t\in T^2} x_t^{g+1} \qquad \forall g \in G; g \neq m$$

$$\sum_{t\in T^2} \sum_{u\in T_0^2\setminus\{t\}} x_{tu}^g \geq \sum_{t\in T^2} \sum_{u\in T_0^2\setminus\{t\}} x_{tu}^{g+1} \qquad \forall g \in G; g \neq m$$

$$\sum_{t\in T^2} \sum_{u\in H_t} x_{tu}^g \geq \sum_{t\in T^2} \sum_{u\in H_t} x_{tu}^{g+1} \qquad \forall g \in G; g \neq m$$

A second set of constraints is based on Jans (2008). A unique number $\sum_{t\in C} 2^t$ is assigned to each possible configuration C of tasks on a machine, and the machines are ordered by decreasing value of this number (with $C \subset T^2$). As before, the following three constraint sets are defined for formulations AB2, F and TI, in that order.

$$\sum_{t\in T^2} x_t^g 2^t \geq \sum_{t\in T^2} x_t^{g+1} 2^t \qquad \forall g \in G; g \neq m$$

$$\sum_{t\in T^2} \sum_{u\in T_0^2\setminus\{t\}} x_{tu}^g 2^t \geq \sum_{t\in T^2} \sum_{u\in T_0^2\setminus\{t\}} x_{tu}^{g+1} 2^t \qquad \forall g \in G; g \neq m$$

$$\sum_{t\in T^2} \sum_{u\in H_t} x_{tu}^g 2^t \geq \sum_{t\in T^2} \sum_{u\in H_t} x_{tu}^{g+1} 2^t \qquad \forall g \in G; g \neq m$$

A third possible set of SBCs states that the first m tasks have to be scheduled on a specific set of machines. More specifically, the task t with $t \leq m$ is scheduled on one of the machines in the set $\{1, \ldots, t\}$. The following constraint sets can be added to the three formulations.

$$\sum_{g=1}^{t} x_t^g = 1 \qquad \forall t \in \{1, \ldots, m\}$$

$$\sum_{u\in T_0^2\setminus\{t\}} \sum_{g=1}^{t} x_{tu}^g = 1 \qquad \forall t \in \{1, \ldots, m\}$$

$$\sum_{g=1}^{t} \sum_{u\in H_t} x_{tu}^g = 1 \qquad \forall t \in \{1, \ldots, m\}$$

For the relevant tasks, these constraints will replace (4.12), (4.14) and (4.20), respectively.

Our fourth set of SBCs forces each task to be scheduled on the machine with the lowest index (giving priority to the lowest-indexed tasks): a task t can only be scheduled on a specific machine g if at least one task in the set $\{1, \ldots, t-1\}$ is planned on machine $g-1$.

The following constraint sets are used for the three retained formulations of Section 4:

$$x_t^g \leq \sum_{v=1}^{t-1} x_v^{g-1} \qquad \forall t \in T^2 \setminus \{1\}; \forall g \in G \setminus \{1\}$$

$$\sum_{u \in T_0^2 \setminus \{t\}} x_{tu}^g \leq \sum_{v=1}^{t-1} \sum_{u \in T_0^2 \setminus \{v\}} x_{vu}^{g-1} \qquad \forall t \in T^2 \setminus \{1\}; \forall g \in G \setminus \{1\}$$

$$\sum_{u \in H_t} x_{tu}^g \leq \sum_{v=1}^{t-1} \sum_{u \in H_v} x_{vu}^{g-1} \qquad \forall t \in T^2 \setminus \{1\}; \forall g \in G \setminus \{1\}$$

We will refer to the foregoing four types of SBCs as SBC1, SBC2, SBC3 and SBC4, respectively.

5.2 Adapted formulations

Chen and Powell (1999) and Mellouli et al. (2009) note that it is not needed to specify which machine is going to execute which sequence as all machines are identical: a solution to our parallel-machine scheduling problem simply consists of m single-machine schedules.

The flow formulation can be adapted by replacing the decision variables x_{tu}^g by the following ones: for all tasks $\{t, u\} \subset T_0^2$,

$$x_{tu} = \begin{cases} 1 & \text{if task } t \text{ is scheduled immediately before task } u \text{ at the same gate,} \\ 0 & \text{otherwise.} \end{cases}$$

Constraints (4.14), (4.15), (4.16), (4.17) and (4.18) are altered in the following way:

$$\sum_{u \in T_0^2 \setminus \{t\}} x_{tu} = 1 \qquad \forall t \in T^2$$

$$\sum_{t \in T_0^2} x_{t_0 t} = m$$

$$\sum_{u \in T_0^2 \setminus \{t\}} x_{ut} - \sum_{u \in T_0^2 \setminus \{t\}} x_{tu} = 0 \qquad \forall t \in T^2$$

$$C_t - (1 - x_{tu})M \leq C_u - p_u \qquad \forall \{t, u\} \subset T^2$$

$$x_{tu} \in \{0, 1\} \qquad \forall \{t, u\} \subset T_0^2$$

For the time-indexed formulation, the decision variables x_{tu}^g can be replaced by the following: for all tasks $t \in T^2$ and for all time periods $u \in H_t$,

$$x_{tu} = \begin{cases} 1 & \text{if processing of task } t \text{ starts in time period } u, \\ 0 & \text{otherwise.} \end{cases}$$

