Ann Oper Res (2014) 213:3–25
DOI 10.1007/s10479-011-0971-7

Optimal solutions for a dock assignment problem with trailer transportation

Lotte Berghman · Roel Leus · Frits C.R. Spieksma

Published online: 17 September 2011

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 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

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

L. Berghman (✉) · R. Leus · F.C.R. Spieksma
Research Group ORSTAT, Katholieke Universiteit Leuven, Leuven, Belgium
e-mail: lotte.berghman@econ.kuleuven.be

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 stated 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 fourfold: (1) we identify the dock assignment problem with trailer transportation as a relevant problem; (2) we propose and compare various integer-programming (IP) formulations for the parallel-machine scheduling problem corresponding to stage two; (3) based on this comparison, we focus on a time-indexed formulation for the dock assignment problem that leads to good computational results for medium-size instances; and (4) some practical extensions of the general problem are described in which the gates are not all identical and where not all trailers need a tractor.

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 Sect. 3. Various IP formulations for stage two (parallel-machine scheduling) are studied in Sect. 4, the computational results of which are described in Sect. 5. A formal statement of the flexible flow-shop problem is given in Sect. 6, and the best performing formulation for the parallel-machine case is extended towards this setting. In Sect. 7 we present additional elements of the practical case encountered at Toyota as extensions of the more generic dock assignment problem with trailer transportation, propose a corresponding mathematical formulation and computationally test the effects of the changes. We round off the article with some conclusions in Sect. 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 a variable denotes 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 CPU 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 Truck and container scheduling

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. Cheong et al. (2010) consider a berth allocation problem which requires the determination of exact berthing times and positions of incoming ships in a container port. 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. Guan et al. (2010) study the crane scheduling problem for a vessel that is moored at a terminal. A flow formulation with so-called "non-crossing constraints", which is able to solve small instances, is given. Moreover, both exact and heuristic solution approaches are developed. A detailed literature review can be found in Stahlbock and Voß (2008) and Bierwirth and Meisel (2010).

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 heuristic (see, e.g., Gupta et al. 1997; Tang and Xuan 2006; Nichi et al. 2010) and optimal approaches (see for instance 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 CPU 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.

A flowshop where a job may return one or more times to any machine and thus revisit a machine, is called a reentrant flow shop. Although these flow shops are usually operated and scheduled as general jobs shops (Graves et al. 1983), some dedicated algorithms can be found in literature (see, e.g., Chen et al. 2007; Choi and Kim 2008). Due to the specificity of our dock assignment problem, however, we will develop new models for producing optimal solutions.

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 $t \in T^2$ 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

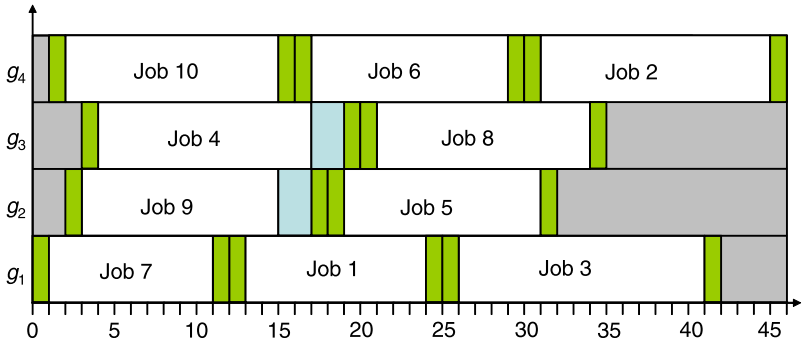

Fig. 1 A feasible schedule for the example instance. Each rectangle labeled 'Job i' represents the stage-two task of the particular job; g_k is the k^{th} gate

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 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 an optimal schedule that minimizes total completion time for a set of tasks with release times is NP-hard, even on a single processor (problem $1|r_j|\sum C_j$, see Lenstra et al. 1977). Consequently, finding an 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 Fig. 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, Sects. 4 and 5) 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. This is indeed the core of our problem: we point out that, in case of *given* starting times of the loading and unloading activities, the only activities to be planned are the transportation activities and as the transportation activities themselves have a unit duration, the remaining

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

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

problem is an assignment problem, as follows. On the one hand, there is a node for each transportation activity. On the other hand, there is a node for each time period and each tractor. Transportation activities are only linked with those right-hand side nodes for which the time period belongs to their time window. For stage-one activities, the time window starts at the ready time of the considered trailer and ends at the starting time of the corresponding stage-two activity. For stage-three activities, the time window starts at the completion time of the corresponding stage-two activity and ends at the deadline. Observe that solving the parallel-machine scheduling problem related to the loading and unloading activities in a first phase, and solving the corresponding assignment problem to schedule the tractors, given the parallel-machine solution, in a second phase does not solve the dock assignment problem to guaranteed optimality. Moreover, it might be the case that the problem in the second phase has no feasible solution although there is a feasible solution for the dock assignment problem.

Thus, our goal in Sects. 4 and 5 is to identify a formulation that can find starting times for the stage-two activities for 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 $P|r_j, \overline{d}_j|\sum w_j C_j$ in the standard three-field notation. Our findings for this case will be useful for producing an appropriate IP formulation for the flexible flow shop in Sect. 6. In the parallel-machine setting, each job's ready time, weight and 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 Fig. 2.

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 an assignment-based formulation (cfr. Jain and Grossmann 2001), a flow formulation (cfr. Bard and Rojanasoonthon 2006) and a time-indexed formulation (cfr. Dyer and Wolsey 1990). The computational results of the different formulations will be given in Sect. 5.

