Flexible Services and Manufacturing Journal (2021) 33:281–311
https://doi.org/10.1007/s10696-019-09366-3

Coordinated optimization of equipment operations in a container terminal

T. Jonker[1,2] · M. B. Duinkerken[1] · N. Yorke-Smith[3,4] · A. de Waal[2] · R. R. Negenborn[1]

Published online: 23 July 2019

Abstract

Increasing international maritime transport drives the need for efficient container terminals. The speed at which containers can be processed through a terminal is an important performance indicator. In particular, the productivity of the quay cranes (QCs) determines the performance of a container terminal; hence QC scheduling has received considerable attention. This article develops a comprehensive model to represent the waterside operations of a container terminal. Waterside operations comprise single and twinlift handling of containers by QCs, automated guided vehicles and yard cranes. In common practice, an uncoordinated scheduling heuristic is used to dispatch the equipment operating on a terminal. Here, uncoordinated means that the different machines that operate in the container terminal seek optimal productivity solely considering their own respective stage. By contrast, our model provides a coordinated schedule in which operations of all terminal equipment can be considered at once to achieve productivity closer to the QC optimal. The model takes the form of a hybrid flow shop (HFS) with novel features for bi-directional flows and job pairing. The former enables jobs to move freely through the HFS in both directions; the latter constrains certain jobs to be performed simultaneously by a single machine. We solve the coordinated model by means of a tailored simulated annealing (SA) algorithm that balances solution quality and computational time. We empirically study time-bounded variants of SA and compare them with a branch-and-bound algorithm. We show that our approach can produce coordinated schedules for a terminal with up to eight QCs in near real time.

Keywords Scheduling · Simulated annealing · Port operations · Hybrid flow shop

✉ T. Jonker
timjonker@live.nl

Extended author information available on the last page of the article

1 Introduction

Worldwide today, 60% of the total cargo volume is transported in standardized boxes called twenty-feet equivalent units (TEU) containers (Stahlbock and Voß 2008). As the total transported volume increases, so does the complexity of the port logistics (Gracia et al. 2017). The function of a port container terminal is to trans-ship such TEU containers: deep-sea vessels, barges, trains and trucks arrive to the terminal to deliver or collect containers. The terminal operations must ensure that the correct containers are loaded and unloaded to and from their respective carriers. The speed at which this trans-shipment is achieved is an important benchmark for container terminals. In particular, the turnaround time of deep-sea vessels is used to determine the performance of a terminal (Bish 2003; De Koster et al. 2009). A terminal with lower turnaround time is more attractive for shipping lines to move their goods through, and the shorter the turnaround time of deep-sea vessels, the higher the terminal capacity.

One of the most important performance indicators in terms of container turnaround times is the productivity of terminal's quay cranes (QCs) (Dik and Kozan 2017). By coordinating the operations of the equipment operating on the terminal, including the QCs, the equipment can be used more efficiently than the current uncoordinated heuristic approaches. However, the optimization-based, non-heuristic scheduling models currently available do not match with the operations on a container terminal, because twin-lift operations are ignored and simultaneous loading and unloading of multiple ships is not considered. Further, the available optimization models cannot be used in near real-time—whereas computational times should remain low since re-planning is often required at a container terminal, since the current situation can change rapidly.

Our main contribution is the development of (1) a hybrid flow shop (HFS) model that contains novel features for bi-directional flows and job pairing; and (2) a tailored simulated annealing (SA) solution technique that efficiently approximates exact solutions to the developed mathematical model, allowing for near real-time usage. Until now, bi-directional flows were only addressed sequentially or by dedicating a group of machines to one of the two directions. Within our model different machines all work in both directions and each machine is free to work in any direction independent of other machines.

The article is structured as follows. Section 2 introduces the function of a container terminal and identifies the scheduling problems. Section 3 surveys related work and gives background information for our approach. Section 4 presents our developed mathematical model. Section 5 introduces the SA algorithm that is used to find an approximate solution to the mathematical model. Section 6 presents our computational results. Section 7 concludes and discusses future work.

2 Background and problem formulation

In this section we briefly introduce the layout of a container terminal and the scheduling problems related to its waterside operations. A container terminal is primarily a material handling system; it is a link in the intercontinental transportation chain.

Within this chain, the container terminal fulfills a buffer function between sea and land transportation. Further, the container terminal provides secondary services like inspection, washing, repairs and cargo consolidation (Gurning 2000). This article focuses on the waterside operations on a container terminal: transporting containers between the yard and the deep sea vessels. Three different types of equipment are used to perform waterside operations: yard cranes (YCs), automated guided vehicles (AGVs) and quay cranes (QCs) (Dragović et al. 2017).

A schematic drawing of a typical container terminal is given in Fig. 1. From this figure the tasks of the different equipment types become clear: QCs move containers between the ship and the quay, the AGVs move containers between the quay and the transfer point and YCs move containers within the stack.

This article investigates how a coordinated schedule could lead to increased productivity of Quay Cranes on a container terminal. A coordinated schedule applied to container terminals is a schedule that considers simultaneously the operations of all equipment operating on the container terminal to achieve overall optimal performance. This is in contrast to an uncoordinated schedule that considers operations of all equipment separately and tries to find optimal solutions for all equipment individually. Since different equipment operating on a container terminal could have conflicting objectives, an uncoordinated planning does not lead to the optimal holistic solution. Therefore, a richer coordinated scheduling model than currently exists is desired to find the optimal holistic solution.

The coordinated schedule primarily focuses on decreasing the makespan of QC operations, since these are most critical operations (Bish 2003). Therefore, some research omits the coordinated aspect and specifically focuses on QC scheduling (Dik and Kozan 2017). However, another objective of the coordinated schedule is

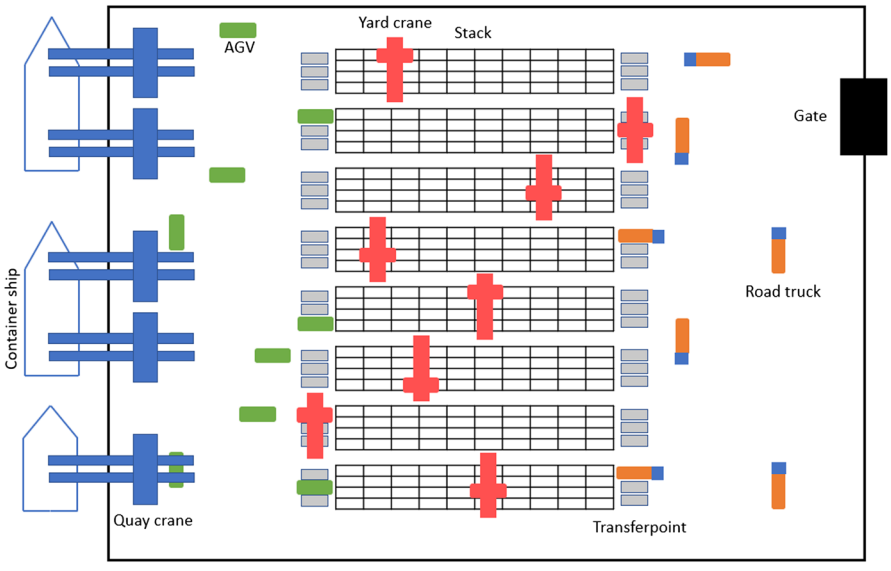

Fig. 1 Schematic plan of a generic container terminal

to reduce the makespan of YC operations. Finally, an evenly distributed workload is also desired. In general, schedules can be subdivided into three related categories: scheduling, dispatching and routing. *Scheduling* signifies the process of assigning operating times to the individual containers. Subsequently, *dispatching* is a process during which equipment is assigned to handle individual containers. Thereafter, *routing* describes the process of determining the routes of equipment required to handle the containers. In coordinated schedules we recognise the connection between the different categories and adapt the schedule accordingly.

Specifically, Fig. 2 identifies the set of scheduling problems on a container terminal. From left to right, the first two scheduling problems occur upon arrival of a deep-sea vessel: berth allocation and bay scheduling (Al-Dhaheri and Diabat 2015; Bierwirth and Meisel 2010; Tavakkoli-Moghaddam et al. 2009; Kim and Park 2004; Lin et al. 2018; Liu et al. 2017; Jin et al. 2015; Iris et al. 2015). *Berth allocation* is used to determine which ship can berth at which quay in ports having multiple berth sites available. *Bay scheduling* concerns the problem of allocating QCs to specific bays. Therefore this problem is sometimes also called a *QC assignment* problem. A bay is one of the three coordinates used to describe the position of a container within the coordinate system introduced in Fig. 3.

The next problem is determining the yard location of containers (Steenken et al. 2004; Cao and Uebe 1995; Kim et al. 2000; Zhang et al. 2003; Petering et al. 2017; Jin et al. 2014, 2015). The solutions to three scheduling problems define the starting point of the individual container scheduling problem considered within this research. Individual container schedules and yard allocation are closely related. Therefore, the quality of the yard allocation solution partially determines the performance of the individual container handling schedule; there is some work on integrating yard allocation with individual container scheduling (Kim and Bae 1998).

Our focus is the scheduling and dispatching problems: assigning equipment to handle operations on containers, and determining the starting times of these operations. Scheduling and dispatching are strongly intertwined; operations cannot be performed simultaneously on the same machine (e.g., QC) and a machine requires a sequence-dependent setup time to start a new operation.