The objective of the TI formulation is then modified as follows:

$$\min \sum_{t \in T^2} w_t \left(\left(\sum_{u \in H_t} u x_{tu} \right) - 1 + p_t \right)$$

Constraints (4.20), (4.21) and (4.22) are adapted in the following way:

$$\sum_{u \in H_t} x_{tu} = 1 \qquad \forall t \in T^2$$

$$\sum_{t \in T^2} \sum_{v=u-p_t+1}^{u} x_{tv} \leq m \qquad \forall u \in \{1, \ldots, H_{\max}\}$$

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

5.3 Computational results

Tables 3, 4 and 5 display the computation times for the different SBCs for the models AB2, F and TI, and also for the adapted formulations F and TI. All combinations of the different SBCs have also been tested, but the results were not better for any combination. The adapted formulations perform best for F and TI, while for the AB2 formulation, SBC2 performs better for larger instances. The columns labeled '–' represent the setting without the tentative refinements.

# jobs	# gates	–	SBC 1	SBC 2	SBC 3	SBC 4
3	2	1.29 s	1.85 s	1.80 s	1.28 s	1.54 s
5	2	1.80 s	2.07 s	2.05 s	2.07 s	2.09 s
7	2	2.33 s	2.34 s	3.36 s	3.39 s	2.60 s
7	3	1.54 s	1.81 s	2.85 s	1.85 s	1.55 s
9	3	3.61 s	29.89 s	6.75 s	4.15 s	23.64 s
10	3	55.37 s	138.22 s	44.17 s	76.99 s	184.06 s
12	3	> 1 h	> 1 h	> 1 h	> 1 h	> 1 h

Table 3: Computation times for symmetry elimination in formulation AB2.

# jobs	# gates	–	SBC 1	SBC 2	SBC 3	SBC 4	adapted form.
3	2	1.55 s	2.31 s	1.55 s	1.81 s	2.07 s	1.82 s
5	2	1.56 s	2.07 s	2.33 s	2.09 s	1.82 s	1.55 s
7	2	2.08 s	2.62 s	2.59 s	3.36 s	3.62 s	2.34 s
7	3	1.53 s	2.58 s	2.10 s	2.06 s	2.07 s	1.53 s
9	3	7.83 s	19.46 s	16.38 s	21.26 s	31.80 s	3.68 s
10	3	299.22 s	502.28 s	663.11 s	1786.32 s	> 1 h	36.95 s
12	3	> 1 h	> 1 h	> 1 h	> 1 h	> 1 h	> 1 h

Table 4: Computation times for symmetry elimination in formulation F.

# jobs	# gates	–	SBC 1	SBC 2	SBC 3	SBC 4	adapted form.
3	2	1.04 s	1.31 s	1.29 s	1.02 s	1.04 s	1.54 s
5	2	0.79 s	1.04 s	1.02 s	0.76 s	1.06 s	1.54 s
7	2	1.02 s	1.54 s	1.31 s	1.04 s	1.29 s	2.09 s
7	3	0.79 s	0.78 s	1.02 s	0.79 s	0.79 s	1.55 s
9	3	1.29 s	0.78 s	0.78 s	0.78 s	1.02 s	1.31 s
10	3	0.78 s	1.02 s	1.04 s	0.79 s	1.02 s	1.30 s
12	3	0.76 s	1.06 s	1.02 s	1.27 s	0.79 s	1.56 s
15	5	1.04 s	0.78 s	69.90 s	1.06 s	1.31 s	1.32 s
20	5	1.02 s	1.31 s	1.02 s	1.32 s	1.82 s	1.28 s
24	6	1.31 s	1.31 s	16.92 s	1.29 s	1.29 s	1.32 s
28	7	3.41 s	9.98 s	21.13 s	2.32 s	66.91 s	1.81 s
30	7	1.29 s	3.12 s	24.18 s	1.59 s	1.80 s	1.29 s
35	7	1.31 s	1.80 s	12.26 s	1.31 s	5.28 s	1.30 s
42	7	8.61 s	19.31 s	916.88 s	2.10 s	564.92 s	1.82 s
32	8	1.02 s	1.29 s	> 1 h	1.04 s	4.14 s	1.53 s
34	8	1.84 s	28.26 s	> 1 h	1.80 s	3.36 s	1.31 s
40	8	8.12 s	13.04 s	> 1 h	8.89 s	32.62 s	1.54 s
48	8	4.92 s	5.19 s	1979.96 s	4.18 s	21.13 s	1.53 s
36	9	1.56 s	3.38 s	44.38 s	1.31 s	9.11 s	1.81 s
45	9	3.63 s	9.62 s	> 1 h	3.13 s	8.90 s	1.57 s
54	9	43.97 s	> 1 h	> 1 h	15.95 s	> 1 h	1.55 s

Table 5: Computation times for symmetry elimination in formulation TI.