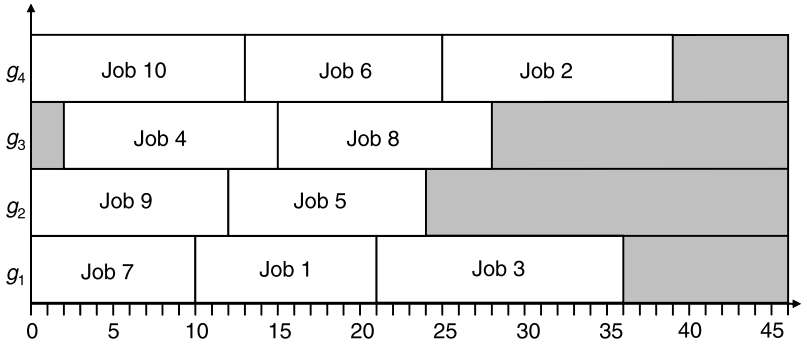

Fig. 2 A feasible parallel-machine schedule for the example instance

4.1 Assignment-based formulation

For all tasks $t \in T^2$ and for every gate $g \in G$, we have

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

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.

Our assignment-based formulation (henceforth 'AB' for short) for the considered parallel-machine problem is the following:

$$\min \sum_{t \in T^2} w_t C_t \tag{1}$$

subject to

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

$$x_t^g + x_u^g - z_{tu} - z_{ut} \leq 1 \qquad \forall \{t, u\} \subset T^2; \forall g \in G \tag{3}$$

$$x_t^g + x_u^h + z_{tu} + z_{ut} \leq 2 \qquad \forall \{t, u\} \subset T^2; \forall \{g, h\} \subset G \tag{4}$$

$$C_t - (1 - z_{tu})M \leq C_u - p_u \qquad \forall \{t, u\} \subset T^2 \tag{5}$$

$$r_t \leq C_t - p_t \qquad \forall t \in T^2 \tag{6}$$

$$C_t \leq \overline{d}_t \qquad \forall t \in T_L \cap T^2 \tag{7}$$

$$x_t^g \in \{0, 1\} \qquad \forall t \in T^2; \forall g \in G \tag{8}$$

$$z_{tu} \in \{0, 1\} \qquad \forall \{t, u\} \subset T^2 \tag{9}$$

$$C_t \geq 0 \qquad \forall t \in T^2 \tag{10}$$

The objective function (1) minimizes the total weighted completion time of all tasks $t \in T^2$. Constraint (2) demands that each task be assigned to exactly one gate. Constraint (3) ensures that if tasks t and u are assigned to the same gate g, then one must be processed before the other and constraint (4) demands that the sequencing variables z_{tu} and z_{ut} both be zero if tasks t and u are assigned to different gates. Constraint (5) ensures 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 (6) imposes that a task cannot start before its ready time and constraint (7) demands that a task be finished by its deadline. Finally, constraints (8) and (9) state that the decision variables z_{tu} and x_t^g are binary and constraint (10) requires all completion times to be non-negative. 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.

Mathematical formulations in which tasks are assigned to positions on machines also appear in literature. Preliminary experiments have indicated that the above formulation performs better. Details can be found in our working paper Berghman et al. (2010).

4.2 Flow formulation

In the following formulation, subsequently referred to as formulation F (for 'flow'), 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 $\{t, u\} \subset T_0^2$,

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

Similarly 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

$$\sum_{u\in T_0^2\setminus\{t\}} x_{tu} = 1 \qquad \forall t \in T^2 \tag{11}$$

$$\sum_{t\in T_0^2} x_{t_0 t} = m \tag{12}$$

$$\sum_{u\in T_0^2\setminus\{t\}} x_{ut} - \sum_{u\in T_0^2\setminus\{t\}} x_{tu} = 0 \qquad \forall t \in T^2 \tag{13}$$

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

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

$$C_t \leq \overline{d}_t \qquad \forall t \in T_L \cap T^2$$
$$x_{tu} \in \{0,1\} \qquad \forall \{t,u\} \subset T_0^2$$
$$C_t \geq 0 \qquad \forall t \in T^2$$

Constraint (11) 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 (12) limits the number of initial tasks. These constraints (11) and (12) indirectly specify that each machine can process at most one task at a time. Constraint (13) 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. The remaining constraints correspond with formulation AB. 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.

Note that an alternative flow formulation can be set up with decision variables $x_{tu}^g = 1$ if task t is the immediate predecessor of task u at gate g and $= 0$ otherwise, for all tasks $\{t,u\} \subset T_0^2$ and for every gate $g \in G$. Constraint (12) is then stated for each machine separately, with right-hand side equal to one. This induces issues related to symmetry, however (multiple essentially equivalent solutions, only one of which needs to be considered, are feasible). We have looked into various ways for eliminating this symmetry, including the addition of multiple so-called 'symmetry-breaking' constraints, in the working paper Berghman et al. (2010). The above model came out as leading to the best computational results.