The last scheduling problem present on container terminals is routing (Steenken et al. 2004), which consists of determining the route of Lift Automated Guided Vehicles (Lift-AGVs) on the container terminal. Typically, routing problems are

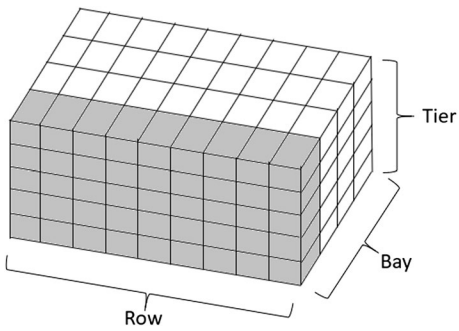

Fig. 2 Relation of the research problem with other logistics problems encountered on container terminals

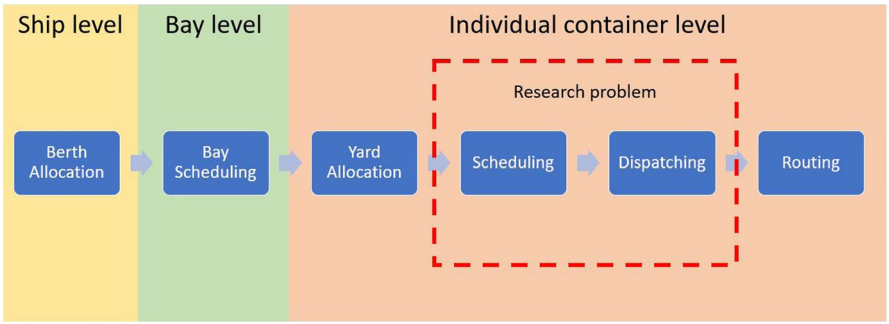

Fig. 3 Coordinate system to describe the position of a container within a block, a single bay is highlighted

solved after scheduling and dispatching problems are solved. Since we consider both scheduling and dispatching, routing is the first encountered problem after the individual container schedule is constructed. Routing is frequently studied and the interested reader is referred to Reveliotis (2000), Vuuren (2017) and Qiu and Hsu (2000).

3 Related work

Some research is restricted to scheduling container operations only at one of the three introduced stages (Dik and Kozan 2017; Huang and Li 2017; Yu et al. 2017; Lim et al. 2015; Jin et al. 2014). In some cases side-effects such as environmental impact (Yu et al. 2017) or disturbance recovery (Lim et al. 2015) are considered. Our research addresses the important and realistic setting of integrated scheduling. The most common approach to modelling mathematically the operations of a container terminal as a whole is to view the waterside operations as a Hybrid Flow Shop (HFS). In this section we review previous work and point out its limitations with regards to integrated container terminal scheduling. For a survey of container terminal operations, we refer to Stahlbock and Voß (2008).

In HFS problems, n different jobs are processed in a series of m stages while optimizing an objective function (Ruiz and Vázquez-Rodríguez 2010; Ribas et al. 2010). A job consists of multiple operations which should be performed on the different stages. Specifically, in a HFS (Ruiz and Vázquez-Rodríguez 2010; Chen et al. 2007):

1. The number of processing stages m is at least 2.
2. Each stage k has $M^k \geq 1$ machines in parallel and at least one stage has $M^k > 1$.
3. All jobs follow the same production flow through the stages. However, a job is allowed to skip one or more stages.
4. Each job j requires a processing time p_{jk} on stage k.

Within the standard form of the HFS, each job is available from the start and can be processed on any of the different parallel machines within a stage. Further, each

machine is identical and has the same deterministic processing time which is known in advance, and each machine can only serve one job simultaneously. Additionally, sequence dependent setup times are negligible, buffers between stages are unlimited and preemption of operations is not allowed. Clearly, the standard form of the HFS problem is not directly applicable to container terminal operations; various extensions are used. We note that even the standard HFS is NP-hard (Gupta 1988).

At a container terminal, a job signifies a transfer between a vessel and the yard or between the yard and a vessel. To complete the job a container requires processing on different stages. All machines of equal type (YCs, AGVs and QCs) compose a stage. Therefore, as we will elaborate in the next section, the waterside of a container terminal can be represented by a three-stage HFS problem that operates in both directions.

Modelling a container terminal as a variant HFS problem is a common approach. The literature studies either additional features added to the standard HFS (Xin 2015; Xin et al. 2014, 2015a, b; He et al. 2015; Chen et al. 2013), or the solution technique (Chen et al. 2007, 2013; Lau and Zhao 2008; Bish 2003; Lu and Le 2014; Kim and Bae 2004; He et al. 2015; Kaveshgar and Huynh 2015; Cao et al. 2010a, b; Kizilay et al. 2018). To compare different HFS model variants, a classification scheme has been introduced (Vignier et al. 1999; Ruiz and Vázquez-Rodríguez 2010; Ribas et al. 2010). The classification consists of three categories: α, β, γ. These categories respectively identify the structure of the shop, the additional features and the objective function. We use this classification to classify previous efforts to construct integrated container terminal schedules applying HFS models, as shown in Table 1. The table also indicates the problem size and the applied solution technique. Since not all studies concern AGVs a more generic term is used in Table 1: horizontal transport vehicles (HTVs). HTVs includes all container transport vehicles that could transport one or more containers without the ability to lift containers.

Previous work has the shortcoming that the models have important differences from real container terminals. Namely, loading and discharging is separated (apart from He et al. 2015) and there is no possibility for twinlift operations. Further, blocking and precedence are not always included. Second, apart from He et al. (2015), Chen et al. (2007, 2013), the trend in the literature is to study only small terminals. Third, the most realistic models in the literature (He et al. 2015; Chen et al. 2007, 2013) require substantial computational time, making them unsuitable for real-time use. The model developed in this article is more complete as it possesses the identified shortcoming features of container terminal models. Further, it is realistic with regards to real container terminal sizes when solved with the metaheuristic approach we propose.

We discuss in detail notable prior work from Table 1. First, Lu and Le (2014) formulated a stochastic HFS problem, in order to model uncertainties within normal operations. Although stochastic, the model is otherwise less extensive compared to others models, and notably with our model.

Chen et al. (2013) integrate the individual container scheduling and dispatching problem with the routing problem. This is achieved by adding multiple layers. Combining our expressive model for scheduling and dispatching with the routing problem is one extension for future work.

Table 1 Classification scheme of other papers considering integrated container terminal scheduling

Article	α	β	γ	Cont	QC	HTV	YC	Solution technique
Bish (2003)	$FH2, ((PM)^k)_{k=1}^2$	$M_j, S_{sd}, block$	$\bar{C}^w$	2500	4	40	–	Transshipment based heuristic
Kim and Bae (2004)	$FH2, ((PM)^1, (RM)^k)_{k=2}^3$	S_{sd}	$\bar{L} + \bar{C}$	100	2	7	5	Dispatching method
Chen et al. (2007, 2013)	$FH3, ((RM)^k)_{k=1}^3$	$M_j, S_{sd}, block$	C_{max}	500	5	30	20	Tabu search, constraint programming
Lau and Zhao (2008)	$FH3, ((RM)^k)_{k=1}^3$	$prec, S_{sd}, M_j$	$\bar{C}^w$	16	2	4	2	Genetic algorithm
Cao et al. (2010b)	$FH2, ((PM)^k)_{k=2}^3$	$S_{sd}, M_j, block$	C_{max}	500	–	60	40	MIP + Benders cut
Cao et al. (2010a)	$FH2, ((PM)^k)_{k=2}^3$	$S_{sd}, M_j, block$	C_{max}	15	1	4	–	Genetic algorithm + Heuristic
Lu and Le (2014)	$FH2, ((PM)^k)_{k=1}^2$	$p_{ij} \sim N(\mu, \sigma), S_{sd}, block$	T_{max}	30	–	4	1	Particle swarm optimization
Xin (2015), Xin et al. (2015a)	$FH3, ((RM)^k)_{k=1}^3$	$block$	C_{max}	40	5	10	8	Variable neighbourhood search
He et al. (2015)	$FH3, ((RM)^k)_{k=1,3}, (PM)^2$	$prec, S_{sd}$	$\bar{L}^w + energy$	500	10	50	20	Genetic algorithm
Kaveshgar and Huynh (2015)	$FH2, ((RM)^k)_{k=1}^2$	$block, prec$	C_{max}	1050	2	15	–	Genetic algorithm
Kizilay et al. (2018)	$FH2, ((PM)^k)_{k=1,3}$	$prec, S_{sd}, M_j$	C_{max}	550	4	∞	6	Constraint programming

Extended with the maximum managed problem size [# of containers, QCs, horizontal transport vehicles (HTVs) and YCs] and the applied solution technique

Xin et al. (2014) and He et al. (2015) integrate energy consumption into their models. Xin et al. (2014) used a multi-stage approach. By contrast, He et al. (2015) directly integrate energy consumption in the HFS problem. This is done by reducing travel distances of the different equipment in addition to optimizing QC operations. Xin et al. extended their work with elements of routing, by adding a layer to avoid collisions of AGVs (2015a). In the most extended version (Xin 2015) of this work, 5 QCs, 10 AGVs and 8 YCs are considered. However the focus is the integration of discrete-event dynamics (the schedule) and continuous-time dynamics (energy consumption and collision avoidance), rather than developing the schedule itself, as in our work.