6 Valid inequalities

For assignment-based formulations, the following inequalities are suggested by Jain and Grossmann (2001) and Zhu and Heady (2000):

$$\sum_{t \in T_L \cap T^2} x_t^g p_t \leq \max_{t \in T_L \cap T^2} \{\overline{d}_t\} - \min_{t \in T_L \cap T^2} \{r_t\} \qquad \forall g \in G \qquad (6.1)$$

$$z_{tu} + z_{ut} \leq 1 \qquad \forall \{t, u\} \subset T^2 \qquad (6.2)$$

$$x_t^g + x_u^h + z_{tu} + z_{ut} \leq 2 \qquad \forall \{t, u\} \subset T^2; \forall \{g, h\} \subset G \qquad (6.3)$$

$$z_{tu} + z_{uv} - z_{tv} \leq 1 \qquad \forall \{t, u, v\} \subset T^2 \qquad (6.4)$$

Constraint (6.1) guarantees that the total processing time of all loading tasks scheduled on one machine does not exceed the difference between the latest deadline and the earliest ready time. Equation (6.2) states that for any pair of tasks, either one task comes before the other or the other way round (on one machine), or the two tasks need not be sequenced. Constraint (6.3) demands that the sequencing variables z_{tu} and z_{ut} both be zero if tasks t and u are assigned to different gates. Finally, expression (6.4) assures that the precedence relation implied by the z-variables is transitive. All these constraints can be added to the formulation AB2 extended with the corresponding SBC2.

For the flow formulation, inequalities analogous to expressions (6.1) and (6.2) are the following:

$$\sum_{t\in T_L\cap T^2}\sum_{u\in T_0^2\setminus\{t\}} x_{tu}^g p_t \le \max_{t\in T_L\cap T^2}\{\overline{d}_t\} - \min_{t\in T_L\cap T^2}\{r_t\} \qquad \forall g\in G \qquad (6.5)$$

$$\sum_{g\in G} x_{tu}^g + \sum_{g\in G} x_{ut}^g \le 1 \qquad \forall\{t,u\}\subset T^2 \qquad (6.6)$$

$$x_{tu} + x_{ut} \le 1 \qquad \forall\{t,u\}\subset T^2 \qquad (6.7)$$

Constraints (6.5) and (6.6) can be added to the formulation F, while constraint (6.7) can be added to the adapted formulation F.

Finally, the inequality (6.1) has the following equivalent for the time-indexed formulation, which can be added to the formulation TI:

$$\sum_{t\in T_L\cap T^2}\sum_{u\in H_t} x_{tu}^g p_t \le \max_{t\in T_L\cap T^2}\{\overline{d}_t\} - \min_{t\in T_L\cap T^2}\{r_t\} \qquad \forall g\in G \qquad (6.8)$$

For the time-indexed formulation, the following inequality based on Proposition 2 of Sousa and Wolsey (1992) can be added to the formulation TI:

$$\sum_{g\in G}\left(\sum_{v\in H_t'} x_{tv}^g + \sum_{\substack{u\in T^2\setminus\{t\}\\ p_u\ge i}}\sum_{w\in H_u''} x_{uw}^g\right) \le m \quad \forall t\in T^2;\forall h\in\{1,\dots,H_{\max}\};\forall i\in\{2,\dots,p_t^{\max}\} \qquad (6.9)$$

with $H_t' = H_t\cap\{h-p_t+1, h+i-1\}$, $H_t'' = H_t\cap\{h-p_t+i, h\}$ and $p_t^{\max} = \max_{j\in T^2\setminus\{t\}}\{p_j\}$. The corresponding inequality for the adapted formulation TI is:

$$\sum_{v\in H_t'} x_{tv} + \sum_{\substack{u\in T^2\setminus\{t\}\\ p_u\ge i}}\sum_{w\in H_u''} x_{uw} \le m \quad \forall t\in T^2;\forall h\in\{1,\dots,H_{\max}\};\forall i\in\{2,\dots,p_t^{\max}\} \qquad (6.10)$$

For the single-machine case, the right-hand side of the inequalities is one. The left-hand side of the expressions selects a set of variables whose sum cannot exceed the machine capacity. For an example with three tasks, one machine, $p_1 = 5$, $p_2 = 4$ and $p_3 = 3$, a possible solution is $x_{12} = x_{13} = x_{22} = x_{16} = x_{34} = x_{38} = 0.5$ (all other $x_{tu} = 0$). However, this solution violates the inequality $x_{22} + x_{23} + x_{24} + x_{25} + x_{26} + x_{12} + x_{13} + x_{14} + x_{15} + x_{34} + x_{35} \le 2$, which corresponds with $t = 2$, $h = 5$ and $i = 2$.

In order to compare computational performance for larger instances, we have created new instances with $m\in\{10,\dots,15\}$ and $|T|\in\{4m, 5m, 6m\}$. The ready times for the unloading jobs are integers randomly drawn from $\{0,\dots,25\}$ (uniformly distributed). The processing times are chosen as $1+X$ with X binomially distributed with parameters 16 (trials) and 0.5 (probability of success). The deadlines for the loading tasks t are set to $\max\{\overline{d_t'}, r_t + p_{l_{\max}}\}$ with $\overline{d_t'}$ uniformly distributed on $\{\beta-10,\dots,\beta+10\}$, $\beta = \frac{0.5\sum_{t\in T^2} p_t}{m}$ and $l_{\max} = \arg\max_{t\in T^2}\{p_t\}$.

On comparing the computational results for the different best performing formulations extended with all valid inequalities in Tables 6, 7 and 8, it becomes clear that the adapted

# jobs	# gates	SBC 2	SBC 2 + (6.1)	SBC 2 + (6.2)	SBC 2 + (6.3)	SBC 2 + (6.4)
3	2	1.80 s	1.82 s	1.80 s	2.08 s	1.83 s
5	2	2.05 s	2.10 s	2.07 s	2.58 s	2.10 s
7	2	3.36 s	3.36 s	3.36 s	3.65 s	2.58 s
7	3	2.85 s	2.85 s	2.87 s	2.86 s	2.34 s
9	3	6.75 s	6.77 s	8.28 s	4.39 s	8.87 s
10	3	44.17 s	44.74 s	49.40 s	19.33 s	196.47 s
12	3	> 1 h	> 1 h	> 1 h	648.40 s	> 1 h

Table 6: Computation times for the valid inequalities for AB2.

