Transportation Research Part E 72 (2014) 77–100

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

The strategic berth template problem

Akio Imai [a,b,*], Yukiko Yamakawa [a], Kuancheng Huang [c]

[a] Graduate School of Maritime Sciences, Kobe University, Fukae, Higashinada, Kobe 658-0022, Japan
[b] Port and Airport Research Institute, Nagase, Yokosuka 239-0826, Japan
[c] Department of Transportation Technology and Management, National Chiao Tung University, 1001 Ta Hsueh Rd., Hsinchu City 300, Taiwan

ARTICLE INFO

Article history:
Received 19 May 2014
Received in revised form 29 September 2014
Accepted 30 September 2014

Keywords:
Berth template problem
Berth allocation problem
Subgradient optimization
Lagrangian relaxation
Container terminal
Heuristics

ABSTRACT

A marine container terminal operator may have a situation with excessive calling requests to be served especially when some new service contracts are under consideration. For this situation, we propose a strategic berth template problem (BTPS) that selects the ships among the requesting ones to be served and arrange their berth-windows within a limited planning horizon. The BTPS employs the subgradient optimization procedure, which is an improved version of the procedure that the authors developed for the operational berth allocation problem. A wide variety of numerical experiments indicate the improved subgradient procedure works well for the BTPS.

1. Introduction

Sea-borne container shipping plays a major and important role in the world transportation system and the global supply chain. A container terminal, as a nodal point in the transportation network, acts as an interchange of the different modes involved in the overall transportation process. Therefore, efficiency and productivity improvements in the terminal operations are crucial in reducing the overall trip duration and reducing costs and thus have been gaining more attention lately.

The primary aim of a terminal is a quick turnaround or a secured departure deadline of calling ships. Also, the terminal attempts to utilize its costly infrastructure efficiently. Major container ports feature the so-called "multi-user container terminals (MUTs)", which serve a lot of calling ships of different shipping lines with a long quay and vast yard space to provide a huge ship handling capacity. In an era of cost-cutting and competition, shipping lines are less inclined to operate private terminals they used to be (Mongelluzzo, 2013). Due to this growing demand, the need to operate MUTs more efficiently as well as the issues pertaining to the efficient berth scheduling at an MUT have been receiving much attention these days.

Most decision makings can be classified as three broad categories: strategic, tactical and operational. As far as the berth scheduling is concerned, the existing literature may fall into two categories in a relative sense: long-term (tactical) and short-term (operational). Contents of those categories may be summarized as below whilst there exists some diversity in decision making for each category at MUTs.

(1) Tactical berth scheduling (or berth template problem, BTP): finds a set of berth-windows (i.e., berthing locations with the start and end times for service) within the fixed length of planning horizon so as to maximize the service objective.

* Corresponding author at: Graduate School of Maritime Sciences, Kobe University, Fukae, Higashinada, Kobe 658-0022, Japan.
E-mail address: imai@maritime.kobe-u.ac.jp (A. Imai).

http://dx.doi.org/10.1016/j.tre.2014.09.013
1366-5545/

(2) Operational berth scheduling (or berth allocation problem, BAP): finds a set of berth-windows within an open-ended planning horizon so as to maximize the service objective.

As will be discussed in the subsequent section, most of the existing papers about berthing decision fall into the operational scheduling, while some are in the tactical scheduling. Also note that it is common in the literature berth scheduling is considered together with other facilities so as to take into account the actual situations of the related resources jointly operated as well as the updated information of the ships to be served.

For a terminal operator, the service contracts with shipping lines are reviewed and renewed as with a regular interval or whenever needed. Alternatively, the terminal operator may receive the berthing request from a new customer. The typical contract negotiation process between a shipping line and a terminal operator can be illustrated by JICT (2014), a webpage of a South Asian terminal operator. Throughout the negotiation process, the operator arranges a template for berthing. As briefly depicted above, the berth template problem (BTP) in the literature determines the template for berthing, i.e., a set of berth-windows of serving ships during a fixed planning horizon, given a long-term calling request profile from shipping lines. In particular, as the most notable distinction between BTP and BAP, the fixed planning horizon is repeated in a cyclic fashion in the BTP and hereafter referred to as the *cylinder*, as used in a pioneering work on the BTP by Moorthy and Teo (2006).

In general, most container shipping services are provided weekly on a fixed day of the week, thus the BTP normally arranges berth-windows to meet all the calling requests within a week. It is noteworthy that, though not very common, it is possible that a terminal has a different calling ship request profile from one week to another. Then, the cylinder for the BTP should be set as the least common multiple. A terminal operator normally applies the same template every week, though some adjustment may be made to accommodate the irregularity associated with the ship arrivals.

For most of the cases for such a new contract or contract renewal, there might be no significant change in the number of overall calling ships when updating the berth template design. This scenario leads to the situation that all calling ship requests can be accommodated in any berth template design. In fact, as will be reviewed in the next section, all of the existing BTP studies assume such a full coverage of calling ships. Their focuses are mainly on reducing the operational cost and/or meeting the requirements/performances of the calling ships.

In contrast, under the inauguration of a new MUT or the completion of major capacity expansion at an existing MUT, the terminal operator may need to design a brand new berth template to incorporate all prospective demands of calling ships. In addition to the decision factors similar to those in the existing BTP literature, the terminal operator may face the issue of excessive demand and require a decision making methodology for determining which part of the demand to be satisfied. This scenario is not addressed in the existing BTP studies. We hereby propose a strategic level of berth scheduling: the strategic berth template problem (BTPS), which chooses ships to be served and those not to be and finds a set of berth-windows for the served ships within a pre-determined fixed length of planning horizon so as to maximize the service objective. For convenience, the BTP at the tactical level addressed in the existing papers is hereafter referred to as BTPT.

This paper introduces an integer programing model for the BTPS and develops a Lagrangian relaxation-based heuristic for it. The contribution of this paper is twofold. One is the introduction of the BTPS concept that deals with the selection of the ships to be served, including the consideration over the mother ship (or shipping line) and the associated feeder under the condition of tight berthing capacity. The other is an approximate solution method for berth scheduling. The BTPS formulation is structured based on the formulation of the dynamic BAP (DBAP) in Imai et al. (2001) to take advantages of the established solution methodology: the subgradient procedure with Lagrangian relaxation. However, we develop new heuristics, on the foundation of the DBAP solving technique, to achieve a better BTPS solution. As will be shown in the numerical experiments, the superiority of these new heuristics for both BTPS and DBAP is demonstrated.

The paper is organized as follows. The next section provides a literature review on the berth scheduling. An integer programming formulation of the BTPS is discussed in Section 3. This is followed by Section 4 which introduces a solution method for the BTPS. In Section 5, a number of computational analyses are carried out, while the final section concludes the paper.

2. Literature review

As the issues related to efficient terminal operations have been constantly gaining importance, there have been a growing number of studies that deal with the BAP models. On the other hand, the BTP is a relatively new research topic with few research works. These two types of problems are reviewed in this section. In particular, to the authors' knowledge, there is no existing BTP research work that focuses on the strategic decision of selecting the shipping lines such as the BTPS in this study.

One of the earliest works of the BAP is Imai et al. (1997) who addressed a BAP in discrete location indices (hereafter referred to as BAPD in this section) for commercial ports. Most service queues are in general processed on an FCFS (First-Come-First-Served) basis. They concluded that in order to achieve high port productivity, an optimal set of ship-to-berth assignments had to be determined, instead of considering the FCFS rule. Their study assumed a static situation where ships to be served for a planning horizon had all arrived at a port before one planed the berth allocation. Thus, their study can be applied only to tremendously busy ports. As far as container shipping is concerned, such busy ports are neither competitive