Most recently, Kizilay et al. (2018) consider an infinite number of HTVs. The container terminal these authors consider has its performance bottleneck is in the QCs and the YCs. Therefore, by eliminating the scheduling process at the HTV stage the computational time can be reduced. However this is not realistic for terminals in general. Another novelty of Kizilay et al. (2018) is that bay scheduling is included, which again can be an extension of the model we propose.

Turning to solving the models proposed: if the HFS is small enough, branch-and-bound (i.e., mixed integer programming) can be used to arrive at an exact solution. In practice, it is common to use (meta)heuristics to approximate the exact solution of a HFS within reasonable time. Popular metaheuristics are genetic algorithm (GA), Tabu Search and simulated annealing (SA). Jungwattanakit et al. (2009) report that SA is superior to GA and Tabu Search for large scale HFS problems. There is successful precedent in using SA for problems in terminal waterside operations (Lin et al. 2018).

4 Model formulation

In this section we present the mathematical model of coordinated container terminal operations as an extended Hybrid Flow Shop (HFS). Figure 4 shows a schematic overview of the generic container terminal considered in this research. Each container can only be handled by one specific YC, but the sequence of operations is variable. Each YC has access to its own transfer point modelled as a buffer. This means that the buffer stage contains multiple spots to place containers. The buffer is not shared between different YCs. From an operational point of view, a system without a buffer is more sensitive than one with a buffer, since direct interchange of containers is required. However, from a modelling perspective, a system with a buffer is more challenging as the number of relations increases. Further, it is likely that in the future most automated ports will operate with a buffer zone since it is shown be the most cost-effective alternative (Saanen 2016; Yang et al. 2004; Schroër et al. 2014). Therefore, a system with a buffer is the most applicable scenario and that is why it is considered in this article.

At the QC stage, not only the specific machine but also the sequence of operations is known in advance. This is because the loading plan is determined by the shipper and not by the port, since the shipper is responsible for the stability of the ship.

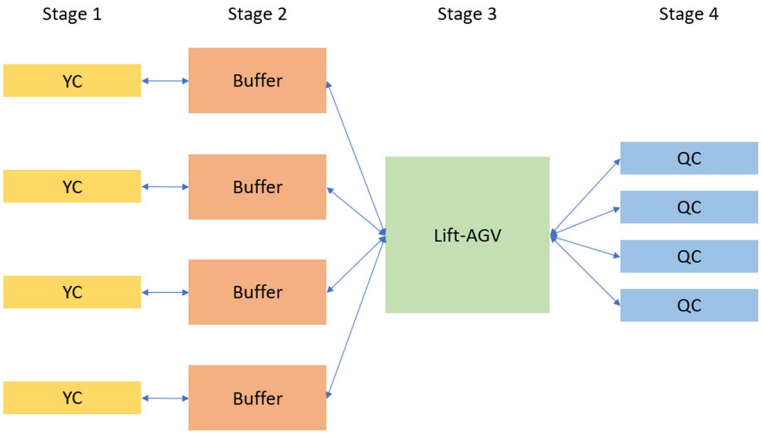

Fig. 4 Illustration of the HFS applied to a container terminal. The YC and QC stages are represented by multiple small blocks since the machine assignment is fixed. The Lift-AGV stage is represented by a large block since the machine selection is an element of the to be developed schedule. Arrows are used to indicate how the different blocks are connected. Containers can only move along the arrows

4.1 Model description

First we describe the coordinated container scheduling problem; below we then give its mathematical formalisation.

- Four stages with unrelated parallel machines at each stage model the YC, buffer, Lift-AGV and QC respectively. The buffer is inserted as an extra stage where the number of machines equals the buffer capacity. Within this research a buffer capacity of five at each YC is studied: this matches the value used in practice by a port operations company.
- Precedence relations exist at the QC stage to ensure that containers can only be (un)loaded in a specific sequence.
- Sequence-dependent setup times describe the *back route* of equipment: a route that equipment should travel to pickup their next container after delivering a container.
- Machine eligibility is applied at the YC and QC stage to ensure that certain containers can only be handled by specific machines.
- Blocking constrains the available buffer space between different stages. Blocking signifies that equipment can only proceed with its next operation once the next equipment type becomes available. This means that certain containers block a machine for as long as the next machine is not available. In our case blocking applies on all stages.
- Bi-directional flow shop enables containers to move in both directions through the shop. This is required since both loading and unloading containers are simultaneously present on the container terminal.

- Release dates signify that machines and containers become available not directly from the start. This enables the mathematical model to be constructed at an arbitrary time during operation of the container terminal.
- Job pairs enable machines to simultaneously handle two jobs. Each job pair finishes at the same time but the two jobs of the pair can be started at different times. Job pairs are used to enable the HFS model to handle twin lift moves. Twin lift moves are moves consisting of two containers that require simultaneous handling at the QC stage and the Lift-AGV stage. Job pairs could be formed differently on different stages and could have a different processing time dependent on their pick-up sequence. Note that, since twinlift moves are handled individually at the YC, the pair of moves must be considered separately.
- Preemption of operations is not allowed.
- Lastly, the objective of the HFS is to minimize the makespan of the entire container plan.

Our HFS problem can be classified with the $\alpha|\beta|\gamma$ classification as:

$$FH4, ((RM)^k)_{k=1}^4 \mid r_j, M_j, S_{sd}, block, prec, Bidirec, Jobpairs \mid \bar{C}$$

The features *Bi-direc* and *Job-pairs* are newly introduced in this research. Bi-direc is used to describe the relaxation of unidirectional flow in the flow shop. Job-pairs is a feature introduced to specify that some containers require simultaneous handling on a stage by one machine. Table 2 summarizes how our model is the most expressive HFS model for scheduling and dispatching container terminal waterside operations.

4.2 Mathematical model

In this section we give the formal model of the problem in terms of its parameters, decision variables, constraints and objective.

4.2.1 Parameters

i, k	Job index $i, k \in \phi_1 \cap \phi_2$
j	Stage index $j \in 1, 2, 3, 4$
j'	The next stage for a container. $j' = j + 1$ during loading and $j' = j - 1$ during unloading.
m	Machine index $m \in 1, 2, \ldots n_j$
n_j	Number of machines on stage j
N	Total number of containers
ϕ	Set of all jobs excluding $i = 0, i = N + 1$
ϕ_1	Set of all jobs including the first dummy job $i = 0$
ϕ_2	Set of all jobs including the last dummy job $i = N + 1$
P	Set of job pairs with precedence relations
O_{ij}	Operation i at stage j

Table 2 Comparison of the model developed with the literature, indicating the structure, features and objective value according to the classification scheme of Vignier et al. (1999), Ruiz and Vázquez-Rodríguez (2010), Ribas et al. (2010)

Article	Stages	Machine type	M_j	S_{sd}	Block	Prec	$p_{ij} \sim N(\mu, \sigma)$	Bidirec.	Jobpairs	Objective
Bish (2003)	3	P	×	×	×					$\bar{C}^w$
Kim and Bae (2004)	2	P		×						$\bar{L} + \bar{C}$
Chen et al. (2007, 2013)	3	R	×	×	×					C_{max}
Lau and Zhao (2008)	3	R	×	×		×				$\bar{C}^w$
Cao et al. (2010a, b)	2	P	×	×	×					C_{max}
Lu and Le (2014)	2	P		×	×		×			T_{max}
Xin (2015), Xin et al. (2015a)	3	R			×					C_{max}
He et al. (2015)	3	P, R		×		×				$\bar{L}^w + energy$
Kaveshgar and Huynh (2015)	2	R			×	×				C_{max}
Kizilay et al. (2018)	2	R	×	×		×				C_{max}
This research	4	R	×	×	×	×		×	×	$\bar{C}^w$

M_{ij}	Set of machines which can process O_{ij}
E_m	Set of jobs that can be performed on machine m, includes jobs $i = 0$ and $i = N + 1$
K_{ij}	Set of jobs that can form a pair with job i on stage j, includes jobs $k = 0$ and $k = N + 1$
T	Set of twinlift operations
T_i	The twin pair of container i
A	Set of containers that is already assigned to equipment
Pr	Set of containers that is already assigned to equipment but not in current progress
J_i	Set of stages that a container i already past or is being currently processes job i
Q_{jm}	Number of jobs that must be performed on machine m
r_{ij}	Release date of job i on stage j
p_{ij}	Processing time of operation O_{ij}
p_{ij}^{ik}	Processing time of operation O_{ij} whenever operation i directly precedes operation k
p_{ij}^{ki}	Processing time of operation O_{ij} whenever operation k directly precedes operation i
s_{ikj}	Setup time between job i and job k at stage j
s_{0kjm}	Setup time between the dummy job and job k on stage j for machine m.
H	A sufficiently large constant
β	A constant indicating the weight of completion time of operations on the YC stage, $\beta \ll 1$

The stage indices 1 to 4 respectively represent: the YC stage, the rack stage, the Lift-AGV stage and the QC stage.

4.2.2 Decision variables

- t_{ij} The start time of job i on stage j
- z_{ikj} 1, if O_{ij} immediately precedes O_{kj} on stage j; 0, otherwise $\forall i \in \phi, \forall k \in \phi_2, \forall j \in \{1, 2, 3, 4\}$.
- z_{0kjm} 1, if O_{0jm} immediately precedes O_{kj} on stage j for machine m; 0, otherwise $\forall k \in \phi_2, \forall j \in \{1, 2, 3, 4\}, \forall m \in M_{ij}$.