# jobs	# gates	–	adapted form.	– + (6.5)	– + (6.6)	adapted form. + (6.7)
3	2	1.55 s	1.82 s	1.79 s	1.54 s	2.33 s
5	2	1.55 s	1.55 s	1.54 s	1.81 s	1.55 s
7	2	2.08 s	2.34 s	2.61 s	2.64 s	2.32 s
7	3	1.53 s	1.53 s	2.06 s	1.80 s	1.85 s
9	3	7.83 s	3.68 s	6.59 s	5.75 s	3.15 s
10	3	299.22 s	36.95 s	570.33 s	392.93 s	38.81 s
12	3	> 1 h	> 1 h	> 1 h	> 1 h	> 1 h

Table 7: Computation times for the valid inequalities for F.

time-indexed formulation is still by far the best performing. Even by combining the different valid inequalities, it was not possible to outperform the results of the adapted time-indexed formulation. Therefore, this latter formulation will be extended for the flexible flow-shop configuration.

7 Flexible flow-shop scheduling

In this section, the full dock assignment problem is modeled as a three-stage flexible flow shop. Each job is now composed of three tasks, one for each stage. We first describe a time-indexed formulation and then report the computational results.

7.1 Time-indexed formulation

As the adapted time-indexed formulation performed best for the parallel-machine scheduling problem and therefore seems the most promising, this formulation is extended to the three-stage flexible flow-shop problem. The decision variables are the following. For all tasks $t \in T$ and for all time periods $u \in H_t$,

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

with for the loading jobs $H_t = \{r_t + 1, \ldots, H_{\max} - p_t - 1\}$ if $t \in T_U \cap T^1$, $H_t = \{r_t + 2, \ldots, H_{\max} - p_t\}$ if $t \in T_U \cap T^2$ and $H_t = \{r_t + 2 + p_t, \ldots, H_{\max}\}$ if $t \in T_U \cap T^3$, where $H_{\max}$ is again an upper bound on the length of an optimal schedule. For the load jobs, $H_t = \{r_t + 1, \ldots, \overline{d}_t - p_t - 1\}$ if $t \in T_L \cap T^1$, $H_t = \{r_t + 2, \ldots, \overline{d}_t - p_t\}$ if $t \in T_L \cap T^2$ and $H_t = \{r_t + 2 + p_t, \ldots, \overline{d}_t\}$ if $t \in T_L \cap T^3$.

# jobs	# gates	–	adapted form.	– + (6.8)	– + (6.9)	adapted form. + (6.10)
3	2	1.04 s	1.54 s	0.78 s	1.03 s	1.55 s
5	2	0.79 s	1.54 s	0.76 s	0.77 s	1.55 s
7	2	1.02 s	2.09 s	1.04 s	1.02 s	2.32 s
7	3	0.79 s	1.55 s	0.79 s	0.77 s	1.55 s
9	3	1.29 s	1.31 s	0.78 s	1.29 s	1.28 s
10	3	0.78 s	1.30 s	0.78 s	0.76 s	1.28 s
12	3	0.76 s	1.56 s	1.04 s	0.78 s	1.59 s
15	5	1.04 s	1.32 s	1.02 s	1.02 s	1.31 s
20	5	1.02 s	1.28 s	1.04 s	1.04 s	1.30 s
24	6	1.31 s	1.32 s	1.29 s	1.28 s	1.29 s
28	7	3.41 s	1.81 s	3.44 s	3.40 s	1.82 s
30	7	1.29 s	1.29 s	1.02 s	1.31 s	1.54 s
35	7	1.31 s	1.30 s	1.80 s	1.32 s	1.29 s
42	7	8.61 s	1.82 s	6.05 s	8.57 s	1.82 s
32	8	1.02 s	1.53 s	1.31 s	1.04 s	1.82 s
34	8	1.84 s	1.31 s	1.57 s	1.83 s	1.28 s
40	8	8.12 s	1.54 s	9.73 s	8.37 s	1.83 s
48	8	4.92 s	1.53 s	5.74 s	4.66 s	1.57 s
36	9	1.56 s	1.81 s	1.56 s	1.57 s	1.81 s
45	9	3.63 s	1.57 s	3.10 s	3.63 s	1.56 s
54	9	43.97 s	1.55 s	27.67 s	43.80 s	1.56 s
40	10	2.10 s	1.02 s	2.05 s	2.10 s	1.05 s
50	10	5.96 s	1.31 s	5.97 s	5.98 s	1.28 s
60	10	8.34 s	1.02 s	10.89 s	8.04 s	1.53 s

Table 8: Computation times for the valid inequalities for TI.