4.3 Time-indexed formulation

The time-indexed formulation TI relies on a discretization of the planning horizon, for the length of which we use a practical upper bound $H_{\max}$ (see Sect. 5). The formulation is based on Dyer and Wolsey (1990) but adapted for the parallel-machine scheduling problem. For all tasks $t \in T^2$ and for all time periods $u \in H_t$,

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

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:

$$\min \sum_{t \in T^2} w_t \left(\left(\sum_{u \in H_t} u x_{tu} \right) - 1 + p_t \right) \tag{14}$$

subject to

$$\sum_{u \in H_t} x_{tu} = 1 \qquad \forall t \in T^2 \tag{15}$$

$$\sum_{t \in T^2} \sum_{v=u-p_t+1}^{u} x_{tv} \leq m \qquad \forall u \in \{1, \ldots, H_{\max}\} \tag{16}$$

$$x_{tu} \in \{0,1\} \qquad \forall t \in T^2; \forall u \in H_t \tag{17}$$

The objective function (14) has a similar interpretation as before. Constraint (15) requires each task to be started exactly once. Constraint (16) ensures that in a given time period u, only m tasks can be executed. Here and below, decision variables that are undefined because of the time windows do not appear in the model. Finally, constraint (17) 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. Note that by the choice of the decision variables, symmetry is again avoided, which would not be the case for the alternative choice x^g_{tu} that decides on the start of task t in period u at gate g; detailed comparisons (Berghman et al. 2010) indicate that the latter choice is less preferable.

In order to potentially strengthen this formulation, we consider the following valid inequalities:

$$\sum_{v\in Q_{t,u,\Delta}} x_{tv} + \sum_{s\in T^2\setminus\{t\}} \sum_{v\in Q'_{s,u,\Delta}} x_{sv} \le m$$
$$\forall t \in T^2; \forall u \in \{1,\ldots,H_{\max}\}; \forall \Delta \in \{2,\ldots,\overline{p}_t\}, \tag{18}$$

where $\overline{p}_t = \max_{s\in T^2\setminus\{t\}}\{p_s\}$, $Q_{t,u,\Delta} = [u - p_t + 1, u + \Delta - 1]$ and $Q'_{s,u,\Delta} = [u + \Delta - p_s, u]$ for all $s \neq t$ such that $p_s \ge \Delta$ and $Q'_{s,u,\Delta} = \emptyset$ otherwise. The validity of these inequalities can be seen using integer rounding: consider two constraints (16), one for period u and one for period $u + \Delta - 1$, and consider one constraint (15) for some specific task t. Summing these three constraints, each with coefficient $\frac{1}{2}$, and next rounding down the resulting left-hand side and right-hand side yields our inequality (18). This inequality is a generalization of inequalities that were mentioned by van den Akker (1994) (who describes the case $\Delta = 2$) and of inequalities considered in Crama and Spieksma (1996) (who assume equal processing times, and a single machine).

5 Computational results for the parallel-machine formulations

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 one-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 CPU 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

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

Table 2 CPU times for the linear formulations of Sect. 4

n	m	AB	F	TI
3	2	2.08 s	1.82 s	1.54 s
5	2	2.58 s	1.55 s	1.54 s
7	2	3.65 s	2.34 s	2.09 s
7	3	2.86 s	1.53 s	1.55 s
9	3	4.39 s	3.68 s	1.31 s
10	3	19.33 s	36.95 s	1.30 s
12	3	648.40 s	>1 h	1.56 s
15	5	>1 h	>1 h	1.32 s
20	5	>1 h	>1 h	1.28 s
24	6	>1 h	>1 h	1.32 s
28	7	>1 h	>1 h	1.81 s
30	7	>1 h	>1 h	1.29 s
35	7	>1 h	>1 h	1.30 s
42	7	>1 h	>1 h	1.82 s
32	8	>1 h	>1 h	1.53 s
34	8	>1 h	>1 h	1.31 s
40	8	>1 h	>1 h	1.54 s
48	8	>1 h	>1 h	1.53 s
36	9	>1 h	>1 h	1.81 s
45	9	>1 h	>1 h	1.57 s
54	9	>1 h	>1 h	1.55 s

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.

Concerning the TI formulation, we determine a tight value for $H_{\max}$ in the following way: 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 $r_{\max} + \lfloor \frac{\sum_{t \in F \setminus \{l_{\max}\}} p_t}{m} \rfloor + p_{l_{\max}}$, so throughout Sects. 4 and 5, we set $H_{\max} = r_{\max} + \lfloor \frac{\sum_{t \in F \setminus \{l_{\max}\}} p_t}{m} \rfloor + p_{l_{\max}}$.

Table 2 contains the results of our formulations on the small and medium-size test instances; here and below, a time limit of one hour is imposed on the CPU time. We find that the formulation TI ((14)–(17)) performs best, probably because of the tight LP bound. This result is in line with previous literature, and we will extend the TI formulation to model our gate assignment problem.

Out of the 21 instances considered in Table 2, there are six whose LP-relaxation of the TI-formulation is not integral. For these six instances, we added all inequalities (18) to the model. The solution of the corresponding LP relaxation improved for three out of these six

Table 3 Objective values for the LP relaxation and the IP of the time-indexed formulation of Sect. 4

n	m	LP ((14)–(17))	LP ((14)–(18))	IP
28	7	1121	1121	1122
42	7	4563.2	4563.3	4564
40	8	1743	1743	1744
48	8	2288.8	2289	2289
45	9	1904.3	1904.5	1905
54	9	2788.2	2788.2	2789

instances, closing, on average, over 20% of the gap between the LP-value and the value of the integer optimum (see Table 3). Since adding all inequalities to the model increases the CPU times needed to solve the IP, we chose not to do so for the computational experiments involving the flowshop model. Notice however, that there is a potential gain in solving the separation problem for these inequalities, and next only add violated inequalities to the model.

6 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.