The difference between z_{ikj} and z_{0kjm} is that z_{0kjm} concerns the dummy job for which it is required to know at which machine this job is scheduled. Together with z_{ikj} and the machine selected for the dummy job it is possible to backtrack the respective machine for each container.

The parameters p_{ij}^{ik} and p_{ij}^{ki} are only required for twin moves ($\forall i, k \in T$) since twin moves are represented by two sequential single moves with different processing times. Whenever i, k are pairs of a twin move p_{ij} is replaced by $p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij}$.

4.2.3 Constraints and objective

$$\text{Minimize} \quad Z = \beta \sum_{i=1}^{N} t_{i1} + \sum_{i=1}^{N} t_{i4} - \beta \sum_{j=1}^{4} \sum_{m=1}^{n_j} \sum_{k=1}^{N+1} z_{0kjm} \tag{1}$$

$$\text{Subject to} \quad t_{ij} \geq r_{ij} \qquad \forall i \in \phi, \forall j \in \{1, 4\} \tag{2}$$

$$H(1 - z_{0kjm}) + t_{kj} \geq s_{0kjm} \qquad \forall k \in \phi \cap A^C, j \in \{3\}, \forall m \in n_j \tag{3}$$

$$t_{ij} + p_{ij} \leq t_{ij'} \qquad \forall i \in \phi \cap T^C, \forall j \in \{3,4\} \tag{4}$$

$$t_{ij} + p_{ij} \leq t_{ij'} \qquad \forall i \in \phi, \forall j \in \{1,2\} \tag{5}$$

$$t_{ij} + p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij} \leq t_{ij'} \qquad \forall i \in T, \forall j \in \{3,4\} \tag{6}$$

$$\sum_{i=1}^{N} \sum_{k=1}^{N} z_{ikj} = Q_{jm} + 1 \qquad \forall i,k \in E_m, \forall j \in \{1,2,3,4\}, \forall m \in M_{ij} \tag{7}$$

$$\sum_{k=1}^{N} z_{0kj} = n_j \qquad \forall k \in E_m, \forall j \in \{1,2,3,4\} \tag{8}$$

$$\sum_{i=1}^{N} z_{i(N+1)j} = n_j \qquad \forall i \in E_m, \forall j \in \{1,2,3,4\} \tag{9}$$

$$\sum_{k=1}^{N+1} z_{ikj} = 1 \qquad \forall i \in \phi, \forall k \in \phi_2 \cap K_{ij}, \forall j \in \{1,2,3,4\} \tag{10}$$

$$\sum_{i=0}^{N} z_{ikj} = 1 \qquad \forall i \in \phi_1 \cap K_{kj}, \forall k \in \phi, \forall j \in \{1,2,3,4\} \tag{11}$$

$$z_{ikj} + z_{kij} = 1 \qquad \forall i,k \in T, j \in \{3,4\} \tag{12}$$

$$t_{ij'} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi \cap T^C, \forall j \in \{1,2,3,4\} \tag{13}$$

$$t_{ij} + p_{ij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi \cap T^C, \forall j \in \{4\} \tag{14}$$

$$t_{ij} + p_{ij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in \phi, \forall j \in \{1,2\} \tag{15}$$

$$t_{ij} + p_{ij}^{ik} z_{ikj} + p_{ij}^{ki} z_{kij} + s_{ikj} \leq t_{kj} + H(1 - z_{ikj}) \qquad \forall i,k \in T, \forall j \in \{3,4\} \tag{16}$$

$$\sum_{i=1}^{N} t_{ij} \leq 0 \qquad \forall i \in A \cap T^C, \forall j \in J_i \tag{17}$$

$$\sum_{i=1}^{N} t_{ij} \leq H(1 - z_{ik3}) \qquad \forall i, k \in A \cap T, \forall j \in J_i \tag{18}$$

$$z_{0kjm} + \sum_{i=1}^{N} z_{ikj} = 1 \qquad \forall i \in A \cup T_k, \forall k \in Pr, \forall j \in \{1, 2, 3\} \tag{19}$$

$$z_{ikj}, z_{ikjm} = 0 \; or \; 1 \qquad \forall i, k \in \phi_1 \cup \phi_2, \forall j \in \{1, 2, 3, 4\} \tag{20}$$

The objective function (Eq. 1) minimizes the makespan, weighted over the stages with the most weight to the completion time of operations at the QC stage. Since each operation is considered in the objective function, most weight is indirectly added to the first few operations on each QC. This is because delays in the beginning propagate to other operations. This is a desired feature of the objective function since the first few operations are most accurately predicted and are most important to be finished on time. Besides that, distribution of jobs to different machines is stimulated by the small negative factor β. Whenever the makespan is equal, the jobs should be evenly distributed over all machines. Additionally, β assures that operations start as early as possible without delaying the operations at later stages. β is set very small, e.g., 0.01, to prevent from conflicting with the main objective.

Constraint 2 enforces that jobs can only start after they are being released. This is only required at the starting stage and the first job of the sequence since other operations and jobs are forced to start later. Constraint 3 has a similar function as Constraint 2 but, focuses on the Lift-AGV stage. This is required since it is unknown which container is handled by which Lift-AGV and each Lift-AGV can become available independently.

Constraint 4 ensures that a job can only start at the next stage once the current stage is completed. Constraint 6 is required to make use of the sequence dependent processing times of twin moves. However, at the YC stage containers cannot be twin containers. Therefore, the processing time remains unchanged and thus Constraint 4 applies for all containers in ϕ. This is expressed by Constraint 5.

Constraint 7 describes machine eligibility at the YC stage. It ensures that a sequence of z_{ikj} is only selected from a subset of ϕ. Furthermore, the number of sequence variables is constraint to the number of jobs that should be handled on the machine plus one. This is because each sequence starts and ends with a dummy job.

Constraint 8 ensures that the number of started sequences is equal to the number of machines. However, once a sequence cannot start at an arbitrary job, that machine receives its personal constraint. The personal constraint enforces that a sequence should start at one of the available jobs. Constraint 9 is similar to Constraint 8 but,

now to ensure that each sequence is ended. Similarly to Constraint 8, if a machine cannot process every job; that machine is given its personal variant of Constraint 9.

Constraint 10 ensures that each operation is proceeded by another operation or is the end of the sequence. Similar is Constraint 11, which ensures that each operation is preceded by another operation or is the start of the sequence. Both of these constraints exclude infeasible job pairs whenever they are obliged to be performed on different machines.

Constraint 12 ensures that a pair of twin moves is handled by the same machine. This is not required at stage 1 or 2 since different YCs might work on the same twin pair.

Constraint 13 is the blocking constraint; it states that whenever job i immediately precedes job k on stage j, job k cannot start earlier than the start time of job i on the next stage plus the setup time between job i and job k. This is because the start of job i on the next stage marks the point where the container is released from the current stage.

Constraint 14 is used to separate operations by their processing times and setup times. Practically, this constraint is only required at the first and last stage since intermediate stages automatically satisfy this constraint by a combination of Constraints 4 and 13. Constraint 16 is the same as Constraint 14 only for twin moves which have their processing time dependent on the sequence of operations. For the same reason as for Constraint 5, Constraint 15 is added specifically for the first stage since the twin moves are separated here and thus processing times are sequence independent.

Constraint 17 ensures that containers that have already passed a stage or that are currently processed on a stage start their operation at time zero. However, this constraint is not valid whenever an unloading twin container is currently assigned to a QC or to a Lift-AGV. This is because twin containers are modelled as sequential single moves with different processing times. Therefore, Constraint 18 is introduced which specifies that at least one of the twin containers starts at zero. Constraint 12 subsequently ensures that the other container of the twin pair is processed directly after its twin pair.

Constraint 19 ensures that operations that are assigned to equipment but, that have not currently started can only proceed: containers that are in current progress or containers that have passed that stage already or their twin pair or the dummy start container.

Finally, Constraint 20 is used to enforce binary solutions to the decision variables z_{ikj} and z_{0kjm}.

We can reduce the number of decision variables, by observing that the sequence of operations on stage 4 is already known entirely. This means all associated z variables to this stage become superfluous as decision variables. Additionally, variable z_{iij} does not exist: a job can never immediately proceed or precede itself. Therefore, these variables are also excluded.

Hence, the total number of decision variables, which depends on the number of stages, the number of containers and the number of Lift-AGVs and the number of buffer locations, is:

$$\#\ of\ variables\ = N \cdot (\#\ of\ stages) + (N+1) \cdot (n_2 + n_3) + (N+1)^2 + 2(N^2 + N) \tag{21}$$

Equation 21 is based on the real-life practice at an industry-leading port operations software company where the containers can only be planned on one YC and one QC. The Lift-AGVs are allowed to handle any container except that they cannot preempt a container handling. Further, the sequence of operations at the QC is known in advance and thus requires no decision variables. The equation consists of four terms. The first term derives from the starting time variables of the various equipment items; the second term derives from the sequence variables required for the first container at the AGV and rack stage; the third term derives from the sequence variables at the YC stage; the final term derives from the remaining sequence variables at the AGV and rack stage.

5 Solution technique

In this section we describe how the model of Sect. 4 can be solved effectively to achieve a balance between the performance (in terms of the objective) and the computational time. Following Jungwattanakit et al. (2009), we adopt simulated annealing (SA) for our HFS problem. Figure 5 gives an overview of the approach.