A linear formulation for the flexible flow-shop problem with these variables is the following:

$$\min \sum_{t \in T_U \cap T^2} w_t \left(\left(\sum_{u \in H_t} u x_{tu} \right) - 1 + p_t \right) + \sum_{t \in T_L \cap T^3} w_t \left(\sum_{u \in H_t} u x_{tu} \right) \tag{7.1}$$

subject to

$$\sum_{u \in H_t} x_{tu} = 1 \qquad \forall t \in T \tag{7.2}$$

$$\sum_{(t_1,t_2,t_3) \in J} \left(x_{t_1 u} + x_{t_3 u} + \sum_{v \le u} (x_{t_2 v} - x_{t_3 v}) \right) \le m \qquad \forall u \in \{1, \ldots, H_{\max}\} \tag{7.3}$$

$$\sum_{(t_1,t_2,t_3) \in J} (x_{t_1 u} + x_{t_3 u}) \le \tau \qquad \forall u \in \{1, \ldots, H_{\max}\} \tag{7.4}$$

$$x_{t_1 u} - x_{t_2,u+1} = 0 \qquad \forall (t_1, t_2, t_3) \in J; \forall u \in H_{t_1} \tag{7.5}$$

$$\sum_{u \in H_{t_3}} u x_{t_3 u} - \sum_{u \in H_{t_2}} u x_{t_2 u} \ge p_t \qquad \forall (t_1, t_2, t_3) \in J \tag{7.6}$$

$$x_{tu} \in \{0, 1\} \qquad \forall t \in T; \forall u \in H_t \tag{7.7}$$

The objective function (7.1) minimizes the weighted completion time of the stage-two unloading tasks and the stage-three loading tasks. Constraint (7.2) requires each task to be

processed exactly once, either on a gate or by a terminal tractor. Constraint (7.3) ensures that in each time period, at most m activities can be executed. This constraint is based on the fact that for each gate, the finished trailers need to have arrived at the parking lot before the start of the movement of the following trailer from the parking lot to the same gate. The stage-one and stage-three activities and the trailers for which the (un)loading has already started but the transport back to the parking lot has not yet begun, all count as gate occupation. Constraint (7.4) enforces the capacity of the terminal tractors. Constraints (7.5) and (7.6) implement the precedence constraints between the three stages. Finally, constraint (7.7) states that the decision variables x_{tu} are binary.

We observe that a stage-two task can always begin immediately after the corresponding stage-one task has been completed. Therefore, we can simply substitute all stage-two variables by an appropriate stage-one variable according to (7.5), so that the transportation towards the gate and the loading or unloading activities are treated as one single task with duration $1 + p_t$, needing a tractor only in the first period of its processing. For reasons of clarity, we have included in the model above all variables relating to the three stages. In our computational experiments, the aggregator of CPLEX eliminates these variables through substitution (see ILOG 2008).

The following inequality is valid for this formulation:

$$\sum_{v=1}^{u} x_{t_3 v} \leq \sum_{v=1}^{u-p_{t_2}} x_{t_2 v} \qquad \forall (t_1, t_2, t_3) \in J; \forall u \in \{1, \ldots, H_{\max}\} \tag{7.8}$$

Informally, this equation states that in fractional solutions, a stage-three task can only be started up to the fraction to which its stage-two task has been started. As an example, for a job (t_1, t_2, t_3) with $p_{t_2} = 4$, a possible solution is

$$x_{t_1 1} = x_{t_1 2} = x_{t_1 3} = \frac{1}{3}; \quad x_{t_2 2} = x_{t_2 3} = x_{t_2 4} = \frac{1}{3}; \quad x_{t_3 6} = x_{t_3 8} = 0.5$$

(with all other $x_{tu} = 0$). Constraint (7.6) holds for this solution, while constraint (7.8) is violated. For $u = 6$, constraint (7.8) is not respected because $x_{t_3 1} + x_{t_3 2} + x_{t_3 3} + x_{t_3 4} + x_{t_3 5} + x_{t_3 6} = 0.5$ while $x_{t_2 1} + x_{t_2 2} = \frac{1}{3}$. Note that these constraints (7.8) can also function as precedence constraints by themselves; they are equivalent to the disaggregated precedence constraints of Christofides et al. (1987). Although constraint (7.8) makes the mathematical-programming formulation theoretically stronger since constraints (7.2) and (7.8) together imply (7.6), Artigues et al. (2008) observe that the additional computation time needed to solve the larger linear program can counterbalance the significant improvement of the bound. Both constraint types will be tested empirically.

Define set $F = (T_U \cap T^2) \cup \{v \in (T_L \cap T^2) : \overline{d}_v > \max_{t \in T^1}\{r_t\}\}$ and let $p_{(t)}$ be the t^{th} largest stage-two duration among the jobs in F. Throughout Section 7, we use $H_{\max} = \max_{t \in T^1}\{r_t\} + p_{(1)} + p_{(2)} + \ldots + p_{(\lceil \frac{|F|}{m} \rceil)} + \lceil \frac{2|F|}{\tau} \rceil$, which constitutes an upper bound on the completion time of the last job in at least one optimal flow-shop schedule. Due to the blocking phenomenon, the earlier computation for $H_{\max}$ can no longer be followed. Instead, we can schedule the jobs in F in $\lceil \frac{|F|}{m} \rceil$ batches of size at most m, the length of which is upper-bounded by the values $p_{(\cdot)}$. In the worst case, all stage-one and stage-three tasks are sequentially scheduled on the τ terminal tractors, which gives rise to the final term $\lceil \frac{2|F|}{\tau} \rceil$ in the summation.