nor realistic because of the long delay in the interchange process at ports. In this context, Imai et al. (2001, 2005a) extended the static version of the BAPD to a dynamic treatment that is similar to the static treatment, but with the difference that some ships arrive while work is in progress. Due to the difficulty in finding an exact solution, they developed a heuristic by using a subgradient method with the Lagrangian relaxation. Their study assumed the same water depth for all the berths, while in practice there are berths with different water depths in certain ports. Nishimura et al. (2001) further extended the dynamic version of the BAPD for the multi-water depth configuration. They employed genetic algorithm (GA) to solve that problem. In some real situations, the terminal operator assigns different priorities to calling vessels. For instance, at a terminal in China, small feeder ships have priority, as handling work associated with them is completed in a short period of time and larger vessels do not have to wait for a long time. On the other hand, a terminal in Singapore treats large vessels with higher priority because they are good customers to the terminal. Imai et al. (2003) extended the dynamic BAPD in Imai et al. (2001, 2005a) to treat the ships with different priorities and see how the extended BAPD differentiates the handling of ship in terms of the service time associated with ships. Imai et al. (2007) proposed the BAPD with simultaneous berthing of multiple ships at the indented berth, which was potentially useful for fast turnaround of mega-containerships. Cordeau et al. (2005) developed a tabu search heuristic for the dynamic BAP in two versions with both discrete and continuous location indexes. They analyzed the solution quality of the proposed heuristic for the discrete location with the exact solution by CPLEX; however, the applied problem cases were relatively small sized ones. Monaco and Sammarra (2007), inspired by the dynamic BAPD of Imai et al. (2001), proposed an improvement in its formulation and also developed the Lagrangian relaxation-based subgradient optimization, which was the same approach for Imai et al. (2001, 2005a) but with some modifications. Imai et al. (2001, 2005a) proposed three heuristics embedded in the subgradient procedure. Monaco and Sammarra reported that their algorithm outperformed that of Imai et al. (2001, 2005a). However, they did not mention which one of the three heuristics embedded in the subgradient procedure in Imai et al. (2001, 2005a) was used for performance comparison. Hansen et al. (2008) developed a variable neighborhood search method for the BAPD. Mauri et al. (2008) applied the Population Training Algorithm with Linear Programming to the dynamic BAPD, which was formulated in Cordeau et al. (2005). Imai et al. (2008) extended the BAPD developed in Imai et al. (2001, 2005a) for a terminal who assigned some calling ships to another terminal when the terminal was congested. Golias et al. (2009) proposed the dynamic BAPD with customer service differentiation based on respective agreements. They formulated their BAPD as a multi-objective problem and developed a GA-based heuristic. They also proposed, in Golias et al. (2010), another heuristic based on a lamda optimal. Buhrkal et al. (2011) treated the dynamic BAPD and formulated the problem as the improved heterogeneous VRP with time windows based on the discrete version of BAP of Cordeau et al. (2005). Saharidis et al. (2010) proposed a hierarchical optimization for the BAPD with two conflicting objectives terminal operators face. Xu et al. (2012) proposed the BAPD with different water depths at berths and tidal condition. Imai et al. (2013) discussed a terminal efficiency in terms of berthing ships in different types of innovative terminal designs by comparing the total service time of calling ships when their berth-windows are optimally scheduled with ad-hoc BAPDs for those different terminals. Recently some studies such as de Oliveira et al. (2012), Lalla-Ruiz et al. (2012), and Ting et al. (2014) proposed new heuristics for the dynamic BAPD that had been discussed in Imai et al. (2001), Cordeau et al. (2005), and Monaco and Sammarra (2007). All the three papers tested their heuristics with problem instances that were provided in Cordeau et al. (2005). Ting et al. (2014) indicated that their algorithm outperformed the others.

There is another type of the BAP, which is the one with a continuous location index (referred to as BAPC). In the aforementioned studies the entire terminal space is partitioned into several parts (or berths) and the allocation is planned based on the divided berth space. This may result in having some berthing space unused. Under the continuous location approach, ships are allowed to be served wherever the empty spaces are available to physically accommodate the ships via a continuous location system. This type of problem resembles more or less the cutting-stock problem where a set of commodities is packed into some boxes in an efficient manner. A ship in service at a berth can be shown by a rectangle in a time–space representation, therefore efficient berth usage is a sort of packing "ship rectangles" into a berth-time availability as a box with some limited packing scheme such as that no rotation of ship rectangles is allowed. For this type of the BAP, Lim (1998) first addressed a problem with the objective of minimizing the maximum amount of quay space used at any time with the assumption that once a ship is berthed, it will not be moved to any place else along the quay before it departs. He also assumed that every ship was berthed as soon as it arrived at the port. On the other hand, Li et al. (1998) solved the BAPC both with and without the ship's movement restriction. Their objective is to minimize the makespan of the schedule. Park and Kim (2002) developed a subgradient procedure with Lagrangian relaxation for the BAPC. Imai et al. (2005b) addressed a BAPC, but with a major difference from the other BAPCs in that the handling time depended on the berthing location of ship. They developed a heuristic for that problem in cooperation with a heuristic for the dynamic BAPD in Imai et al. (2001, 2005a). The conclusion of their study was that the best approximate solution was identified with the best solution in discrete location where the berth length was the maximum length of ships involved in the problem. This implies that the solution in discrete location is applicable for practice in berth allocation planning and the improved solution can be obtained from the solution in discrete location. As mentioned before, Cordeau et al. (2005) developed a tabu search heuristic for the dynamic BAP in both discrete and continuous location indexes. For the continuous location version, the solution quality was assessed by comparisons with solutions by the straightforward heuristic. Lee et al. (2010) developed two greedy randomized adaptive search procedures for the BAPC. Cheong et al. (2010) dealt with BAPC with multi-objectives of makespan, waiting time and degree of deviation from a predetermined priority schedule. They developed a multi-objective evolutionary algorithm for that problem.

There are quite few papers dealing with the tactical berth scheduling. Moorthy and Teo (2006) was the first one to present the BTP, by which this study is greatly inspired. Their BTP defines berth-windows of serving calling ships in a continuous space within the predetermined length of the planning horizon. The berth template design takes into account the scheduling of periodicity, that is, the wrap-around effect of the cylinder. Their problem had two objectives: one is to maximize the service level, which is simply defined as the percentage of vessels served within two hours of their arrival, and the other is to minimize the connectivity cost, which is related to the distances between berths within vessel transshipment groups. Another tactical berth scheduling problem is studied by Giallombardo et al. (2010). They proposed the BTPT in discrete location indexes with the integration of quay crane (QC) allocation decision. In addition to the cost associated with QCs, they introduced the other cost component in the objective function, the yard cost that depends on the berthing location. Their study arranges all berth-windows within the time duration, similar to the concept of the cylinder length. Zhen et al. (2011) proposed an integrated template planning model for both berthing location in continuous indexes and yard container stack arrangement. In addition, the cyclic scheduling consideration and the QC allocation were considered. They developed a heuristic with a recursive process based on two stages: berth template and yard template. Hendriks et al. (2012) addressed a BTPS under a unique berthing service circumstance where ships can berth at any terminal in a port with inter-terminal service agreements, which allow containers to be unloaded from a ship at a remote partner terminal and transferred by trucks to the terminal the ship was originally scheduled to berth. Their BTP implicitly imposed the cylinder on the model since it assumed to serve cyclically calling ships. It takes into account the QC assignment to ships, resulting in the inclusion of the associated QC utilization cost in the objective function, which also considers the inter-terminal container transfer cost. Whereas their model is referred to as strategic, it may be categorized into a tactical model according to our hierarchical scheme of berth scheduling since all the ship calls are assumed to be served. Thus, it still can be thought as a tactical model if categorized by our hierarchical scheme of berth scheduling. Hendriks et al. (2013) addressed a BTPT that deals with berth allocation and yard planning within the cylinder. Lee and Jin (2013) studied a BTPT for feeder vessels to determine berth allocation for feeders in discrete locations and yard storage assignment for their transshipment cargoes. Whereas it considered cyclically calling feeders, it did not impose the cylinder on the model. Following the framework of Giallombardo et al. (2010), Vacca et al. (2013) developed an exact-solution algorithm based on the technique of branch and price for the integrated problem of berth and QC planning. Their study does not apply the cylinder, within which all berth-windows are planned to be placed. Instead, every ship calling request has a preferred start and end times of the handling service. This preferred time duration is wide enough to place an actual berth-window of the ship appropriately so as to minimize the objective function. Finally, note that all the above BTP studies implicitly assume that the berthing capacity is large enough to cover all the calling requests.