Our approach is a tailored SA algorithm. We summarize it here and then describe each part in detail below. The algorithm uses local search and a temperature based accepting/rejection criterion to arrive at new solutions. The first step is to generate an initial solution. Two different approaches are used for this and the best solution is selected as initial solution. Subsequently, the initial solution is modified locally to investigate better alternatives. The local modification is performed by altering the sequence of operations on the different machines. To do this, two containers are selected to be either inserted or swapped. This creates a neighbouring sequence. The neighbouring sequence might require repair to remain feasible. Thereafter, the quality of the newly obtained neighbouring sequence can be evaluated with linear programming (LP). The quality of the

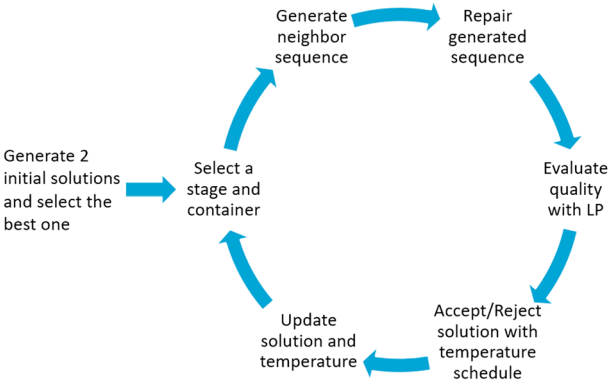

Fig. 5 Overview of the SA approach

neighbouring sequence and the temperature schedule are used to decide whether the new solution is accepted or rejected. The process repeats until a stopping criterion is satisfied, whereupon the incumbent solution is returned as the final solution. The notable features of our approach are: (1) we analyse the obtained solutions and use this analysis to select containers that are likely to reduce delays upon modifying sequences; (2) we use the hindsight of the problem during the repair process; and (3) we use LP over the continuous variables to quickly determine solution quality.

5.1 Initial solution

We leverage two different perspectives in order to generate two diverse initial solutions. One initial solution is created from a container perspective and the other from a machine perspective. Subsequently, the quality of both these solutions is evaluated and the best one is selected as initial solution to start the local search.

First, when scheduling from the container perspective, all containers are sorted based on their desired starting times at the first stage. Subsequently, the containers are scheduled in that order on all stages. This initial solution exploits the fact that the sequence of operations is known at the QC stage. Another name for this initial schedule is a *non-delay schedule* since machines are never kept idle while there is a container available (Pinedo 2016; Magalhães-Mendes 2011; Chen et al. 2007).

Second, when scheduling from the machine perspective, all machines and all containers are sorted based on their availability. Machine availability does also consider the setup time to a particular container. Subsequently, the machine which comes first available is paired with the container that becomes first available. This is different from the container perspective since containers are not scheduled through the entire shop immediately.

5.2 Selection

Local modifications consists of a swap or insert of two containers to update the sequence of operations. A prerequisite is to select two containers that will be used to modify the original sequence. Since the loadplan is fixed, the sequence of operations at the QC cannot be changed. Therefore, containers are selected from the YC stage and from the Lift-AGV stage. At the YC stage; containers are selected randomly. This is because the possible variation of the schedule is small since containers cannot switch machines.

At the Lift-AGV stage; containers are selected with lookahead. The aim is to predict which combination of two containers is most likely to improve the schedule the most. This means one container that is associated with delays and one container that precedes an idle period of the Lift-AGV. This selection process involves two steps. First, containers are sorted based on the delay that they cause

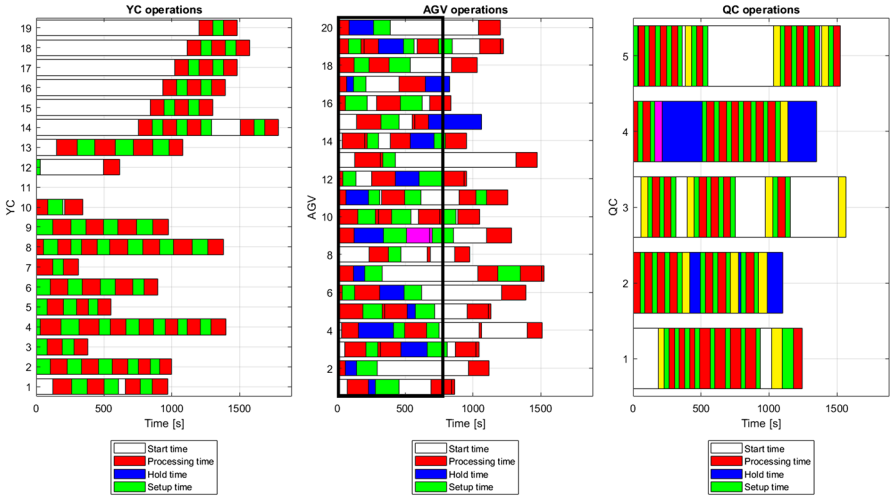

Fig. 6 A schedule of the YC, Lift-AGV and QC, the yellow boxes represent containers that cause delay in the QC schedule. The pink box represents the container that causes the most delay in the QC schedule. Processing time is the productive working time of a machine on a container. Setup represents the driving empty time between two containers. Hold time is the time that a machine cannot proceed with its next operation since there are no discharge opportunities for the current operation. Within white areas a machine is idle

in the QC schedule. These containers are marked yellow in Fig. 6. The longer the delay, the more likely it becomes to select that container. Randomness is important to promote diversity and to prevent the algorithm from getting stuck in a local minimum. Let us assume that the container marked pink in Fig. 6 is selected. This becomes the first of two containers. The second step is to select another container with some slack. This is done by selecting the containers with slack within the black box in Fig. 6. The black box is drawn surrounding the first selected container. Thereafter, the more slack a container has, within the black box, the more likely it is to be selected. This selection ensures that two containers are selected which are likely to reduce the objective value the most. During the selection process, machine eligibility is taken into account. Further, containers that are already in progress cannot be reassigned to another machine since preemption is not allowed.

5.3 Neighbourhood generation

We next derive a new solution that reassigns the previously-selected containers. The solution is first split into two parts. Namely, the sequence of operations (the discrete variables) and the associated starting times (the continuous variables). Subsequently, the sequence of operations is altered by swapping or inserting one container after another. There are four different options to generate a neighbouring solution from the initial solution. First, either the YC or Lift-AGV stage is selected to be modified.

Subsequently, either a swap or an insert should be performed. To decide which of these four options is selected randomness is applied. Generally, inserts are more productive than swaps. This is because inserts directly move containers that cause delays to idle periods. However, it was found that allowing a small portion of swaps does lead to a better performance compared to inserts alone: we presume this is because diversity is maintained. Since containers are selected based on the same characteristics each iteration, it could be that the algorithm is stuck with an unproductive insert. However, because there is always a chance to select another option and because of swaps, the algorithm can break out of a stuck pattern. Note that, at each iteration, only one swap/insert can be performed on either the YC or Lift-AGV stage. This is important since it ensures that the generated neighbouring sequence remains similar to the original sequence.

5.4 Repair

The downside of performing a swap or insert in the sequence of operations is that it could produce an infeasible sequence of operations. This is because precedence or blocking constraints could be violated. Whenever this happens, the sequence of operations should be repaired. We search in the space of feasible solutions only because otherwise the continuous optimisation is not able to find feasible solutions and thus a quality of the solutions cannot be obtained. Further, by ensuring that the solution is always feasible we are able to stop the optimisation process at any time. Finally, once we would proceed with infeasible solutions it is likely to drift away from the feasible region quickly making it harder to restore the solution. An example of the repair process is presented in Table 3. These types of errors are easily repaired by sorting the containers on the machine to be repaired in the same order as they are on the sequence that is used to fix the faulty sequence. This type of fix is also indicated in Table 3.

5.5 Solution evaluation

With all sequence variables feasibly assigned, we rapidly determine the optimal starting time of each operation using linear programming (LP). The starting times also determine the quality of the new solution, according to the objective function (1).

Table 3 Left: A faulty sequence. Right: A repaired sequence

AGV1	2	7	1	9	5	AGV1	1	7	2	9	5
AGV2	6	3	4	8	10	AGV2	6	3	4	8	10
QC1	1	2	3	4	5	QC1	1	2	3	4	5
QC2	6	7	8	9	10	QC2	6	7	8	9	10

The sequence in the left is faulty since the order of containers handled on AGV1 is different from the order of handled containers on QC1

Table 4 Parameters used in the temperature schedule by the SA algorithm

Start temperature	100
Multiplication	0.5
Reheat	75
Iterations till multiplication	7
Iterations till reheat	50

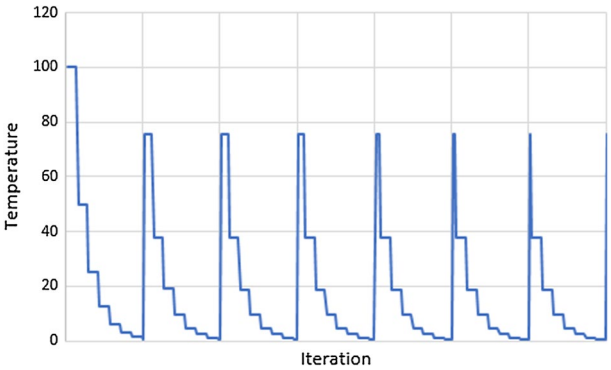

Fig. 7 The temperature schedule used by the SA algorithm

5.6 Acceptance criterion

Should the neighbour solution is an improvement with respect to the previous solution, the neighbour solution is automatically accepted. Otherwise, the probability of accepting the new solution is dependent on its quality and the temperature, as usual in SA (Eq. 22).

$$P_{accept}(T) = e^{\frac{f(\mathbf{x})-f(\mathbf{y})}{T}} \tag{22}$$

The temperature is dependent on a temperature schedule that progresses along with the iterations. Figure 7 depicts the temperature schedule used in our experiments, aiming to balance exploration and exploitation in the metaheuristic. We performed an initial exploratory analysis to derive the temperature schedule parameters subsequently used in all our experiments. Table 4 gives the values.