7.2 An alternative time-indexed formulation

Inspired by Tang and Xuan (2006), another time-indexed formulation with slightly different decision variables can be proposed: for all tasks $t \in T$ and time periods $u \in H_t$,

$$x_{tu} = \begin{cases} 1 & \text{if task } t \text{ is processed or blocked at time } u, \\ 0 & \text{otherwise,} \end{cases}$$

with the following re-definitions: $H_t = \{r_t + 1, \ldots, H_{\max} - p_t - 1\}$ if $t \in T_U \cap T^1$, $H_t = \{r_t + 2, \ldots, H_{\max} - 1\}$ if $t \in T_U \cap T^2$, $H_t = \{r_t + 2 + p_t, \ldots, H_{\max}\}$ if $t \in T_U \cap T^3$, $H_t = \{r_t + 1, \ldots, \overline{d}_t - p_t - 1\}$ if $t \in T_L \cap T^1$, $H_t = \{r_t + 2, \ldots, \overline{d}_t - 1\}$ if $t \in T_L \cap T^2$ and $H_t = \{r_t + 2 + p_t, \ldots, \overline{d}_t\}$ if $t \in T_L \cap T^3$. Moreover, for each task $t \in T_U \cap T^2$, y_t equals its tardiness, which is its contribution to the objective function, namely the time between the completion of the unloading task $t \in T_U \cap T^2$ and the due date d_t. For each $t \in T_L \cap T^3$, y_t is the negative of its earliness, which is the time between the completion of the transportation task $t \in T^3$ and the deadline $\overline{d}_t$.

The formulation is the following:

$$\min \sum_{t \in (T_U \cap T^2) \cup (T_L \cap T^3)} w_t y_t \tag{7.9}$$

subject to

$$\sum_{u \in H_t} x_{tu} = 1 \qquad \forall t \in T^1 \cup T^3 \tag{7.10}$$

$$\sum_{u \in H_t} x_{tu} \geq p_t \qquad \forall t \in T^2 \tag{7.11}$$

$$\sum_{(t_1,t_2,t_3) \in J} (x_{t_1 u} + x_{t_2 u} + x_{t_3 u}) \leq m \qquad \forall u \in \{1, \ldots, H_{\max}\} \tag{7.12}$$

$$\sum_{(t_1,t_2,t_3) \in J} (x_{t_1 u} + x_{t_3 u}) \leq \tau \qquad \forall u \in \{1, \ldots, H_{\max}\} \tag{7.13}$$

$$x_{t_1 u} + x_{t_2 u} - x_{t_2,u+1} - x_{t_3,u+1} \leq 0 \qquad \forall (t_1, t_2, t_3) \in J; \forall u \in \{1, \ldots, H_{\max}\} \tag{7.14}$$

$$u x_{tu} - d_t \leq y_t \qquad \forall t \in T_U \cap T^2; \forall u \in H_t \tag{7.15}$$

$$u x_{tu} - \overline{d}_t \leq y_t \qquad \forall t \in T_L \cap T^3; \forall u \in H_t \tag{7.16}$$

$$x_{tu} \in \{0, 1\} \qquad \forall t \in T; \forall u \in H_t$$

$$y_t \text{ unrestricted in sign} \qquad \forall t \in (T_U \cap T^2) \cup (T_L \cap T^3)$$

The objective function (7.9) minimizes for all tasks $t \in T$ the weighted contribution to the objective, which is computed via (7.15) and (7.16). Constraint (7.10) requires each transportation task to be processed exactly once, and constraint (7.11) enforces that each stage-two task t lasts at least as long as the processing time p_t. Constraints (7.12) and (7.13) impose the capacity of the gates and the terminal tractors, respectively. Constraint (7.14) implements the precedence constraints between the three stages.

7.3 Computational results

Table 9 shows that the time-indexed formulation with precedence constraint (7.8) performs significantly better than the alternative formulations, especially for medium-sized instances. Uetz (2001) finds that there exist instances for which the LP relaxation of a time-indexed formulation for a project scheduling problem with precedence constraints (7.8) provides lower bounds that are up to 75% higher than a relaxation with constraints (7.6), which is probably one of the major reasons for the difference in performance also in our setting. Adding other precedence constraints as valid inequalities to the different formulations does not improve the running times. We observe that a higher number of terminal tractors (even two) frequently decreases the required computational effort.

Table 10 explores the limits of the instance sizes that can be solved to guaranteed optimality within acceptable running times with the best formulation. Compared to the parallel-machine instances, some of the deadlines have been enlarged to create feasible instances. More details on these changes as well as the instances themselves can be found at the earlier-mentioned website. In the examined instances, the optimal objective-function value seems to be concavely decreasing with τ (indicating a decreasing marginal benefit of additional terminal tractors). For the instances that were not solved within one hour, the objective values for the best solution found are close to the lower bound produced by CPLEX OPL, which is the optimal objective value of the LP relaxation extended with some valid inequalities. This gap, defined as $\frac{\text{UB}-\text{LB}}{\text{LB}} * 100\%$, is shown in Table 11, where UB and LB stand for upper and lower bound, respectively.