Lastly, we look at the relation between the BTPS and the Machine Scheduling Problem (MSP), both of which may sometimes assume a similar problem framework. As pointed out in Imai et al. (1997) and Imai et al. (2001), there is a similarity between the BAP and the Parallel MSP. Since the BTPS is an extension of the BAP, it corresponds to the MSP with identical machines in parallel (Pinedo, 2012). This scheduling problem for instance was tackled by Norman and Bean (1999). The Periodic (or Cyclic) Machine Scheduling, a specific type of the MSP, may arise from the flow-shop problem with multiple machines in series that serve jobs according to the First-In-First-Out discipline (Pinedo, 2012). In normal job-shop, flow-shop and open-shop environments, the cyclic scheduling is tackled by existing studies such as Matsuo et al. (1991), Roundy (1992), Hanen (1994), McCormick and Rao (1994), Crama and Van de Klundert(1997), Hall et al. (1997), Lee and Posner (1997), Crama et al. (2000), and Brucker and Kampmeyer (2008). However, these research works are significantly different from the BTPS of this study. First of all, their objectives are minimizing the cycle time (or maximizing the throughput as an equivalent) and/or minimizing the flow time (or minimizing the work-in-process). In addition, they do not consider the release time of jobs as the schedule is repeated over and over again based on the concept of the *Minimum Part Set*. In the BTPS, the berth-window is regarded as a job, but with only single process performed by any single berth. In addition, while the berthing service is repeated weekly, each calling ship has a preferred service starting time (the release time in the scheduling terminology), since a ship calls at the terminal within an entire voyage consisting of multiple calling ports.

3. Problem formulations

3.1. Problem overview

The BTPS of this study focuses only on berthing decision. As reviewed in the previous section, a few existing BTPT models not only schedule berth-windows but also make decisions for other facility and/or equipment usage such as yard container slot allocation and QC assignment. These modelling approaches implicitly assume the availability of precise calling ship profiles, such as the amount of cargo to be loaded and unloaded. Otherwise, the ship handling time, an input for the BTPTs, cannot be estimated by linking the number of QCs used for ship handling. For the cases with precise ship information, we think that the BTPT models in these existing papers are very suitable.

The main feature of this study is the strategic selection of the calling ships (or shipping lines) to be served within a spatial (berth) and temporal (cylinder length) berthing capacity of the terminal and the determination of the associated berth-windows for handling the ships. Under the assumption that ship profiles are not necessarily accurate or precise for long-term planning, the BTPS, unlike the BTPT, can exclude other decision making factors such as QC and storage yard arrangement

in the decision process. Nonetheless, in the following tactical phase, additional and more detailed scheduling issues may be taken into account. For example, if the berth template determined in the strategic level is found to be non-executable in the subsequent tactical phase due to QC availability, the terminal operator may invest in some more QCs as a long-term decision or re-arrange the berth-windows subject to the QC restriction based on the approach in those BTPT studies with the integration of the QC decision.

In addition to the issue of data availability assumption, the other issue is about model complexity and computational burden. Indeed, these integrated models are very sophisticated and able to incorporate a wider range of decisions at container terminals. However, the complexity of those models may restrain the capacity for handling large-scale problems. In addition, most of the integrated BTPT models employ a structure of iteratively solving the sub-problems in a sequential fashion. For instance, a model dealing with two kinds of facilities (such as berths and QCs) consists of two sub-models, one for each facility. Before the termination of the two-stage iterative procedure, the solution of the first sub-model is used as the input of the second sub-model, and vice versa. Therefore, if it is needed to integrate the decisions of multiple facilities of a container terminal, there is a good chance that quality solutions can be derived by applying the BTPS of this study and some other well-established solution algorithms for other facilities (such as QC arrangement) in an iterative way.

It may be noteworthy that an example for making a berth template without the consideration of other equipment such as QC is observed at a terminal of the port of Osaka, Japan. The operator makes an initial berth template as information used for the negotiation process to sign the service contracts with shipping lines. However, the template is built based on the ship handling time without taking into account the QCs. Instead, the operator includes some margin (30–40% of the average value) to the ship handling time so as to cope with the variation of the handling time affected by the number of deployed QCs.

The BTPS is basically the same as the BAP but with a cylinder constraint that ensures the berth-windows of all ships are packed within the predetermined cylinder length. As mentioned before, most containerships call at a terminal on weekly basis; the cylinder length is one week in general. Because of its relatively long-term planning aspect, the BTPS assumes the same handling time of a ship regardless of its assigned berth. Thus, for this reason the BTPS aims to minimize the total ship start time delay (TDT), which is the sum of the deviation between ship arrival time and actual ship service start time (or

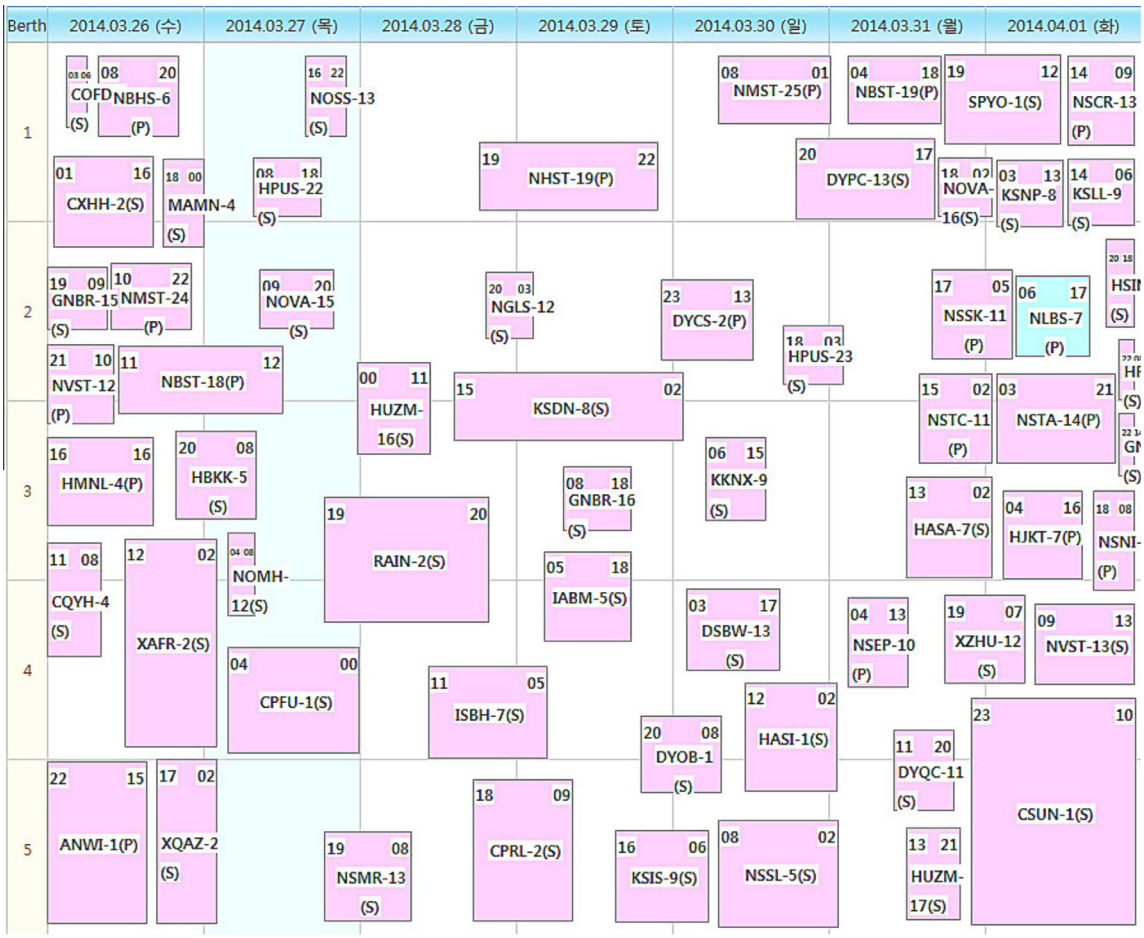


Fig. 1. Berth template. *Source*: Korea Express Busan Container Terminal Co. Ltd.

berthing time), while the BAP minimizes the total ship service time (TST), in which the service time of a ship consists of its handling time and delay (or waiting) time for berthing.

There are two types of berthing location indexes in berth scheduling problems: discrete and continuous. There may be pros and cons of these two schemes. In general, the discrete one has an advantage in computational complexity and a disadvantage in terminal productivity. We adopt the discrete scheme given the following considerations.

(1) The continuous scheme is promising in terms of berth utilization by fully occupying the entire quay space with different ships in length. However, the sophisticated solution with better utilization of quay space from the continuous scheme may result in an unnecessary level of details, given the rough ship profiles assumed in the BTPS.
(2) The discrete scheme may lead to theoretical implications for the solving methods of the BTPS.