5.7 Stopping criterion

The optimization cycle is terminated whenever one of the two stopping criteria is encountered. The first stopping criterion is a maximum number of iterations. This maximum number of iterations is determined based on exploratory experiments studying the performance of the algorithm with respect to the number of iterations, as shown in Fig. 8. We selected 550 iterations since the possible further gain in performance was founded to be limited, and used this limit as the default in our

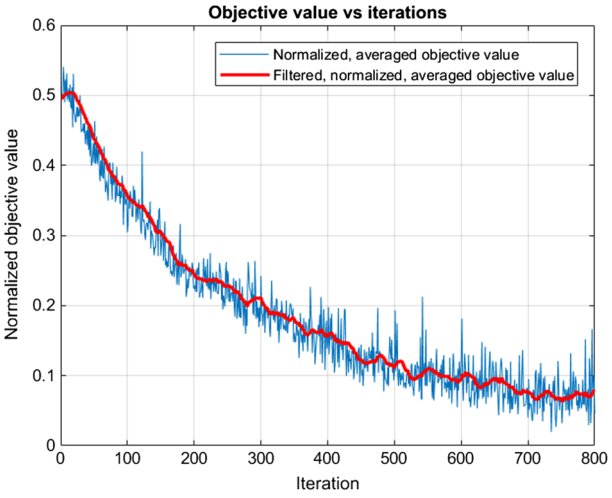

Fig. 8 Averaged, normalized progress of the objective value of 40 different instances with 5 QCs

experiments. The second criterion specifies termination whenever there are no delays in the QC schedule. This is determined during the selection procedure, once we cannot find any containers that are associated with delays in the QC schedule.

6 Experimental results

We perform an empirical analysis of the performance of our solving approach for the coordinated individual container scheduling and dispatching problem. We study the performance versus the theoretical optimum, obtained with the exact solution technique Branch and Bound (B&B). Second, we study the scaling of our algorithm from small to large, realistic terminal sizes. Third, we study the performance of a short and longer computational time limit.

Table 5 reports the results. For SA, we report averages over fifty runs. For B&B we use Gurobi MIP solver version 7.5.2 with default settings. For SA, the 'quick solution' is our algorithm with the termination criteria stated in the last section; the Gurobi LP solver is used for the solution quality evaluation. The 'better solution' is our algorithm with the number of iterations increased from 550 to 1000, and the whole SA restarted 10 times with different initial solutions. The optimality gap is measured according to the objective function (Eq. 1), while the makespan gap which is measured according to the difference in makespan. The computational times shown are obtained with an Intel i7-7700HQ 2.8 GHz machine with 8 cores and 8 GB memory.

The table shows that, in contrast to standard B&B, the computational times for our SA approach can solve the expressive HFS model in less than 30 seconds, even for the largest terminal sizes. This verifies that our approach is feasible for repetitive

Table 5 Optimization results for 24 different sized instances

Problem size				Branch and bound			Simulated annealing quick solution					Simulated annealing better solution				
QC	AGV	YC	CNT	Objective value	Makespan (MS) (s)	Time (s)	Objective value	Makespan (MS) (s)	Time (s)	Opt gap	MS gap	Objective value	Makespan (MS) (s)	Time (s)	Opt gap	MS gap
1	4	3	10	7299.8	1568	0.1	7313.4	1568.0	1.1	0%	0%	7313.4	1568	21.7	0%	0%
2	8	5	20	19,316.8	1784	0.9	20,727.3	1968.8	1.3	7%	10%	19,429.4	1816	23.9	1%	2%
3	12	8	30	34,791.4	1896	81.3	35,928.0	1916.3	1.4	3%	1%	35,524.8	1876	29.5	2%	− 1%
4	16	10	40	41,764.1	1960	71.1	42,336.5	2014.0	1.8	1%	3%	42,310.9	2008	29.7	1%	2%
5	20	13	50	50,791.1	1910	2928.4	52,934.4	2001.6	2.8	4%	5%	51,517.2	1917	50.1	1%	0%
6	24	15	60	63,987.5	2093	3762.8	66,146.0	2175.4	2.6	3%	4%	65,900.6	2169	43.1	3%	4%
7	28	18	70	69,608.4	2373	7206.1	76,446.7	2547.3	2.9	10%	7%	73,937.1	2373	52.2	6%	0%
8	32	20	80	90,604.2	2340	7206.1	95,027.8	2352.7	3.4	5%	1%	93,617.7	2340	57.6	3%	0%
9	36	23	90	–	–	–	10,2783.1	3330.0	3.8	–	–	10,1007	3330	64.3	–	–
10	40	25	100	–	–	–	10,5091.0	2163.6	4.3	–	–	10,3365.4	2141	69.6	–	–
11	44	28	110	–	–	–	11,0276.5	2368.6	4.9	–	–	10,9714.6	2369	78.0	–	–
12	48	30	120	–	–	–	13,1902.6	2827.1	5.6	–	–	12,8267.2	2634	84.3	–	–
13	52	33	130	–	–	–	14,2891.1	2652.3	6.2	–	–	14,0837.6	2468	100.5	–	–
14	56	35	140	–	–	–	16,4616.8	2998.5	6.7	–	–	16,3711.6	2787	108.7	–	–
15	60	38	150	–	–	–	15,9693.0	2416.6	7.3	–	–	15,7237.3	2397	117.0	–	–
16	64	40	160	–	–	–	16,8870.9	2576.0	8.3	–	–	16,4655.5	2550	135.8	–	–
17	68	43	170	–	–	–	19,3742.6	2600.7	9.2	–	–	19,0296.5	2599	148.6	–	–
18	72	45	180	–	–	–	21,2019.8	2576.4	9.8	–	–	20,6932.7	2420	169.3	–	–
19	76	48	190	–	–	–	21,9356.3	2900.9	11.0	–	–	21,3542.3	2816	178.4	–	–
20	80	50	200	–	–	–	21,4759.8	2414.3	11.8	–	–	20,9483.9	2404	194.2	–	–
21	84	53	210	–	–	–	24,5368.3	3004.8	12.8	–	–	23,9076	2995	216.6	–	–

Table 5 (continued)

Problem size				Branch and bound			Simulated annealing quick solution					Simulated annealing better solution				
QC	AGV	YC	CNT	Objective value	Makespan (MS) (s)	Time (s)	Objective value	Makespan (MS) (s)	Time (s)	Opt gap	MS gap	Objective value	Makespan (MS) (s)	Time (s)	Opt gap	MS gap
22	88	55	220	–	–	–	24,5070.2	2235.8	14.8	–	–	23,7181.5	2163	254.0	–	–
23	92	58	230	–	–	–	29,0627.3	2939.7	15.7	–	–	27,8780.2	2930	282.4	–	–
24	96	60	240	–	–	–	30,2322.3	2762.0	21.0	–	–	29,7615	2609	398.9	–	–

The SA results have a quick solution (1 replication 550 iterations) and a better solution (10 replications 1000 iterations). The number of AGVs, YCs and containers (CNT) is a function of the number of QCs. '–' indicates that no results were found within 7200 min

Table 6 Sensitivity results for 24 different sized instances based on 50 samples

Problem size				Simulated annealing quick solution			Simulated annealing better solution		
QC	AGV	YC	CNT	Standard deviation of the objective value	Standard deviation of the makespan (s)	Standard deviation of the time (s)	Standard deviation of the objective value	Standard deviation of the makespan (s)	Standard deviation of the time (s)
1	4	3	10	0.00	0.00	0.05	0.00	0.00	0.34
2	8	5	20	619.71	80.69	0.12	300.06	37.01	0.36
3	12	8	30	616.52	87.99	0.08	4.81	0.00	0.82
4	16	10	40	33.73	25.69	0.12	0.04	0.00	1.09
5	20	13	50	1070.72	88.51	0.10	70.02	0.00	0.37
6	24	15	60	336.54	31.46	0.11	11.11	0.00	1.16
7	28	18	70	1274.80	109.49	0.06	464.45	9.40	0.27
8	32	20	80	668.06	41.07	0.07	63.84	0.00	0.55
9	36	23	90	670.57	0.00	0.07	264.00	0.00	0.53
10	40	25	100	768.38	31.86	0.08	105.17	0.00	1.30
11	44	28	110	487.94	5.80	0.17	15.20	0.00	1.14
12	48	30	120	1013.32	82.76	0.25	471.49	3.76	2.13
13	52	33	130	1445.30	112.68	0.15	24.08	0.00	0.78
14	56	35	140	614.77	120.14	0.15	39.55	0.00	1.48
15	60	38	150	1340.80	40.81	0.16	90.45	0.00	1.11
16	64	40	160	1430.31	21.42	0.17	605.69	8.63	1.86
17	68	43	170	1791.81	12.02	0.14	46.79	0.00	1.57
18	72	45	180	2166.08	50.02	0.17	176.03	0.57	3.42
19	76	48	190	2045.49	76.92	0.21	566.88	0.00	2.74
20	80	50	200	2202.30	23.82	0.17	196.46	0.00	0.43
21	84	53	210	2381.09	12.66	0.20	300.60	0.00	0.25