6.1 Time-indexed formulation

As the 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 $H_t = \{r_t + 1, \ldots, H_{\max} - p_t - 1\}$ 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 $t \in T_L \cap T^3$. In Berghman et al. (2010), other choices for the decision variables of a time-indexed formulation are also tested, but the current choice led to the best results in terms of runtime. A linear formulation for the flexible flow-shop problem with these variables is the following:

$$\min \sum_{t \in T_U \cap T^2} w_t \left(\left(\sum_{u \in H_t} u x_{tu} \right) - 1 + p_t \right) + \sum_{t \in T_L \cap T^3} w_t \left(\sum_{u \in H_t} u x_{tu} \right) \tag{19}$$

subject to

$$\sum_{u \in H_t} x_{tu} = 1 \qquad \forall t \in T \tag{20}$$

$$\sum_{(t_1,t_2,t_3)\in J}\left(x_{t_1u}+x_{t_3u}+\sum_{v\le u}\left(x_{t_2v}-x_{t_3v}\right)\right)\le m \qquad \forall u\in\{1,\ldots,H_{\max}\} \tag{21}$$

$$\sum_{(t_1,t_2,t_3)\in J}\left(x_{t_1u}+x_{t_3u}\right)\le \tau \qquad \forall u\in\{1,\ldots,H_{\max}\} \tag{22}$$

$$x_{t_1u}-x_{t_2,u+1}=0 \qquad \forall (t_1,t_2,t_3)\in J;\forall u\in H_{t_1} \tag{23}$$

$$\sum_{u\in H_{t_3}} u x_{t_3u}-\sum_{u\in H_{t_2}} u x_{t_2u}\ge p_t \qquad \forall (t_1,t_2,t_3)\in J \tag{24}$$

$$x_{tu}\in\{0,1\} \qquad \forall t\in T;\forall u\in H_t \tag{25}$$

The objective function (19) minimizes the weighted completion time of the stage-two unloading tasks and the stage-three loading tasks. Constraint (20) requires each task to be processed exactly once, either on a gate or by a terminal tractor. Constraint (21) 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 (22) enforces the capacity of the terminal tractors. Constraints (23) and (24) implement the precedence constraints between the three stages. Finally, constraint (25) 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 (23), 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:

$$\sum_{v=1}^{u} x_{t_3v}\le \sum_{v=1}^{u-p_{t_2}} x_{t_2v} \qquad \forall (t_1,t_2,t_3)\in J;\forall u\in\{1,\ldots,H_{\max}\}. \tag{26}$$

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

$$x_{t_11}=x_{t_12}=x_{t_13}=\frac{1}{3}; \qquad x_{t_22}=x_{t_23}=x_{t_24}=\frac{1}{3}; \qquad x_{t_36}=x_{t_38}=0.5$$

(with all other $x_{tu}=0$). Constraint (24) holds for this solution, while constraint (26) is violated. For $u=6$, constraint (26) is not respected because $x_{t_31}+x_{t_32}+x_{t_33}+x_{t_34}+x_{t_35}+x_{t_36}=0.5$ while $x_{t_21}+x_{t_22}=\frac{1}{3}$. Note that these constraints (26) can also function as precedence constraints by themselves; they are equivalent to the disaggregated precedence constraints of Christofides et al. (1987). Although constraint (26) makes the mathematical-programming formulation theoretically stronger since constraints (20) and (26) together imply (24), Artigues et al. (2008) observe that the additional CPU time needed to solve the

Table 4 CPU times for the time-indexed formulations of Sect. 6.1

n	m	τ	(24)	(26)
3	2	1	2.60 s	2.59 s
7	3	1	3.12 s	2.33 s
12	3	1	5.97 s	2.84 s
15	5	1	7.55 s	4.94 s
15	5	2	3.62	2.58 s
20	5	1	3548 s	23.12 s
20	5	2	10.39 s	3.11 s
24	6	1	>1 h	25.50 s
24	6	2	7.80 s	4.65 s
24	6	3	21.37 s	3.63 s
28	7	1	4.43 s	4.66 s
28	7	2	25.77 s	9.61 s
28	7	3	5.96 s	3.67 s
30	7	1	68.40 s	39.56 s
30	7	2	7.26 s	4.99 s
30	7	3	5.44 s	3.92 s
35	7	1	146.95 s	53.50 s
35	7	2	105.57 s	25.02 s
35	7	3	40.29 s	6.00 s

larger linear program can counterbalance the significant improvement of the bound. Both constraint types will be tested empirically in Sect. 6.2.

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 Sect. 6, we use $H_{\max} = \max_{t \in T^1}\{r_t\} + p_{(1)} + p_{(2)} + \cdots + p_{(\lceil \frac{|F|}{m} \rceil)} + \lceil \frac{2|F|}{\tau} \rceil$, 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.

6.2 Computational results for the flexible flow shop

Table 4 shows that the time-indexed formulation with precedence constraint (26) performs significantly better than the alternative formulation with precedence constraint (24), especially for medium-sized instances. In our case, the formulation with (26) gives a strictly better lower bound for 18 out of the 19 instances, which is probably one of the major reasons for the difference in performance. This is in line with the observations of Uetz (2001). We observe that a higher number of terminal tractors (even two) frequently decreases the required computational effort.

In order to investigate the computational performance for larger instances, we have created new instances with $m \in \{10, \ldots, 15\}$ and $|T| \in \{4m, 5m, 6m\}$. The ready times for the unloading jobs are integers randomly drawn from $\{0, \ldots, 25\}$ (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 $\{\beta - 10, \ldots, \beta + 10\}$, $\beta = \frac{0.5 \sum_{t \in T^2} p_t}{m}$ and $l_{\max} = \arg\max_{t \in T^2}\{p_t\}$. 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.