Note that when the starting times of the stage-two activities are fixed, the problem at hand can be modeled as an assignment problem due to the fact that transportation times are unit. We have tested the formulation in which we do not impose an integrality

# jobs	# gates	# tractors	(7.6)	(7.8)	alternative
3	2	1	2.60 s	2.59 s	3.89 s
7	3	1	3.12 s	2.33 s	13.46 s
12	3	1	5.97 s	2.84 s	$>$ 1 h
15	5	1	7.55 s	4.94 s	$>$ 1 h
15	5	2	3.62	2.58 s	$>$ 1 h
20	5	1	3548 s	23.12 s	$>$ 1 h
20	5	2	10.39 s	3.11 s	$>$ 1 h
24	6	1	$>$ 1 h	25.50 s	$>$ 1 h
24	6	2	7.80 s	4.65 s	$>$ 1 h
24	6	3	21.37 s	3.63 s	$>$ 1 h
28	7	1	4.43 s	4.66 s	$>$ 1 h
28	7	2	25.77 s	9.61 s	$>$ 1 h
28	7	3	5.96 s	3.67 s	$>$ 1 h
30	7	1	68.40 s	39.56 s	$>$ 1 h
30	7	2	7.26 s	4.99 s	$>$ 1 h
30	7	3	5.44 s	3.92 s	$>$ 1 h
35	7	1	146.95 s	53.50 s	$>$ 1 h
35	7	2	105.57 s	25.02 s	$>$ 1 h
35	7	3	40.29 s	6.00 s	$>$ 1 h

Table 9: Computation times for the extended time-indexed formulations.

# jobs	# gates	# tractors	(7.8)	# jobs	# gates	# tractors	(7.8)
42	7	1	494.96 s	32	8	1	6.27 s
42	7	2	27.81 s	32	8	2	12.73 s
42	7	3	31.48 s	32	8	3	4.19 s
34	8	1	17.46 s	40	8	1	175.52 s
34	8	2	12.21 s	40	8	2	47.35 s
34	8	3	5.45 s	40	8	3	5.19 s
48	8	1	> 1 h	36	9	1	20.14 s
48	8	2	79.56 s	36	9	2	17.77 s
48	8	3	79.96 s	36	9	3	4.72 s
45	9	1	15.58 s	54	9	1	109.71 s
45	9	2	12.26 s	54	9	2	2068.52 s
45	9	3	7.79 s	54	9	3	17.68 s
40	10	1	68.35 s	50	10	1	61.21 s
40	10	2	22.25 s	50	10	2	174.25 s
40	10	3	17.67 s	50	10	3	26.29 s
60	10	1	75.47 s	44	11	1	31.43 s
60	10	2	149.55 s	44	11	2	21.33 s
60	10	3	44.73 s s	44	11	3	15.08 s
55	11	1	> 1 h	66	11	1	106.38 s
55	11	2	258.41 s	66	11	2	> 1 h
55	11	3	30.39 s	66	11	3	441.00 s
48	12	1	42.77 s	60	12	1	104.90 s
48	12	2	> 1 h	60	12	2	> 1 h
48	12	3	15.86 s	60	12	3	49.55 s
72	12	1	199.89 s	52	13	1	85.00 s
72	12	2	123.05 s	52	13	2	46.60 s
72	12	3	551.91 s	52	13	3	56.96 s
65	13	1	229.88 s	78	13	1	260.26 s
65	13	2	73.60 s	78	13	2	> 1 h
65	13	3	184.23 s	78	13	3	262.74 s
56	14	1	99.28 s	70	14	1	263.53 s
56	14	2	287.81 s	70	14	2	> 1 h
56	14	3	47.34 s	70	14	3	183.78 s
84	14	1	*	60	15	1	> 1 h
84	14	2	> 1 h	60	15	2	129.98 s
84	14	3	938.72 s	60	15	3	217.51 s
75	15	1	*	90	15	1	*
75	15	2	129.52 s	90	15	2	2147.60 s
75	15	3	1811.79 s	90	15	3	> 1 h

Table 10: Running times for larger instances. An asterisk '*' means that CPLEX OPL was not able to solve this instance because of memory problems.

constraint on stage-one and stage-three decision variables, but the computational effort did not significantly improve.

8 Conclusions

In this article, we have compared various mathematical formulations for a parallel-machine scheduling problem representing a dock assignment problem, where trailers are assigned to

# jobs	# gates	# tractors	gap
48	8	1	0.12%
55	11	1	0.02%
66	11	2	0.48%
48	12	2	0.04%
60	12	2	0.60%
78	13	2	0.22%
70	14	2	0.15%
84	14	2	1.64%
60	15	1	0.02%
90	15	3	0.08%

Table 11: Gap achieved by CPLEX OPL after one hour

gates for loading or unloading. Our work includes a study of different ways for dealing with symmetry and the addition of multiple sets of valid inequalities. The adapted time-indexed formulation performs significantly better than the other formulations (which were assignment-based and flow-based). The parallel-machine model is subsequently extended to a three-stage flexible flow shop, where the first and the third stage consist of the movement of the trailers from a parking lot to the gates and back, respectively. With this time-indexed formulation, we are able to solve small to medium-sized instances to guaranteed optimality within reasonable CPU times. Further research is needed to produce optimal solutions for real-life instances within acceptable running times; it seems realistic to assume that one should resort to large-scale heuristic procedures (e.g., meta-heuristics) for finding high-quality solutions for large instances.

References

C. Artigues, S. Demassey, and E. Néron. *Resource-constrained project scheduling*. Control Systems, Robotics and Manufacturing Series. ISTE Ltd, 2008.

J.F. Bard and S. Rojanasoonthon. A branch-and-price algorithm for parallel machine scheduling with time windows and job priorities. *Naval Research Logistics*, 53:24–44, 2006.

L. Bigras, M. Gamache, and G. Savard. Time-indexed formulations and the total weighted tardiness problem. *INFORMS Journal on Computing*, 20(1):133–142, 2008.