In particular, the former restriction can be mitigated by the following observation and countermeasure: As most deep-sea ships are huge not only in capacity but also in length, they normally occupy the whole length of a berth section, typically with length of 400 m. For serving small feeders, it appears less likely to use the slack spaces between big ships. The discrete scheme can become more effective if feeders with close arrival times are grouped and treated as a large ship as long as the combined total length does not exceed the berth length. The above discussion can be supported by an example of the berth template for a five berth-container terminal in the Busan port from March 26th to April 1st of year 2014, as shown in Fig. 1. In this template, each berth accommodates a maximum of two ships at the same time during the congested periods of time. In addition, very few large ships are berthing across berth boundaries.

The BTPS in discrete berthing locations that is discussed in this study is based on the BAP models proposed in Imai et al. (1997) and in Imai et al. (2001). The BTPS assumes (i) all ships can be served at any berth, (ii) each berth serves up to one ship at any time, (iii) handling time for a specific ship is constant regardless of its berthing location, (iv) ships are served within the cylinder, (v) mother and feeder ships with transhipment relations are both served or neither of them is served in order to address the practical issue under the hub-and-spoke operation, and (vi) a ship can be excluded from the berth-window planning at a price of incurring the associated discard cost/penalty.

Assumption (iii) is made since the container storage yard decision is not part of the consideration for optimization in this study. It is assumed that the corresponding yard location can be close to the assigned berth and does not have a serious impact on the handling time, which is consequently assumed to be a constant, regardless of its assigned berth.

Assumption (vi) is made in order for the BTPS to, in addition to berth-window placement, identify ships to be served and those not to be served when the calling request exceeds the berthing capacity at a terminal. By manipulating the penalty level, the priority of the ships of the strategic consideration over the shipping lines can be incorporated into the model. For example, an extremely high value can be imposed on the ships of the customer that the terminal operator cannot afford to lose. In summary, the BTPS can be defined as: select ships to be served or un-served and determine the berth-windows for the served ships in order to minimize the sum of the total penalty for un-served ship and the total cost associated with the deviation between the actual service start times and their preferred start (or arrival) times.

3.2. BTPS formulation

As described before, the BTPS is based on the model structure of DBAP developed in Imai et al. (2001) to take technical advantages for the model formulation and solving tips. For selection of ships to be served or un-served, we prepare a virtual berth (berth zero) in addition to physical berths that actually serve ships. By allocating unserved ships to berth zero, the BTPS can facilitate the ship selection with computational advantages that were found through the DBAP.

For the treatment of mother and feeder ships we define a 0–1 parameter, $R_{jj'}$, which indicates a connection of a pair of ships (one is a mother and the other is a feeder). $R_{jj'} = 1$ if ships j and j' are connected and $R_{jj'} = 0$ otherwise. To avoid the redundancy due to the symmetry of j and j' (i.e., $R_{jj'} = R_{j'j}$), $R_{jj'}$ is defined for $j, j'(>j) \in V$. This scheme does not explicitly distinguish which of j and j' is the mother ship. This allows multiple feeders to belong to a mother ship and/or a feeder ship to belong to multiple mother ships.

[BTPS]

$$\text{Minimize} \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\{(k-1)C_j - A_j\}x_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U} k y_{ijk} + \sum_{j\in V}\sum_{k\in U} G_j x_{0jk} \tag{1}$$

$$\text{subject to} \sum_{i\in B\cup\{0\}}\sum_{k\in U} x_{ijk} = 1 \quad \forall j \in V, \tag{2}$$

$$\sum_{j\in V} x_{ijk} \leqslant 1 \quad \forall i \in B\cup\{0\}, k \in U, \tag{3}$$

$$\sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + y_{ijk} - A_j x_{ijk} \geqslant 0 \quad \forall i \in B, j \in V, k \in U, \tag{4}$$

$$\sum_{j\in V}\sum_{k\in U}(C_j x_{ijk} + y_{ijk}) - Y_i \leqslant CT \quad \forall i \in B, \tag{5}$$

$$Y_i \leqslant \left(1 - \sum_{j\in V} x_{ijk}\right) CT + \sum_{l\in V}\sum_{m\in P_k} (C_l x_{ilm} + y_{ilm}) + \sum_{j\in V} y_{ijk} \quad \forall i \in B,\ k \in U, \tag{6}$$

$$R_{jj'} \sum_{i\in B}\sum_{k\in U} x_{ijk} = R_{jj'} \sum_{i\in B}\sum_{k\in U} x_{ij'k} \quad \forall j, j' (> j) \in V, \tag{7}$$

$$Y_i \leqslant CT \quad \forall i \in B, \tag{8}$$

$$x_{ijk} \in \{0, 1\} \quad \forall i \in B \cup \{0\}, j \in V, k \in U, \tag{9}$$

$$0 \leqslant y_{ijk} \leqslant A_j \quad \forall i \in B, j \in V, k \in U, \tag{10}$$

where

$i\ (= 0, \ldots, I) \in B$	: set of berths
$j\ (= 1, \ldots, T) \in V$	: set of ships
$k\ (= 1, \ldots, T) \in U$	: set of service orders that are numbered in descending order from the last ship to be served
P_k	: subset of U such that $P_k = \{p \mid p > k \in U\}$
A_j	: preferred arrival time of ship j
C_j	: handling time spent by ship j
CT	: cylinder length (or the cycle time of the planning horizon)
G_j	: penalty cost (or priority) of ship j when it is not served.
Y_i	: start of the cylinder for berth i
x_{ijk}	: =1 if ship j is served as the k th ship at berth i, and =0 otherwise
y_{ijk}	: idle time of berth i between the departure of the $(k+1)$ th ship and the arrival of the k th ship when ship j is served as the k th ship

Note that $i = 0$ is a dummy berth for ships not to be served. The decision variables are x_{ijk} s, y_{ijk} s and Y_i s.

The objective (1) minimizes two evaluation criteria in different dimensions: time and cost. This study assumes G_j is a time equivalent for convenience, while it is worth to discuss that how the penalty cost is converted to time factor. Constraint set (2) ensures that every ship that is selected to be served must be moored at one of the berths in one of the service orders. Constraints (3) enforce that every berth serves up to one ship at any time. Constraints (4) assure that ships are served after their preferred arrival time. Constraints (5), (6) and (8) guarantee that berth-windows of served ships are located within the cylinder. Equalities (7) ensure that a couple of a mother ship and a feeder in a transshipment contract are both served or neither of them is served.

A formulation that is comprised of the ship service delay part of the objective $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\{(k-1)C_j - A_j\}x_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U} k y_{ijk}$ associated with constraints (2–4), was originally developed for the DBAP in Imai et al. (2001) which the reader is asked to see for its derivation, while its formulation is overviewed in Appendix C. Note that the DBAP model in Imai et al. assumes: (i) the objective is to minimize the TST (the sum of delay time for service and ship handling time), (ii) a ship's handling time depends on the allocated berth, (iii) the start time of berth availability, denoted by S_i, can be set as any arbitrary constant while this paper does not need it or can set it as zero due to the cyclic scheduling nature, and (iv) the ship service order is numbered in ascending order from the first ship to be served while this paper utilizes the reverse order scheme for a simpler objective function structure.

It seems that x_{0jk} for the ship selection part of the objective is redundant since the service order k is irrelevant for unserved ships. As described in Section 1, the solution procedure for [BTPS] is based on the subgradient method with Lagrangian relaxation. The lower bound can be easily obtained by the Lagrangian relaxation problem, which is equivalent to the classical assignment problem (AP), if we use x_{0jk} for the dummy berth just as using x_{ijk} for real berths.

Note that the berth idle time y_{ijk} can take any value. However, it is bounded to the arrival time of ship j as constraints (10), because it is at most A_j due to $S_i = 0$ being assumed in [BTPS].

It is also noteworthy that a mother ship and its associated feeders do not have to berth at the same time for transshipment. Ship loading and unloading is a very complicated job. So, for instance transshipment cargoes unloaded from a feeder are marshaled together with other cargoes (originating from that port) to be loaded onto a mother ship and then stacked at the yard for a while before being loaded on the mother ship arriving at a later time. Of course, another direction of transshipment is also possible. Therefore, both mother and feeder ships do not necessarily need to berth at the same time. Even if they are scheduled to berth simultaneously, a direct transshipment between them is usually not performed.

The same cylinder can be applied to all the berths for a terminal where the start of the cylinder may be set as zero. However, for more flexible and efficient scheduling we allow each berth to have a cylinder starting from a different time slot (i.e., a berth-dependent Y_i). The derivation of the cylinder-related constraints (5) and (6) will be discussed in the following section.