Table 6 (continued)

Problem size				Simulated annealing quick solution			Simulated annealing better solution		
QC	AGV	YC	CNT	Standard deviation of the objective value	Standard deviation of the makespan (s)	Standard deviation of the time (s)	Standard deviation of the objective value	Standard deviation of the makespan (s)	Standard deviation of the time (s)
22	88	55	220	2697.00	80.95	0.20	628.08	0.00	1.70
23	92	58	230	2928.11	39.65	0.19	1243.67	0.00	0.63
24	96	60	240	2049.02	42.81	0.25	7.87	38.91	1.54

The SA results have a quick solution (1 replication 550 iterations) and a better solution (10 replications 1000 iterations). The number of AGVs, YCs and containers (CNT) is a function of the number of QCs

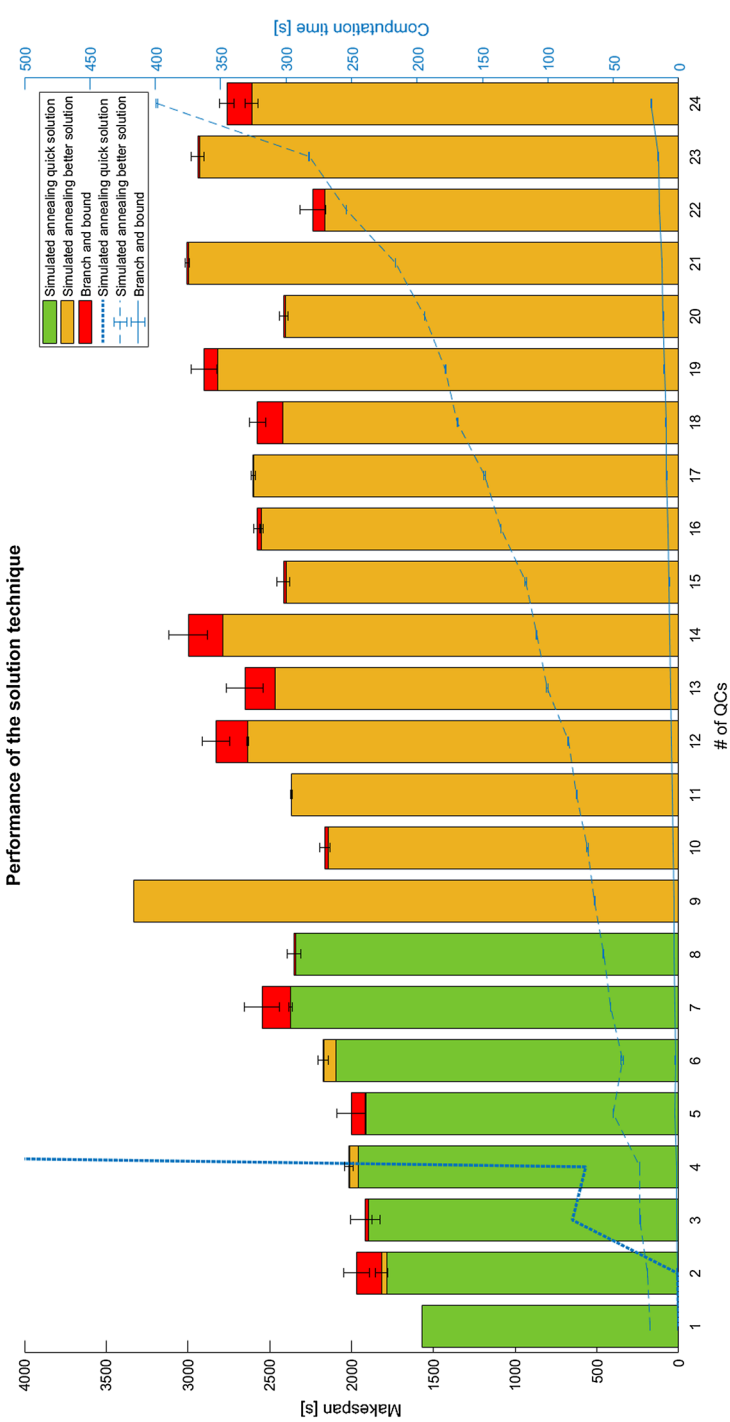

Fig. 9 Performance of both variants of the SA optimisation algorithm and with respect to the B&B solution for smaller sized problems. The standard deviation of the algorithm is also indicated. The left y-axis is makespan; the right y-axis is computational time

continuous re-planning on an operating container terminal. Further, the optimality gap both in terms of objective function and in terms of makespan.

Second, allowing a greater number of iterations and using re-starts, we can achieve further performance improvements with our approach, as seen in the 'better solution' columns. The time to arrive at the solution increases, but remains well under 10 minutes even for the largest terminal sizes. While for operational terminals it is advised to use the 'quick solution', the 'better solution' variant verifies that the SA algorithm is able to find a solution with the shortest makespan in most cases.

The success of our algorithmic design is observed in that the computational time of the SA quick solution is much shorter compared to the SA better solution and B&B solution without leading to huge optimality gaps. Moreover, the size of the problem does not effect the optimality gap.

Table 6 reports the standard deviation of the algorithms. The standard deviation of both the objective value and the makespan is smaller while using the 'better solution' compared to the 'quick solution' since the best obtained solution is used from a set of 10 replications. This decreases the likelihood of finding outliers on the upper side. Further, we observe that the standard deviation of the computational time increases with increasing problem sizes. However, the problem could be very different for each number of QCs. This explains why certain problem sizes do not follow this pattern. Moreover, the standard deviation of the computational time of the 'quick solution' is smaller compared to the computational time of the 'better solution'. This is because with more allowed iterations it is more likely that the number of required iterations until the solution is converged is different. Figure 9 shows both performance and standard deviation in graphical format.

7 Conclusion and outlook

This article integrated the operations of three different types of equipment operating on the waterside of a container terminal. Our integrated model encompasses YCs, Lift-AGVs and QCs that can handle both individual and paired twinlift containers. We constructed a coordinated schedule based on an extended version of a Hybrid Flow Shop. In order to model realistic terminal operations, we added two new features over previous HFS models: fully bi-directional flows and job pairs. The former enables jobs to move freely through the HFS in both directions, while the latter constrains certain jobs to be performed simultaneously by a single machine.

We solved the expressive model using a tailored Simulated Annealing approach. First, we assign the discrete variables uses a local search method to find better alternatives to the current solution by predictive reassignment of jobs to machines. Second, we solve to optimality the continuous variables using LP. Computational results demonstrated that the developed solution technique is capable of finding solutions with relatively small optimality gaps, within a fraction of the time that B&B requires, even for very large terminals.

Our work opens multiple avenues for future work. First, we are coupling our scheduling and dispatching algorithm to a real terminal, in order to validate the schedule and to enable comparison with the current heuristic scheduling techniques applied in real operation. Second, our approach can be further tuned by a sensitivity analysis and dynamic optimization of the meta-parameters to the problem instance at hand. Third, additional features of the terminal operations can be included in the model, such as battery recharging for AGVs, or bay scheduling. The landside operations of the terminal can also be considered (Petering et al. 2017). Lastly, a full study of the multi-objective aspects of container terminal operations is relevant for evaluating trade-offs among solutions (Yu et al. 2017).

Acknowledgements This work was supported in part by TBA Group. We thank the anonymous reviewers for their suggestions.

References

Al-Dhaheri N, Diabat A (2015) The quay crane scheduling problem. J Manuf Syst 36:87–94

Bierwirth C, Meisel F (2010) A survey of berth allocation and quay crane scheduling problems in container terminals. Eur J Oper Res 202(3):615–627

Bish EK (2003) A multiple-crane-constrained scheduling problem in a container terminal. Eur J Oper Res 144(1):83–107

Cao B, Uebe G (1995) Solving transportation problems with nonlinear side constraints with tabu search. Comput Oper Res 22(6):593–603

Cao J, Shi Q, Lee DH (2010a) Integrated quay crane and yard truck schedule problem in container terminals. Tsinghua Sci Technol 15(4):467–474

Cao JX, Lee DH, Chen JH, Shi Q (2010b) The integrated yard truck and yard crane scheduling problem: benders' decomposition-based methods. Transp Res Part E Logist Transp Rev 46(3):344–353

Chen L, Bostel N, Dejax P, Cai J, Xi L (2007) A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. Eur J Oper Res 181(1):40–58

Chen L, Langevin A, Lu Z (2013) Integrated scheduling of crane handling and truck transportation in a maritime container terminal. Eur J Oper Res 225(1):142–152

De Koster M, Balk B, Van Nus W (2009) On using DEA for benchmarking container terminals. Int J Oper Prod Manag 29(11):1140–1155

Dik G, Kozan E (2017) A flexible crane scheduling methodology for container terminals. Flex Serv Manuf J 29(1):64–96

Dragović B, Tzannatos E, Park NK (2017) Simulation modelling in ports and container terminals: literature overview and analysis by research field, application area and tool. Flex Serv Manuf J 29(1):4–34