Table 5 explores the limits of the instance sizes that can be solved to guaranteed optimality within acceptable running times with the best formulation (with precedence constraints (26)). The objective value of the LP relaxation (LP), the objective function of the integral formulation (obj val) and the CPU time for the integral formulation (time) are given. As at day X a schedule for day $X + 1$ is created at Toyota, one hour of CPU time is acceptable. The formulation is rather strong: the difference between the objective function of the LP relaxation and the objective function of the integral formulation is on average 0.16%. Sometimes, both values are even the same. Up to some 90 trailers and 15 gates, most instances can be solved to guaranteed optimality within the time limit of one hour. For a fixed number of gates, the CPU times tend to increase when the number of jobs increases. Clearly, when increasing the number of tractors, the objective cannot increase. It turns out that for our instances the gain induced by adding a third tractor is smaller than the gain of the second tractor. For the instances that were not solved within one hour, the objective values for the best solution found are close to the lower bound reported by CPLEX OPL (ILOG 2008). This gap, defined as $\frac{\text{UB}-\text{LB}}{\text{LB}} * 100\%$, is shown in Table 6, where UB and LB stand for upper and lower bound, respectively.

7 Extensions

In this section, we study two practical extensions of the presented dock assignment problem with trailer transportation. A detailed problem statement will be given, followed by a linear formulation and extensive computational results.

In both extensions, the terminal tractors are not always used. For every trailer, it is known at the start of the planning horizon whether it will remain coupled to the truck or be uncoupled. Coupled trailers are brought to the gate by the trucker, who waits until the load/unload operation is completed and then directly takes the trailer away. Uncoupled trailers are dropped off by the trucker at the parking lot and transferred to a gate by a terminal tractor afterwards, in line with the description in Sect. 3. After unloading or loading at the gate, the uncoupled trailer is moved back to the parking lot by a tractor, where it will be picked up by a trucker later on.

7.1 Preferred gates

The first extension deals with the preference to load or unload products at a gate near the position of the products in the warehouse. Every trailer has a preferred subset of gates where it should be loaded or unloaded, depending on the products it carries. The gates in each subset are treated as identical parallel machines. Loading or unloading a trailer at a gate that is not part of the preferred subset is possible but should not be encouraged.

Table 5 Objective values and CPU times for larger instances for the best time-indexed formulation of Sect. 6.1 where n, m and τ stands for number of jobs, gates and terminal tractors, respectively. An asterisk '*' means that OPL CPLEX was not able to solve the instance to guaranteed optimality within one hour; a double asterisk '**' means that CPLEX OPL was not able to solve this instance because of memory problems

n	m	τ	LP	obj val	time (s)	n	m	τ	LP	obj val	time (s)
42	7	1	7388.1	7447	494.96	32	8	1	1816.63	1822	6.27
42	7	2	5854.03	5855	27.81	32	8	2	1433.36	1434	12.73
42	7	3	5694.89	5706	31.48	32	8	3	1454.33	1455	4.19
34	8	1	2073.47	2097	17.46	40	8	1	2983.97	2999	175.52
34	8	2	1466.22	1467	12.21	40	8	2	2204.52	2207	47.35
34	8	3	1421.8	1422	5.45	40	8	3	2151	2151	5.19
48	8	1	4251.02	*	>1 h	36	9	1	2466.93	2473	20.14
48	8	2	2909	2912	79.56	36	9	2	1783.4	1785	17.77
48	8	3	2842.5	2845	79.96	36	9	3	1721	1721	4.72
45	9	1	3712.5	3742	15.58	54	9	1	5391	5395	109.71
45	9	2	2413.45	2414	12.26	54	9	2	3506.61	3512	2068.52
45	9	3	2349.6	2351	7.79	54	9	3	3422.62	3423	17.68
40	10	1	2573.56	2583	68.35	50	10	1	4835.5	4870	61.21
40	10	2	1682	1682	22.25	50	10	2	2990.75	2995	174.25
40	10	3	1627	1627	17.67	50	10	3	2910.5	2911	26.29
60	10	1	7004	7030	75.47	44	11	1	3554.2	3571	31.43
60	10	2	4262.68	4266	149.55	44	11	2	2155.5	2168	21.33
60	10	3	4155	4155	44.73	44	11	3	2043	2043	15.08
55	11	1	5815.5	*	>1 h	66	11	1	8791	8791	106.38
55	11	2	3170.7	3181	258.41	66	11	2	4549.59	*	>1 h
55	11	3	3057.97	3059	30.39	66	11	3	4414.6	4416	441.00
48	12	1	4518.51	4542	42.77	60	12	1	7151	7151	104.90
48	12	2	2755.17	*	>1 h	60	12	2	3803.5	*	>1 h
48	12	3	2577.33	2579	15.86	60	12	3	3593.57	3594	49.55
72	12	1	9164.5	9165	199.89	52	13	1	4444	4444	85.00
72	12	2	4790.5	4799	123.05	52	13	2	2394	2402	46.60
72	12	3	4536.38	4537	551.91	52	13	3	2207.44	2208	56.96
65	13	1	7810	7810	229.88	78	13	1	9984.3	9999	260.26
65	13	2	3973	3973	73.60	78	13	2	5044.5	*	>1 h
65	13	3	3603	3604	184.23	78	13	3	4538.17	4539	262.74
56	14	1	5656	5656	99.28	70	14	1	9420.5	9421	263.53
56	14	2	3000.83	3002	287.81	70	14	2	4963.5	*	>1 h
56	14	3	2643.57	2644	47.34	70	14	3	4301.88	4302	183.78
84	14	1	12740	**	**	60	15	1	6975	*	>1 h
84	14	2	6359.75	*	>1 h	60	15	2	3510.5	3511	129.98
84	14	3	5381.83	5383	938.72	60	15	3	2936	2941	217.51
75	15	1	**	**	**	90	15	1	**	**	**
75	15	2	5644.89	5651	129.52	90	15	2	8249.5	8272	2147.60
75	15	3	4547.44	4552	1811.79	90	15	3	6020.67	*	>1 h