3.3. Derivation of the cylinder constraints

This section derives constraint sets (5) and (6). The constraints of (4), $\sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + y_{ijk} - A_j x_{ijk} \geq 0$, assure a ship is served after its arrival. By moving the third term of the left-hand side to the right, it turns to be

$\sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + y_{ijk} \geqslant A_j x_{ijk}$. The double summation of the first term in the left-hand side of the transformed constraint represents the total time associated with all of the ships served before the ship in concern. By using the concept similar to the associated total ship staying time in (4), the cylinder restriction can firstly be formulated as below:

$$\sum_{j\in V}\sum_{k\in U}(C_j x_{ijk} + y_{ijk}) \leqslant CT \qquad \forall i \in B,$$

The length CT in the above cylinder constraints implies that the planning horizon starts right from the time slot = zero. However, there is a time capacity loss if the handling of the first served ship starts some time later, as illustrated in Fig. 2(a). As shown in Fig. 2(b), as long as the BTPS can arrange the berth-windows within the constant length of the cylinder after taking into account the wrap-around effect, each berth may have different cylinder placement in the service planning horizon. In other words, the start of cylinder may be shifted afterward in time to place as many ships' berth-windows as possible within the cylinder.

Given the above observation, the shift regarding the start of ship berthing (or the start of the cylinder) for berth i is denoted by the variable Y_i, which serves a counterbalance for the time gap between the actual start of the cylinder and the presumed start of the planning horizon. With the introduction of this new notation, the realistic cylinder constraints can be formulated as follows:

$$\sum_{j\in V}\sum_{k\in U}(C_j x_{ijk} + y_{ijk}) - Y_i \leqslant CT \quad \forall i \in B, \tag{5}$$

Based on the above definition, Y_i is in fact the start time of the first served ship at berth i (i.e., for the ship with the largest k such that $x_{ijk} = 1$). Notice that though the BTPS arranges the berth-windows with respect to a fixed time span of CT (normally one week for container terminal service), berthing service continues with a rolling horizon. This means that during the time duration from zero to Y_i for the current horizon, berth i may not be idle and could be occupied by berthing ships for the previous cycle.

As mentioned above, the start time of each ship (for any service order) at the berth is related to the first and second terms of the left-hand side of constraints (4) as follows:

$$\sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + y_{ijk} \quad \forall i \in B, j \in V, k \in U,$$

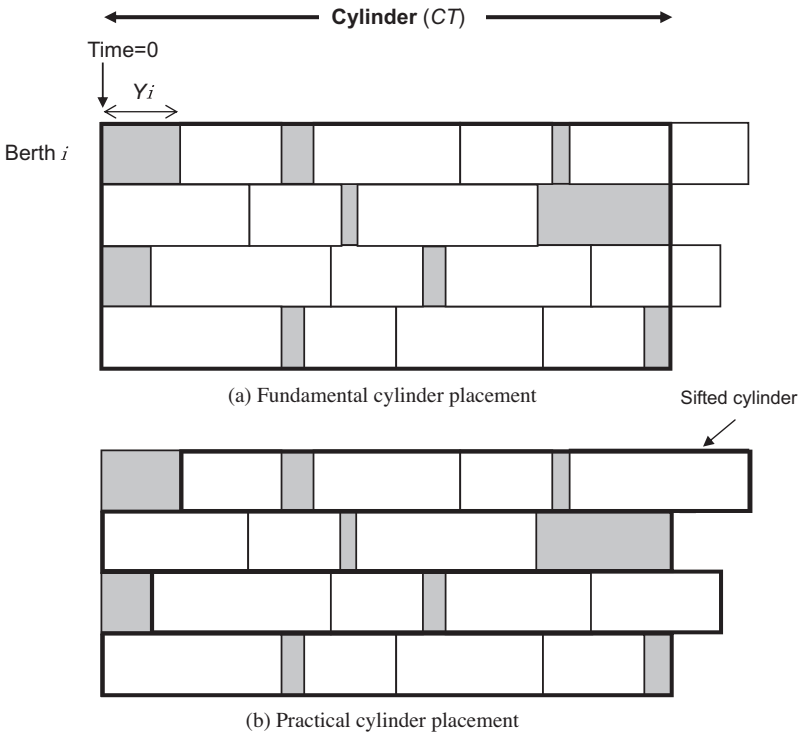


Fig. 2. Cylinder arrangement.

Thus, Y_i is linked to the minimum among all the ship and service order combinations at each berth as below:

$$Y_i \leqslant \sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + y_{ijk} \quad \forall i \in B, j \in V, k \in U,$$

For a service sequence at a berth, ships are supposed to occupy the service slots/orders with as small indices (denoted by k) as possible; otherwise, the objective function value (1) becomes larger, suggesting the solution is not optimal. Given this solution property, for the higher service slots k not occupied by any ship (e.g., the extreme case $k = T$) in the above constraints, Y_i would be incorrectly forced to be zero because of the null values of x_{ilm} and y_{ilm} in the double summation; since given $m > k$ where k is not occupied, the slots m are not occupied either. Thus, it is required to identify the largest service order (i.e., the largest k) with a ship assigned to the berth. For this, the formulation is changed as follows:

$$Y_i \leqslant \left(1 - \sum_{j\in V} x_{ijk}\right) M + \sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + \sum_{j\in V} y_{ijk} \quad \forall i \in B, k \in U,$$

where M is a large constant. The introduction of the large M in the above constraints makes Y_i become free for the service orders without any ship assigned.

It is worthy to note that Y_i is bounded by CT for any feasible solution as specified by constraints (8) because the first served ship at a berth can be served as the latest at CT. Consequently, M can be set as CT. As a result, the above constraints are re-written as follows:

$$Y_i \leqslant \left(1 - \sum_{j\in V} x_{ijk}\right) CT + \sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + \sum_{j\in V} y_{ijk} \quad \forall i \in B, k \in U, \tag{6}$$

4. Solution procedures

The authors developed an approximation algorithm for operational berth scheduling: DBAP, by using the subgradient procedure with Lagrangian relaxation problem. The subgradient procedure with Lagrangian relaxation is widely used for optimization problems. The BAP is an example for its application, as seen in Imai et al. (2001), Park and Kim (2002) and Monaco and Sammarra (2007). This study also employs the subgradient procedure for the BTPS.

4.1. Lagrangian relaxation

We introduce a Lagrangian relaxation problem to [BTPS]. Letting α_{ijk}, β_i, γ_{ik} and $\delta_{jj'}$ be Lagrangian multipliers for four constraint sets, (4–7), the Lagrangian relaxation is formulated as follows:

[RBTPS]

$$\begin{aligned}\text{Minimize } & \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\{(k-1)C_j - A_j\}x_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U} k y_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}\left\{\sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + y_{ijk} - A_j x_{ijk}\right\} \\ & - \sum_{i\in B}\beta_i\left\{CT + Y_i - \sum_{j\in V}\sum_{k\in U}(C_j x_{ijk} + y_{ijk})\right\} - \sum_{i\in B}\sum_{k\in U}\gamma_{ik}\left\{\left(1 - \sum_{j\in V} x_{ijk}\right)CT + \sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + \sum_{j\in V} y_{ijk} - Y_i\right\} \\ & - \sum_{j\in V}\sum_{j'(>j)\in V}\delta_{jj'}R_{jj'}\left\{\sum_{i\in B}\sum_{k\in U} x_{ij'k} - \sum_{i\in B}\sum_{k\in U} x_{ijk}\right\} \end{aligned} \tag{11}$$

subject to $(2), (3), (8-10)$

Objective (11) can be re-written as follows:

$$\begin{aligned} & \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\{(k-1)C_j - A_j\}x_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\sum_{l\in V}\sum_{m\in P_k}\alpha_{ijk}C_l x_{ilm} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}A_j x_{ijk} \\ & + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\beta_i C_j x_{ijk} + CT\sum_{i\in B}\sum_{k\in U}\sum_{j\in V}\gamma_{ik}x_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\sum_{m\in P_k}\gamma_{ik}C_l x_{ilm} \\ & + \sum_{j\in V}\sum_{j'(>j)\in V}\sum_{i\in B}\sum_{k\in U}\delta_{jj'}R_{jj'}x_{ijk} - \sum_{j\in V}\sum_{j'(>j)\in V}\sum_{i\in B}\sum_{k\in U}\delta_{jj'}R_{jj'}x_{ij'k} \\ & + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U} k y_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\sum_{l\in V}\sum_{m\in P_k}\alpha_{ijk}y_{ilm} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}y_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\beta_i y_{ijk} \\ & - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\sum_{m\in P_k}\gamma_{ik}y_{ilm} - \sum_{i\in B}\sum_{k\in U}\sum_{j\in V}\gamma_{ik}y_{ijk} - \sum_{i\in B}\beta_i Y_i + \sum_{i\in B}\sum_{k\in U}\gamma_{ik}Y_i - \sum_{i\in B}\beta_i CT - \sum_{i\in B}\sum_{k\in U}\gamma_{ik}CT \end{aligned}$$