E.K. Bish, T. Leong, C. Li, J.W.C. Ng, and D. Simchi-Levi. Analysis of a new vehicle scheduling and location problem. *Naval Research Logistics*, 48:363–385, 2001.

E.K. Bish, F.Y. Chen, Y.T. Leong, B.L. Nelson, J.W.C. Ng, and D. Simchi-Levi. Dispatching vehicles in a mega container terminal. *OR Spectrum*, 27:491–506, 2005.

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

J. Böse, T. Reiners, D. Steenken, and S. Voss. Vehicle dispatching at seaport container terminals using evolutionary algorithms. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, pages 1–10, 2000.

N. Boysen, M. Fliedner, and A. Scholl. Scheduling inbound and outbound trucks at cross docking terminals. *OR Spectrum*, 32:135–161, 2010.

Z. Chen and W.B. Powell. Solving parallel machine scheduling problems by column generation. *INFORMS Journal on Computing*, 11(1):78–94, 1999.

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

N. Christofides, R. Alvarez-Valdés, and J.M. Tamarit. Project scheduling with resource constraints: a branch-and-bound approach. *European Journal of Operational Research*, 29 (3):262–273, 1987.

Y. Crama and F.C.R. Spieksma. Scheduling jobs of equal length: complexity, facets and computational results. *Mathematical Programming*, 72:207–227, 1996.

Y. Dallery and S.B. Gershwin. Manufacturing flow line systems: a review of models and analytical results. *Queueing Systems*, 12:3–94, 1992.

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

M.E. Dyer and L.A. Wolsey. Formulating the single machine sequencing problem with release dates as a mixed integer problem. *Discrete Applied Mathematics*, 26:255–270, 1990.

J.N.C. Gupta, K. Krüger, V. Lauff, F. Werner, and Y.N. Sotskov. Heuristics for hybrid flow shops with controllable processing times and assignable due dates. *Computers & Operations Research*, 29:1417–1439, 2002.

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

ILOG. *ILOG CPLEX 11.0 user's manual*. ILOG, Inc., 2008. available online at http://www.decf.berkeley.edu/help/apps/ampl/cplex-doc/.

V. Jain and I.E. Grossmann. Algorithms for hybrid milp/cp models for a class of optimization problems. *INFORMS Journal on Computing*, 13(4):258–276, 2001.

R. Jans. Solving lotsizing problem on parallel identical machines using symmetry breaking constraints. *INFORMS Journal on Computing*, 21(1):123–136, 2008.

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

T. Kis and E. Pesch. A review of exact solution methods for the non-preemptive multiprocessor flowshop problem. *European Journal of Operational Research*, 164:592–608, 2005.

H. Kise, T. Shioyama, and T. Ibaraki. Automated two-machine flow-shop scheduling: a solvable case. *IIE Transactions*, 23(1):10–16, 1991.

J.K. Lenstra and A.H.G. Rinnooy Kan. Complexity of scheduling under precedence constraints. *Operations Research*, 26(1):22–35, 1978.

R. Linn and W. Zhang. Hybrid flow shop scheduling: a survey. *Computers and Industrial Engineering*, 37:57–61, 1999.

F. Margot. Symmetry in integer linear programming. 2008. Tepper working paper 2008 E-37.

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*, 179:1150–1165, 2009.

Z. Miao, A. Lim, and H. Ma. Truck dock assignment problem with operational time constraint within crossdocks. *European Journal of Operational Research*, 192:105–115, 2009.

O. Moursli and Y. Pochet. A branch-and-bound algorithm for the hybrid flowshop. *International Journal of Production Economics*, 64:113–125, 2000.

T. Nichi, Y. Hiranaka, and M. Inuiguchi. Lagrangian relaxation with cut generation for hybrid flowshop scheduling problems to minimize the total weighted tardiness. *Computers & Operations Research*, 37:189–198, 2010.

T. Oncan, I. Kuban Altinel, and G. Laporte. A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, 36:637–654, 2009.

C.D. Paternina-Arboleda, J.R. Montoya-Torres, M.J. Acero-Domingues, and M.C. Herrera-Hernandez. Scheduling jobs on a k-stage flexible flow-shop. *Annals of Operations Research*, 164:29–40, 2008.

I. Ribas, R. Leisten, and J.M. Framiñan. Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers & Operations Research*, 37:1439–1454, 2010.

R. Sadykov and L.A. Wolsey. Integer programming and constraint programming in solving a multimachine assignment scheduling problem with deadlines and release dates. *INFORMS Journal on Computing*, 18(2):209–217, 2006.

H.D. Sherali and J.C. Smith. Improving discrete model representations via symmetry considerations. *Management Science*, 47(10):1396–1407, 2001.

J.P. Sousa and L.A. Wolsey. A time indexed formulation of non-preemptive single machine scheduling problems. *Mathematical Programming*, 54:353–367, 1992.

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

M. Uetz. *Algorithms for deterministic and stochastic scheduling*. PhD thesis, Technische Universität Berlin, Germany, 2001.

J.M. van den Akker, C.A.J. Hurkens, and M.W.P. Savelsbergh. Time-indexed formulations for machine scheduling problems: Column generation. *INFORMS Journal on Computing*, 12(2):111–124, 2000.

A. Vignier, J.C. Billaut, and C. Proust. Hybrid flowshop scheduling problems: state of the art. *RAIRO Operations Research*, 33(2):117–183, 1999.

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