Table 6 Gap achieved by CPLEX OPL after one hour

n	m	τ	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%

We model this problem setting as a three-stage flexible flow shop, in line with the previous section, although the extensions require minor modifications. Set T is now partitioned as follows: $T = T_C \cup T_U \cup T_L$, where T_C contains all operations related to trailers that will remain coupled to the truck, T_U gathers all operations related to trailers to be unloaded that will be decoupled at the parking lot at the arrival time and T_L contains all operations related to uncoupled trailers to be loaded. The set J can be partitioned in a similar way as $J = J_C \cup J_U \cup J_L$. Each operation $t \in T_C \cap T^1$ has a ready time r_t, which equals the planned arrival time of the trailer. Each operation $t \in T_C \cap T^3$ has a deadline $\overline{d}_t$, which creates a time window for the coupled trailers as we do not want truckers to stay too long at the plant. The set G, containing $m < n$ gates, can be partitioned into o different gate sets $G_1, G_2, \ldots, G_o$. Each trailer t has one preferred gate set $G_{f(t)}$. Value $\Delta(G_i, G_j)$ is a measure for the physical distance between gate sets G_i and G_j. Our problem consists in scheduling the operations in such a way that the moment at which the coupled trailers $t \in T_C \cap T^3$ leave the plant is minimized, together with the contribution to the objective function of the uncoupled trailers. Loading or unloading trailers at a gate that is not included in the preferred gate set is penalized.

For all operations $t \in T$, for all gate sets $G_i \subset G$ and for all time periods $u \in H_t$,

$$x_{tu}^i = \begin{cases} 1 & \text{if the corresponding trailer is (un)loaded at gate set } G_i \text{ and} \\ & \text{if operation } t \text{ starts in time period } u, \\ 0 & \text{otherwise,} \end{cases}$$

with $H_t = \{r_t + 1, \ldots, \overline{d}_t - p_t - 1\}$ if $t \in T^1 \cup T_C$, $H_t = \{r_t + 2, \ldots, \overline{d}_t - p_t\}$ if $t \in T^2 \cup T_C$ and $H_t = \{r_t + 2 + p_t, \ldots, \overline{d}_t\}$ if $t \in T^3 \cup T_C$. The time windows for the uncoupled trailers remain unchanged.

A linear formulation is the following:

$$\min z = \sum_{t \in T_U \cap T^2} \left(w_t \left(\left(\sum_{u \in H_t} u \sum_{G_i \subset G} x_{tu}^i \right) - 1 + p_t \right) \right) + \sum_{t \in (T_L \cup T_C) \cap T^3} \left(w_t \left(\sum_{u \in H_t} u \sum_{G_i \subset G} x_{tu}^i \right) \right) + \alpha \sum_{t \in T} \sum_{u \in H_t} \sum_{G_i \subset G} x_{tu}^i \Delta(G_{f(t)}, G_i) \quad (27)$$

subject to

$$\sum_{G_i \subset G} \sum_{u \in H_t} x^i_{tu} = 1 \qquad \forall t \in T \tag{28}$$

$$\sum_{(t_1,t_2,t_3)\in J} \left(x^i_{t_1 u} + x^i_{t_3 u} + \sum_{v \le u} \left(x^i_{t_2 v} - x^i_{t_3 v} \right) \right) \le |G_i| \qquad \forall G_i \subset G; \forall u \in \{1, \ldots, H_{\max}\} \tag{29}$$

$$\sum_{(t_1,t_2,t_3)\in (J_L \cup J_U)} \sum_{G_i \subset G} \left(x^i_{t_1 u} + x^i_{t_3 u} \right) \le \tau \qquad \forall u \in \{1, \ldots, H_{\max}\} \tag{30}$$

$$x^i_{t_1 u} - x^i_{t_2, u+1} = 0 \qquad \forall (t_1, t_2, t_3) \in J; \forall u \in H_{t_1}; \forall G_i \subset G \tag{31}$$

$$\sum_{v=1}^{u} x^i_{t_3 v} - \sum_{v=1}^{u-p_{t_2}} x^i_{t_2 v} \le 0 \qquad \forall (t_1, t_2, t_3) \in J; \forall u \in H_{t_3}; \forall G_i \subset G \tag{32}$$

$$x^i_{tu} \in \{0, 1\} \quad \forall t \in T; \forall u \in H_t; \forall G_i \subset G \tag{33}$$

The first part of the objective function (27) (the first two terms in the summation) minimizes the weighted completion times as described in the previous subsection. The second part penalizes trailers that are not (un)loaded at their preferred gate set. The penalization is proportional to the distance between the preferred gate set and the assigned gate set, and is weighted by factor α. Constraint (28) states that all tasks have to be executed exactly once. Constraint sets (29) and (30) are the capacity constraints. According to constraint (29), for each gate set G_i, at most $|G_i|$ gates may be either executing an operation or be blocked during each time period. Constraint (30) states that at most τ stage-one or stage-three tasks of uncoupled trailers may be scheduled in parallel during each time period. Constraint sets (31) and (32) represent the precedence constraints. Each stage-two task starts immediately after its stage-one task, and a stage-three activity cannot start before the corresponding stage-two task is finished.