In the above objective, $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\sum_{l\in V}\sum_{m\in P_k}\alpha_{ijk}C_l x_{ilm}$, $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\sum_{l\in V}\sum_{m\in P_k}\alpha_{ijk}y_{ilm}$, $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\sum_{m\in P_k}\gamma_{ik}C_l x_{ilm}$ and $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\sum_{m\in P_k}\gamma_{ik}y_{ilm}$ can be $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{T\}}\sum_{l\in V}\sum_{m\in P_k}\alpha_{ijk}C_l x_{ilm}$, $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{T\}}\sum_{l\in V}\sum_{m\in P_k}\alpha_{ijk}y_{ilm}$, $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{T\}}\sum_{m\in P_k}\gamma_{ik}C_l x_{ilm}$ and $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{T\}}\sum_{m\in P_k}\gamma_{ik}y_{ilm}$, respectively, because of $m > k$.

Due to the property shown in Appendix A, they are further transformed to

$$\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in V}\sum_{m<k}\alpha_{ilm}C_j x_{ijk},\ \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in V}\sum_{m<k}\alpha_{ilm}y_{ijk},\ \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{m<k}\gamma_{im}C_j x_{ijk} \quad \text{and} \quad \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{m<k}\gamma_{im}y_{ijk}$$

As a result, the objective function turns to be

$$\begin{aligned}
&\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\{(k-1)C_j - A_j\}x_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in V}\sum_{m<k}\alpha_{ilm}C_j x_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}A_j x_{ijk}\\
&\quad + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\beta_i C_j x_{ijk} + CT\sum_{i\in B}\sum_{k\in U}\sum_{j\in V}\gamma_{ik}x_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{m<k}\gamma_{im}C_j x_{ijk}\\
&\quad + \sum_{j\in V}\sum_{j'(>j)\in V}\sum_{i\in B}\sum_{k\in U}\delta_{jj'}R_{jj'}x_{ijk} - \sum_{j\in V}\sum_{j'(>j)\in V}\sum_{i\in B}\sum_{k\in U}\delta_{jj'}R_{jj'}x_{ij'k}\\
&\quad + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}ky_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in V}\sum_{m<k}\alpha_{ilm}y_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}y_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\beta_i y_{ijk}\\
&\quad - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{m<k}\gamma_{im}y_{ijk} - \sum_{i\in B}\sum_{k\in U}\sum_{j\in V}\gamma_{ik}y_{ijk} - \sum_{i\in B}\beta_i Y_i + \sum_{i\in B}\sum_{k\in U}\gamma_{ik}Y_i - \sum_{i\in B}\beta_i CT - \sum_{i\in B}\sum_{k\in U}\gamma_{ik}CT
\end{aligned}$$

Further,

$$\begin{aligned}
&= \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\{(k-1)C_j - A_j\}x_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in V}\sum_{m<k}\alpha_{ilm}C_j x_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}A_j x_{ijk}\\
&+ \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\beta_i C_j x_{ijk} + CT\sum_{i\in B}\sum_{k\in U}\sum_{j\in V}\gamma_{ik}x_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{m<k}\gamma_{im}C_j x_{ijk}\\
&+ \sum_{i\in B}\sum_{k\in U}\sum_{j\in V}\left(\sum_{l(>j)\in V}\delta_{jl}R_{jl} - \sum_{l(<j)\in V}\delta_{lj}R_{lj}\right)x_{ijk}\\
&+ \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}ky_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in V}\sum_{m<k}\alpha_{ilm}y_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}y_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\beta_i y_{ijk}\\
&- \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{m<k}\gamma_{im}y_{ijk} - \sum_{i\in B}\sum_{k\in U}\sum_{j\in V}\gamma_{ik}y_{ijk} - \sum_{i\in B}\beta_i Y_i + \sum_{i\in B}\sum_{k\in U}\gamma_{ik}Y_i - \sum_{i\in B}\beta_i CT - \sum_{i\in B}\sum_{k\in U}\gamma_{ik}CT.
\end{aligned}$$

Given a set of Lagrangian multipliers α_{ijk}, β_i, γ_{ik} and $\delta_{jj'}$, [RBTPS] is completely separable. That is, it can be restructured (without loss of solution accuracy) into the following three subproblems:

[SUB-1]

$$\begin{aligned}
\text{Minimize}\ &\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\{(k-1)C_j - A_j\}x_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in V}\sum_{m<k}\alpha_{ilm}C_j x_{ijk} + \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}A_j x_{ijk}\\
&+ \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\beta_i C_j x_{ijk} + CT\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\gamma_{ik}x_{ijk} - \sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{m<k}\gamma_{im}C_j x_{ijk}\\
&+ \sum_{j\in V}\sum_{j'(>j)\in V}\sum_{i\in B}\sum_{k\in U}\delta_{jj'}R_{jj'}x_{ijk} - \sum_{j\in V}\sum_{j'(>j)\in V}\sum_{i\in B}\sum_{k\in U}\delta_{jj'}R_{jj'}x_{ij'k} \qquad (12)
\end{aligned}$$

subject to (2), (3), (9)

[SUB-2]

$$\begin{aligned}
\text{Minimize}\ &\sum_{i\in B}\sum_{j\in V}(1+\beta_i - \alpha_{ij,1} - \gamma_{i,1})y_{ij,1} +\\
&\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\left(k + \beta_i - \alpha_{ijk} - \gamma_{ik} - \sum_{l\in V}\sum_{m<k}\alpha_{ilm} - \sum_{m<k}\gamma_{im}\right)y_{ijk} \qquad (13)
\end{aligned}$$

subject to (10)

[SUB-3]

$$\text{Minimize}\ \sum_{i\in B}\left(\beta_i - \sum_{k\in U}\gamma_{ik}\right)Y_i \qquad (14)$$

subject to (8)

[SUB-1] is optimally solved with the AP. In [SUB-2], $y_{ij,1} = 0$ if $1 + \beta_i - \alpha_{ij,1} - \gamma_{i,1} \geqslant 0$, and otherwise $y_{ijk} = A_j$. Further, $y_{ijk}(\forall k \in U \setminus \{1\}) = 0$ if $k + \beta_i - \alpha_{ijk} - \gamma_{ik} - \sum_{l\in V}\sum_{m<k}\alpha_{ilm} - \sum_{m<k}\gamma_{im} \geqslant 0$, and otherwise $y_{ijk} = A_j$. Note that $y_{ijk}(\forall k \in U) = 0$ if corresponding $x_{ijk} = 0$. Similarly, for [SUB-3], $Y_i = 0$ if $\beta_i - \sum_{k\in U}\gamma_{ik} \geqslant 0$, while $Y_i = CT$ if $\beta_i - \sum_{k\in U}\gamma_{ik} < 0$.

Subgradients to be used are $\phi_{\alpha,ijk} = \sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + y_{ijk} - A_j x_{ijk}$, $\phi_{\beta,i} = CT + Y_i - \sum_{j\in V}\sum_{k\in U}(C_j x_{ijk} + y_{ijk})$, $\phi_{\gamma,ik} = (1 - \sum_{j\in V} x_{ijk})CT + \sum_{l\in V}\sum_{m\in P_k}(C_l x_{ilm} + y_{ilm}) + \sum_{j\in V} y_{ijk} - Y_i$ and $\phi_{\delta,jj'} = R_{jj'}\{\sum_{i\in B}\sum_{k\in U} x_{ij'k} - \sum_{i\in B}\sum_{k\in U} x_{ijk}\}$. Lagrangian multipliers α_{ijk}, β_i, γ_{ik} and $\delta_{jj'}$ are updated by $\alpha^{n+1} = \alpha^n - t_n\phi^{(n)}_{\alpha,ijk}$, $\beta^{n+1} = \beta^n - t_n\phi^{(n)}_{\beta,i}$, $\gamma^{n+1} = \gamma^n - t_n\phi^{(n)}_{\gamma,ik}$ and $\delta^{n+1} = \delta^n - t_n\phi^{(n)}_{\delta,jj'}$ where the step size t_n is defined by $t_n = d_n(\overline{Z} - Z(\alpha^{(n)}, \beta^{(n)}, \gamma^{(n)}, \delta^{(n)})/\|\phi^{(n)}\|$.