Gracia MD, González-Ramírez RG, Mar-Ortiz J (2017) The impact of lanes segmentation and booking levels on a container terminal gate congestion. Flex Serv Manuf J 29(3–4):403–432

Gupta JN (1988) Two-stage, hybrid flowshop scheduling problem. J Oper Res Soc 39:359–364

Gurning ROS (2000) Value added service strategy for Jakarta international container terminal: comparative study of value added services in the ports of Rotterdam, Malmo and Aarhus. Master thesis, World Maritime University, Malmö, Sweden

He J, Huang Y, Yan W, Wang S (2015) Integrated internal truck, yard crane and quay crane scheduling in a container terminal considering energy consumption. Expert Syst Appl 42(5):2464–2487

Huang SY, Li Y (2017) Yard crane scheduling to minimize total weighted vessel loading time in container terminals. Flex Serv Manuf J 29(3–4):689–720

Iris Ç, Pacino D, Ropke S, Larsen A (2015) Integrated berth allocation and quay crane assignment problem: set partitioning models and computational results. Transp Res Part E Logist Transp Rev 81:75–97

Jin JG, Lee DH, Cao JX (2014) Storage yard management in maritime container terminals. Transp Sci 50(4):1300–1313
Jin JG, Lee DH, Hu H (2015) Tactical berth and yard template design at container transshipment terminals: a column generation based approach. Transp Res Part E Logist Transp Rev 73:168–184
Jungwattanakit J, Reodecha M, Chaovalitwongse P, Werner F (2009) A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria. Comput Oper Res 36(2):358–378
Kaveshgar N, Huynh N (2015) Integrated quay crane and yard truck scheduling for unloading inbound containers. Int J Prod Econ 159:168–177
Kim KH, Bae JW (1998) Re-marshaling export containers in port container terminals. Comput Ind Eng 35(3–4):655–658
Kim KH, Bae JW (2004) A look-ahead dispatching method for automated guided vehicles in automated port container terminals. Transp Sci 38(2):224–234
Kim KH, Park YM (2004) A crane scheduling method for port container terminals. Eur J Oper Res 156(3):752–768
Kim KH, Park YM, Ryu KR (2000) Deriving decision rules to locate export containers in container yards. Eur J Oper Res 124(1):89–101
Kizilay D, Eliiyi DT, Van Hentenryck P (2018) Constraint and mathematical programming models for integrated port container terminal operations. In: Proceedings of the 15th international conference on the integration of constraint programming, artificial intelligence, and operations research (CP-AI-OR). Springer, pp 344–360
Lau HY, Zhao Y (2008) Integrated scheduling of handling equipment at automated container terminals. Int J Prod Econ 112(2):665–682
Lim YF, Zhang Y, Wang C (2015) A quay crane system that self-recovers from random shocks. Flex Serv Manuf J 27(4):561–584
Lin SW, Ting CJ, Wu KC (2018) Simulated annealing with different vessel assignment strategies for the continuous berth allocation problem. Flex Serv Manuf J 30(4):740–763
Liu C, Xiang X, Zheng L (2017) Two decision models for berth allocation problem under uncertainty considering service level. Flex Serv Manuf J 29(3–4):312–344
Lu Y, Le M (2014) The integrated optimization of container terminal scheduling with uncertain factors. Comput Ind Eng 75:209–216
Magalhães-Mendes J (2011) Active, parameterized active, and non-delay schedules for project scheduling with multi-modes. In: Proceedings of the 16th WSEAS international conference on applied mathematics, pp 29–31
Petering ME, Wu Y, Li W, Goh M, de Souza R, Murty KG (2017) Real-time container storage location assignment at a seaport container transshipment terminal: dispersion levels, yard templates, and sensitivity analyses. Flex Serv Manuf J 29(3–4):369–402
Pinedo ML (2016) Scheduling: theory, algorithms, and systems. Springer, Berlin
Qiu L, Hsu WJ (2000) Conflict-free AGV routing in a bi-directional path layout. In: Proceedings of the 5th international conference on computer integrated manufacturing, vol 1, pp 392–403
Reveliotis SA (2000) Conflict resolution in AGV systems. IIE Trans 32(7):647–659
Ribas I, Leisten R, Framiñan JM (2010) Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. Comput Oper Res 37(8):1439–1454
Ruiz R, Vázquez-Rodríguez JA (2010) The hybrid flow shop scheduling problem. Eur J Oper Res 205(1):1–18
Saanen Y (2016) AGV versus lift AGV versus ALV. Port Technol Int 70:63–69
Schroër HJ, Corman F, Duinkerken MB, Negenborn RR, Lodewijks G (2014) Evaluation of inter terminal transport configurations at Rotterdam Maasvlakte using discrete event simulation. In: Proceedings of the winter simulation conference 2014. IEEE, pp 1771–1782
Stahlbock R, Voß S (2008) Operations research at container terminals: a literature update. OR Spectr 30(1):1–52
Steenken D, Voß S, Stahlbock R (2004) Container terminal operation and operations research—a classification and literature review. OR Spectr 26(1):3–49
Tavakkoli-Moghaddam R, Makui A, Salahi S, Bazzazi M, Taheri F (2009) An efficient algorithm for solving a new mathematical model for a quay crane scheduling problem in container ports. Comput Ind Eng 56(1):241–248
Vignier A, Billaut JC, Proust C (1999) Les problèmes d'ordonnancement de type flow-shop hybride: état de l'art. RAIRO Oper Res 33(2):117–183

Vuuren B (2017) Route plan scheduling for automated guided vehicles at container terminals. Master's thesis, TU Delft, Delft, The Netherlands
Xin J (2015) Control and coordination for automated container terminals. Ph.D. thesis, TU Delft, Delft, The Netherlands
Xin J, Negenborn RR, Lodewijks G (2014) Energy-aware control for automated container terminals using integrated flow shop scheduling and optimal control. Transp Res Part C Emerg Technol 44:214–230
Xin J, Negenborn RR, Corman F, Lodewijks G (2015a) Control of interacting machines in automated container terminals using a sequential planning approach for collision avoidance. Transp Res Part C Emerg Technol 60:377–396
Xin J, Negenborn RR, Lodewijks G (2015b) Event-driven receding horizon control for energy-efficient container handling. Control Eng Practice 39:45–55
Yang CH, Choi YS, Ha TY (2004) Simulation-based performance evaluation of transport vehicles at automated container terminals. OR Spectr 26(2):149–170
Yu S, Wang S, Zhen L (2017) Quay crane scheduling problem with considering tidal impact and fuel consumption. Flex Serv Manuf J 29(3–4):345–368
Zhang C, Liu J, Wan YW, Murty KG, Linn RJ (2003) Storage space allocation in container terminals. Transp Res Part B Methodol 37(10):883–903

T. Jonker received his MS in Mechanical Engineering from the Faculty of Mechanical Maritime, and Materials Engineering, TU Delft, Netherlands, in 2018. He is currently pursuing another MS degree in econometrics at the Erasmus University Rotterdam (EUR).

M. B. Duinkerken is an Associate Professor at the TU Delft. His current research interests are operations research, maintenance, and simulation.

N. Yorke-Smith received his PhD in artificial intelligence from Imperial College London in 2004. He is an Associate Professor at TU Delft, Netherlands. His current research interests are agent- and data-driven approaches to complex scheduling problems.

A. de Waal received his MS in Operations Research from the University of Amsterdam, Netherlands, 2000. He is head of simulations at TBA, which specializes in ports, terminal, and warehousing operations.

R. R. Negenborn (TU Delft) is full professor in Multi-Machine Operations & Logistics. He is head of the Section Transport Engineering & Logistics of Department Maritime & Transport Technology. His research interests include automatic control and coordination of transport technology (including autonomous vessels) in general, whereby he proposes multi-agent system and model predictive control approaches that benefit from real-time information availability and the potential of communication. As such, his research anticipates the massive introduction of sensing, computation, and communication technologies. Within the 4S Framework this is materialized into innovative solutions for smart equipment, transport hubs, ports and (synchromodal) networks. He has over 200 peer reviewed academic publications. He leads NWO, EU and industry funded research, and is on the editorial board of the series on "Intelligent Systems, Control and Automation: Science and Engineering". He was moreover general chair of the 6th International Conference on Computational Logistics, has acted as member of the organizing committee of several other international conferences (including IEEE control conferences and maritime systems & logistics conferences) and was guest editor of special journal issues on autonomous vessels and computational logistics. In addition, he is editor of the books "Intelligent Infrastructures", "Distributed Model Predictive Control Made Easy", and "Transport of Water versus Transport over Water".

Affiliations

T. Jonker[1,2] · M. B. Duinkerken[1] · N. Yorke-Smith[3,4] · A. de Waal[2] · R. R. Negenborn[1]

M. B. Duinkerken
M.B.Duinkerken@tudelft.nl

N. Yorke-Smith
n.yorke-smith@tudelft.nl

A. de Waal
arjen.de.waal@tba.group

R. R. Negenborn
R.R.Negenborn@tudelft.nl

1 Department of Maritime and Transport Technology, Delft University of Technology, Delft, The Netherlands

2 Department of Simulation, TBA Group, Delft, The Netherlands

3 Department of Software Technology, Delft University of Technology, Delft, The Netherlands

4 Olayan School of Business, American University of Beirut, Beirut, Lebanon