As the instances tested in the previous section are based on ones we found in literature and therefore do not necessarily reflect the situation at Toyota, the instances used here are generated in a slightly different way. More particularly, the length of the planning horizon is $H_{\max} = 96$, representing a shift or a working day. The weights of the tasks are randomly selected out of $\{1, 2, 3\}$ and the processing times p_t are chosen as $1 + X$ with X binomially distributed with parameters 16 (trials) and 0.5 (probability of success). 25% of the trailers remains coupled. These trailers have a ready time r_t randomly selected out of $\{0, \ldots, 64\}$ (uniformly distributed) and a deadline $\overline{d}_t = r_t + p_t + 18$, such that the truckers do not have to wait too long at the site. Another 30% of the trailers are decoupled and need to be unloaded. These trailers have a ready time r_t similar to the decoupled trailers and a deadline $\overline{d}_t = 96$. The remaining trailers are decoupled and need to be loaded. These trailers have a ready time $r_t = 0$, and the deadlines for the these tasks are set to $\max\{\overline{d}'_t, r_t + 2 + p_{l_{\max}}\}$ with $\overline{d}'_t$ uniformly distributed on $\{\beta - 32, \ldots, \beta + 32\}$, $\beta = \frac{0.7 \sum_{t \in T^2} p_t}{m}$ and $l_{\max} = \arg\max_{t \in T^2}\{p_t\}$. Each gate set G_i contains four gates and $\alpha = 1$. Table 7 shows the CPU times for medium-sized and large instances. Small and medium-size instances can be solved to guaranteed optimality within one hour of CPU time, but for large instances OPL CPLEX regularly encounters memory problems.

Table 7 CPU times for the formulation of Sect. 7.1 with OPL CPLEX. An asterisk '*' means that OPL CPLEX was not able to solve the instance to guaranteed optimality within one hour; a double asterisk '**' means that CPLEX OPL was not able to solve the instance because of memory problems

n	m	τ	obj val	time (s)	n	m	τ	obj val	time (s)	n	m	τ	obj val	time (s)
16	4	1	569	3.36	18	4	1	931	3.64	20	4	1	1025	3.36
16	4	2	548	3.12	18	4	2	919	3.38	20	4	2	1010	3.12
32	8	1	1907	8.83	36	8	1	1672	17.93	40	8	1	2252	21.05
32	8	2	1793	2.91	36	8	2	1581	3.66	40	8	2	2160	4.14
48	12	1	2041	274.46	54	12	1	3355	73.57	60	12	1	3437	839.17
48	12	2	1795	9.61	54	12	2	2700	56.66	60	12	2	2671	73.81
64	16	1	4350	12.74	72	16	2	4307	186.17	80	16	2	*	>1 h
64	16	2	2810	101.27	72	16	3	4148	263.64	80	16	3	3895	51.47
80	20	2	4008	1101.42	90	20	2	*	>1 h	100	20	2	4390	1441.87
80	20	3	3746	224.88	90	20	3	4207	1550.64	100	20	3	4011	1624.45
96	24	2	6055	223.28	108	24	2	6072	277.65	120	24	2	*	>1 h
96	24	3	5056	539.54	108	24	3	5465	129.72	120	24	3	*	>1 h
112	28	2	5931	117.21	126	28	2	8634	58.49	140	28	3	**	**
112	28	3	4608	1524.75	126	28	3	**	**	140	28	4	**	**
128	32	2	9180	253.45	144	32	3	**	**	160	32	3	**	**
128	32	3	**	**	144	32	4	**	**	160	32	4	**	**

7.2 Dedicated gates

When it is not allowed to load or unload a trailer at a gate that is not included in the corresponding gate set, it is known beforehand at which gate-set each trailer will be loaded or unloaded. As this is no longer a scheduling decision, we say that the gates are 'dedicated' instead of 'preferred'.

The decision variables can now be chosen as follows: for all operations $t \in T$ and for all time periods $u \in H_t$,

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

where the time windows are the same as in the previous subsection. A linear formulation is then:

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

subject to

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

Table 8 CPU times for the formulation of Sect. 7.2 with OPL CPLEX. An asterisk '*' means that OPL CPLEX was not able to solve the instance to guaranteed optimality within one hour

n	m	τ	obj val	time (s)	n	m	τ	obj val	time (s)	n	m	τ	obj val	time (s)
16	4	1	569	7.82	18	4	1	931	4.27	20	4	1	1025	3.92
16	4	2	548	6.95	18	4	2	919	4.17	20	4	2	1010	4.20
32	8	1	1908	6.53	36	8	1	1682	8.86	40	8	1	2275	51.02
32	8	2	1798	9.73	36	8	2	1610	5.45	40	8	2	2172	7.30
48	12	1	2051	12.59	54	12	1	3386	78.76	60	12	1	3467	68.2
48	12	2	1807	6.31	54	12	2	2700	9.67	60	12	2	2685	28.16
64	16	1	4351	11.21	72	16	2	4399	59.86	80	16	2	4158	352.33
64	16	2	2832	50.74	72	16	3	4243	10.63	80	16	3	3966	26.08
80	20	2	4016	40.58	90	20	2	4866	468.21	100	20	2	4431	22.19
80	20	3	3759	19.30	90	20	3	4307	13.85	100	20	3	4110	54.90
96	24	2	6139	30.77	108	24	2	6126	61.95	120	24	2	*	>1 h
96	24	3	5139	28.66	108	24	3	5528	34.59	120	24	3	7391	437.51
112	28	2	6048	22.99	126	28	2	8688	16.97	140	28	3	7556	1654.69
112	28	3	4662	53.12	126	28	3	6029	429.86	140	28	4	7114	551.90
128	32	2	9193	9.67	144	32	3	7456	97.94	160	32	3	8563	82.15
128	32	3	6645	33.84	144	32	4	6656	213.56	160	32	4	7492	219.49