4.2. Subgradient procedure

The outline of the subgradient optimization with the above relaxed problem is shown in Appendix B. This procedure is basically the same as the one used for the DBAP in Imai et al. (2001), which proposed three Lagrangian heuristics, SIMPLE, INDIVIDUAL and INTERACT, in order to derive a feasible solution to the DBAP by modifying the solution of its relaxed problem. The relaxed problem defined in their study is equivalent to [SUB-1] in this study. SIMPLE simply determines the start time of a ship service based on ship service orders for each berth, which are obtained by the relaxed problem (called SBAP), but with a constraint that ship services start after arrival. INDIVIDUAL is basically the same as SIMPLE, but it changes service orders of ships if there is a berth idle time between a specific pair of ships in a feasible solution by SIMPLE. INTERACT swaps ships across berths so as to fulfill the berth idle time for decreasing the total service time. See Imai et al. (2001) for the details of those heuristics.

This paper also develops another subgradient procedure for a better approximate solution by using the DBAP formulation structure. The outline of SBAP and DBAP is presented in Appendix C. In the SBAP all calling ships have arrived at port before S_i, which is the start time of berth availability (or planning horizon), and are ready to move to the terminal quay for mooring whenever their berth-windows are scheduled after S_i. Consequently, the SBAP is equivalent to the Single Machine Scheduling Problem (SMSP) where all the ships in the SBAP are already released jobs (or ready jobs to be processed by a machine in the SMSP). Thus, the ships are served in increasing order of their handling times because of the SMSP property (Pinedo, 2012). However, the DBAP forces them to be served after their arrival times that are not necessarily before S_i. Therefore, the SBAP solution may not lead to a better DBAP solution, since berth-windows in the SBAP solution may deviate far before or after the ship arrival times. In this context, we may expect a better DBAP solution if the SBAP solution generates berth-windows close to the ship arrival. The BTPS solution in terms of ship service delay obviously has the same feature of the DBAP despite the independent ship handling time from assigned berth in the BTPS unlike the DBAP.

In the following, a more exact discussion is made. In general, if a berth schedule for the SBAP has earlier arriving ships served earlier than late arriving ships, that schedule is likely to be a better solution for the DBAP due to the issue of waiting time. However, for two ships with $C_{ij'} = C_{ij^*}$ and $A_{j'} \leqslant A_{j^*}$, the SBAP solution $(x_{ij',k} = 1, x_{ij^*,k+1} = 1)$, which is less desirable for the DBAP, has the same objective function value as another solution $(x_{ij^*,k} = 1, x_{ij',k+1} = 1)$, which is more promising for the DBAP. Thus, in order to foster the possibility of generating the latter solution $(x_{ij^*,k} = 1, x_{ij',k+1} = 1)$, we introduce the modified SBAP.

Suppose there is only one berth existing; then, C_{ij} is replaced by C_j. Let $F_j(=A_j + C_j)$ be the earliest departure time of ship j, which is the departure time when the ship is served starting from A_j without any waiting. We then consider two ship arrangements for a pair of ships j' and j^* where $A_{j'} \leqslant A_{j^*}$. For an easier discussion, we let j' and j^* be replaced by 1 and 2. Arrangements of ship arrival for a pair of ships 1 and 2 where $A_1 \leqslant A_2$, are shown in Fig. 3. Both arrangements assume the two ships wish to begin their handling as soon as they arrive. In arrangement-A two ships overlap each other, while in arrangement-B they do not.

Then, in [DBAP] there is the following property:

Property 1. *The total service time for two ships can be minimized by serving the ships in ascending order of C_j when $A_2 \leqslant F_1$, and in ascending order of A_j when $A_2 > F_1$.*

Proof. We have the following two cases.

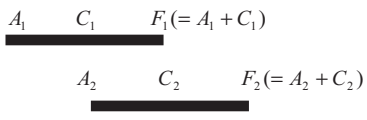
Case 1: When $A_1 \leqslant A_2 \leqslant F_1$ for arrangement-A, let TST-o be the total service time when ship 1 is served first and ship 2 next, and TST-d be the one when they are served oppositely. Then,

$$\text{TST-o} = C_1 + (F_1 - A_2) + C_2 = C_1 + (A_1 + C_1 - A_2) + C_2$$
$$\text{TST-d} = C_2 + (F_2 - A_1) + C_1 = C_2 + (A_2 + C_2 - A_1) + C_1$$

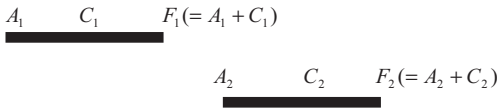
and

$$\text{TST-d} - \text{TST-o} = 2(A_2 - A_1) + C_2 - C_1 \geqslant C_2 - C_1.$$

If $C_2 \geqslant C_1$, ship 1 first and ship 2 next is a better solution; otherwise the opposite is better.Case 2: When $A_2 > F_1$ for arrangement-B,



(a) Arrangement-A



(b) Arrangement-B

Fig. 3. Ship arrangement.

$$\text{TST-o} = C_1 + C_2$$
$$\text{TST-d} = C_2 + (A_2 + C_2 - A_1) + C_1 = 2C_2 + C_1 + (A_2 - A_1)$$

and

$$\text{TST-d} - \text{TST-o} = C_2 + A_2 - A_1.$$

Because of $A_2 > A_1 + C_1$,

$$\text{TST-d} - \text{TST-o} = C_2 + A_2 - A_1 \geqslant A_1 + C_1 + C_2 - A_1 = C_1 + C_2 \geqslant 0.$$

Therefore, ship 1 first and ship 2 next is better regardless of $C_2 \geqslant C_1$ or $C_2 < C_1$.

From the above two cases, the proof is completed. □

Property 1 also indicates that serving the ships in ascending order of handling time is not always better than serving them in other ways. So, without loss of generality, Property 1 leads to that a better solution to [DBAP] does not necessarily exists where ships are served in ascending order of handling time.

Referring to [SBAP] formulated in Appendix C, we modify [SBAP] as follows with the assumption that the ship number, j, is given with decreasing order of arrival time A_j. That is, $A_{j+1} \leqslant A_j$.

[SBAPM]

$$\text{Minimize} \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\left(\frac{k}{j}C_{ij} + S_i - A_j\right)x_{ijk} \tag{15}$$

subject to $(A.2), (A.3), (A.4)$

We prove that a feasible solution from the solution to [SBAPM] is not worse than the one from [SBAP], where we assume $A_{j'} \leqslant A_{j^*}$ (or $j' > j^*$) and $k' > k^*$.

Lemma 1 [*SBAPM*]. *with a single berth defines a single machine scheduling problem with the objective of minimizing the total weighted completion time, i.e.,* $1\|\sum_j w_j G_j$, *if ship's number j corresponds to* w_j, *where* w_j *and* G_j *are weight and completion time of job j.*

Proof. Focusing on two ships j' and j^* where $j' > j^*$, we consider two schedules, S-o and S-d. Under S-o, they are served adjacently j' first and j^* next (i.e., j' and j^* are served as k' and k^* th ships, respectively), whereas under S-d j^* first and j' next. All the other ships remain in their positions both under S-o and S-d. Let SBAP-o and SBAPM-o be the objective function values of SBAP and SBAPM related to the two ships by $x_{i,j',k'} = 1$ and $x_{i,j^*,k^*} = 1$, and SBAP-d and SBAPM-d be the ones by $x_{i,j',k^*} = 1$ and $x_{i,j^*,k'} = 1$. Assuming the start of planning $S_i = 0$, then,

$$\text{SBAP-o} = k'C_{j'} - A_{j'} + k^*C_{j^*} - A_{j^*}$$
$$\text{SBAP-d} = k'C_{j^*} - A_{j^*} + k^*C_{j'} - A_{j'}$$

and

$$\text{SBAP-d} - \text{SBAP-o} = (k' - k^*)C_{j^*} - (k' - k^*)C_{j'} = (k' - k^*)(C_{j^*} - C_{j'}).$$

Since $k' - k^* > 0$, SBAP-o is smaller if $C_{j'} \leqslant C_{j^*}$, while SBAP-d is smaller if $C_{j'} \geqslant C_{j^*}$.