$$\sum_{(t_1,t_2,t_3)\in J:\pi_t=i} \left(x_{t_1u} + x_{t_3u} + \sum_{v\le u} \left(x_{t_2v} - x_{t_3v} \right) \right) \le |G_i| \qquad \forall G_i \subset G; \forall u \in \{1,\ldots,H_{\max}\} \tag{34}$$

$$\sum_{(t_1,t_2,t_3)\in (J_L\cup J_U)} \left(x_{t_1u} + x_{t_3u} \right) \le \tau \qquad \forall u \in \{1,\ldots,H_{\max}\}$$

$$x_{t_1u} - x_{t_2,u+1} = 0 \qquad \forall (t_1,t_2,t_3) \in J; \forall u \in H_{t_1}$$

$$\sum_{v=1}^{u} x_{t_3v} - \sum_{v=1}^{u-p_{t_2}} x_{t_2v} \le 0 \qquad \forall (t_1,t_2,t_3) \in J; u \in H_{t_3}$$

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

The decision variables and the objective function are the same as for the general dock assignment problem with trailer transportation, while the constraints are very similar to those in the model with preferred gates; the capacity constraint of the gates (constraint (34)) is modified as it is known beforehand which trailers will be loaded or unloaded at each subset.

Table 8 shows that the CPU times for the problem formulation with the dedicated gates are significantly smaller than for the preferred gates, because stage-two operations do not require an assignment decision to subsets of gates anymore. Even for large instances, OPL CPLEX can solve the problem to guaranteed optimality within one hour of CPU time. The objective values are never lower than for the case with preferred gates, as a feasible solution

for the dedicated gates is always a feasible solution for the preferred gates but the latter setting offers more flexibility to the decision maker.

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 gates for loading or unloading. The 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 the best time-indexed formulation, we are able to solve small and medium-sized instances to guaranteed optimality within reasonable CPU times. Moreover, two practical extensions are also examined, and the computational performance for these models is even better than for the generic problem. Further research is needed to produce solutions for large instances for the general problem 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

Artigues, C., Demassey, S., & Néron, E. (2008). *Control systems, robotics and manufacturing series. Resource-constrained project scheduling*. London: ISTE Ltd.

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

Berghman, L., Leus, R., & Spieksma, F. C. R. (2010). *Optimal solutions for a dock assignment problem with trailer transportation* (Working Paper KBI-1010). Department of Decision Sciences and Information Management (KBI), Faculty of Business and Economics, KULeuven, Leuven (Belgium).

Bierwirth, C., & Meisel, F. (2010). A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, *202*, 615–627.

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

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

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

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

Böse, J., Reiners, T., Steenken, D., & Voss, S. (2000). Vehicle dispatching at seaport container terminals using evolutionary algorithms. In *Proceedings of the 33rd Hawaii international conference on system sciences* (pp. 1–10).

Boysen, N., Fliedner, M., & Scholl, A. (2010). Scheduling inbound and outbound trucks at cross docking terminals. *OR-Spektrum*, *32*, 135–161.

Chen, J. S., Pan, J. C. H., & Wu, C. K. (2007). Minimizing makespan in reentrant flow-shops using hybrid tabu search. *The International Journal of Advanced Manufacturing Technology*, *34*, 353–361.

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

Cheong, C. Y., Tan, K. C., Liu, D. K., & Lin, C. J. (2010). Multi-objective and prioritized berth allocation in container ports. *Annals of Operations Research*, *180*, 63–103.

Choi, S. W., & Kim, Y. D. (2008). Minimizing makespan on an m-machine re-entrant flowshop. *Computers & Operations Research*, *35*, 1684–1696.

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

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

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

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

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

Graves, S. C., Meal, H. C., Stefeka, D., & Zeghmi, A. H. (1983). Scheduling of re-entrant flow shops. *Journal of Operations Management*, *3*(4), 197–207.

Guan, Y., Yang, K. H., & Zhou, Z. (2010). The crane scheduling problem: models and solution approaches. *Annals of Operations Research*. doi:10.1007/s10479-010-0765-3.

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

Gupta, J. N. D., Hairiri, A. M. A., & Potts, C. N. (1997). Scheduling a two-stage hybrid flow shop with parallel machines at the first stage. *Annals of Operations Research*, *69*, 171–191.

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

ILOG (2008). ILOG. CPLEX, 11, 0 user's manual. ILOG, Inc., available online at http://www.decf.berkeley.edu/help/apps/ampl/cplex-doc/.

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

Kedad-Sidhoum, S., Solis, Y. R., & Sourd, F. (2008). Lower bounds for the earliness-tardiness scheduling problem on parallel machines with distinct due dates. *European Journal of Operational Research*, *189*, 1305–1316.

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

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

Lenstra, J. K., Rinnooy Kan, A. H. G., & Brucker, P. (1977). Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, *1*, 343–362.

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

Mellouli, R., Sadfi, C., Chu, C., & Kacem, I. (2009). Identical parallel-machine scheduling under availability constraints to minimize the sum of completion times. *European Journal of Operational Research*, *179*, 1150–1165.

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

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

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

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

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

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

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

Stahlbock, R., & Voß, S. (2008). Operations research at container terminals: a literature update. *OR-Spektrum*, *30*, 1–52.

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

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

van den Akker, J. M. (1994). *LP-based solution methods for single-machine scheduling problems*. PhD thesis, Technische Universiteit Eindhoven, the Netherlands.

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

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