Similarly,

$$\text{SBAPM-o} = \frac{k'}{j'}C_{j'} - A_{j'} + \frac{k^*}{j^*}C_{j^*} - A_{j^*} \quad \text{and} \quad \text{SBAPM-d} = \frac{k'}{j^*}C_{j^*} - A_{j^*} + \frac{k^*}{j'}C_{j'} - A_{j'}.$$

These objectives are the same as SBAP-o and SBAP-d but with $\frac{k}{j}C_j$ as a substitution for kC_j. Therefore, SBAPM-o is smaller if $\frac{C_{j^*}}{j^*} \geqslant \frac{C_{j'}}{j'}$ (or $\frac{j^*}{C_{j^*}} \leqslant \frac{j'}{C_{j'}}$) and consequently in [SBAPM] ships' services are ordered in decreasing order of j/C_j. This feature defines the single machine scheduling problem. □

Lemma 2. *In [SBAPM], if the handling time C_{ij} is independent from berth, i,e, $C_{ij} \equiv C_j$ for all j, [SBAPM] with multiple berths defines parallel machine scheduling problem with the minimization of the total weighted completion time, $Pm\|\sum_j w_j G_j$.*

Proof. As the so-called *weighted shortest processing time first* (WSPT) rule for the single machine scheduling also minimizes the total weighted completion time for the parallel machine scheduling problem, the proof is completed. □

Lemma 3. *An [SBAPM] solution with an independent ship handling time from berth serves those ships earlier that arrive earlier when the ship handling time increases with increasing arrival time. Also there is the case that the solution serves ships earlier, when the ship handling time does not increase with increasing arrival time.*

Proof. From Lemma 1, SBAPM-o is smaller than SBAPM-d if $\frac{C_{j^*}}{j^*} \geqslant \frac{C_{j'}}{j'}$. In case of $C_{j'} = C_{j^*}$, this implies SBAPM-o is smaller because $j' > j^*$. In case of $C_{j'} < C_{j^*}$, it is also smaller. Next, we consider the case of $C_{j'} > C_{j^*}$. $\frac{C_{j^*}}{j^*} \geqslant \frac{C_{j'}}{j'}$ is transformed to $\frac{C_{j^*}}{C_{j'}} \geqslant \frac{j^*}{j'}$. Since $\frac{j^*}{j'}$ could take any value between 0 and 1, either SBAPM-o or SBAPM-d is better depending on situations when $C_{j'} > C_{j^*}$. As a result, the solution serves those ships earlier that arrive earlier when their handling times increase with their arrival time because of the examination with $C_{j'} = C_{j^*}$ and $C_{j'} < C_{j^*}$. Also, the solution may serve earlier those ships that arrive earlier when their handling times do not necessarily increase with their arrival time because of the discussion for $C_{j'} > C_{j^*}$. □

Lemma 4. *SBAP-o is less than SBAP-d only if the ship handling time increases with ship arrival time.*

Proof.

$$\text{SBAP-d} - \text{SBAP-o} = (k' - k^*)C_{j^*} - (k' - k^*)C_{j'} = (k' - k^*)(C_{j^*} - C_{j'}). \text{ Thus, only if } C_{j'} < C_{j^*},$$
$$\text{SBAP-d} - \text{SBAP-o} > 0 \text{ due to } k' > k^*.$$

□

Lemma 5. *An [SBAP] solution with an independent ship handling time from berth serves ships in ascending order of handling time.*

Proof. Suppose two ships served adjacently in decreasing order of handling time. If we have j' served first and j^* next where $A_{j'} \leqslant A_{j^*}$ and $C_{j'} \geqslant C_{j^*}$, the objective function value can be reduced by serving them oppositely because of Lemma 4. Next, we consider the case that j' is served first and j^* next where $A_{j'} > A_{j^*}$ and $C_{j'} \geqslant C_{j^*}$. By switching the ship numbers, we have j^* served first and j' next where $A_{j'} < A_{j^*}$ and $C_{j'} \leqslant C_{j^*}$. The objective function value for this service order corresponds to SBAP-d. By Lemma 4, the objective function value by serving the ships oppositely results in SBAP-o, which is less than SBAP-d. Therefore, ships should be served in ascending order of handling time in SBAP. □

Theorem 1. *[SBAPM] does not lead to a worse feasible solution than [SBAP].*

Proof. From Property 1, a better solution of [DBAP] does not necessarily serve ships in ascending order of handling time. From Lemma 3 an [SBAPM] solution may serve earlier those ships with earlier arrival regardless of handling time. On the other hand, from Lemma 5 an [SBAP] solution as a relaxed problem to [DBAP] serves ships in increasing order of handling time. Consequently, [SBAPM] does not provide a worse feasible solution for [DBAP] than [SBAP] does. □

The minimization of the total delay time for the BTPS is equivalent to the minimization of the total service time for the DBAP because of the independent ship handling time from berthing location in the BTPS. Therefore, taking into account the above property, the relaxed problem is modified by substituting $\frac{C_j}{j}$ for C_j. [SUB-1] part of the resulting relaxed problem is as follows, whilst [SUB-2] and [SUB-3] remains the same.

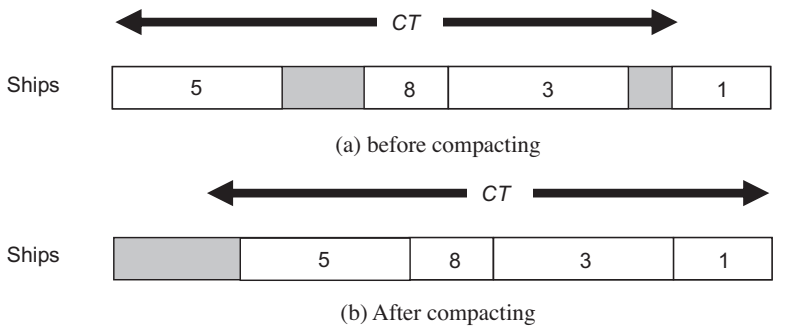


Fig. 4. Compacting ship services in the cylinder.

[SUB-1′]

$$\begin{aligned}\text{Minimize } & \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\left\{(k-1)\frac{C_j}{j}-A_j\right\}x_{ijk}-\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in V}\sum_{m<k}\alpha_{ilm}\frac{C_j}{j}x_{ijk}\\ &+\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\alpha_{ijk}A_jx_{ijk}+\sum_{i\in B}\sum_{j\in V}\sum_{k\in U}\beta_i\frac{C_j}{j}x_{ijk}+CT\sum_{i\in B}\sum_{k\in U}\sum_{j\in V}\gamma_{ik}x_{ijk}\\ &-\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{m<k}\gamma_{im}\frac{C_j}{j}x_{ijk}\\ &+\sum_{j\in V}\sum_{j'(>j)\in V}\sum_{i\in B}\sum_{k\in U}\delta_{jj'}R_{jj'}x_{ijk}-\sum_{j\in V}\sum_{j'(>j)\in V}\sum_{i\in B}\sum_{k\in U}\delta_{jj'}R_{jj'}x_{ij'k}\end{aligned}\tag{16}$$

subject to (2), (3), (9)

Notice that this relaxed problem does not provide a lower bound for [BTPS] because [SUB-1′] is constructed with $\frac{C_j}{j}$ but not with C_j; whilst it provides a basis, with which a feasible solution to [BTPS] is constructed.

The modified subgradient method with this relaxed problem is outlined in Appendix D.

4.3. Lagrangian heuristic for finding a feasible solution

In Step 3 of the subgradient procedure shown in Appendix B and in Step 3–2 of the modified one shown in Appendix D, a feasible solution is found from an optimal solution to the Lagrangian relaxation problem of [BTPS] by a Lagrangian heuristic. This section describes the outline of the Lagrangian heuristic.

In the subgradient procedure for [DBAP] in Imai et al. (2001), a feasible solution is found by one of three different processes: SIMPLE, INDIVIDUAL and INTERACT. The Lagrangian heuristic first performs one of those processes for [DBAP]. If the resulting solution satisfies the relaxed constraints (5), i.e., the cylinder constraint, the Lagrangian heuristic terminates. If it does not, the Lagrangian heuristic continues the process that shifts some ships from their berth without cylinder satisfaction to other berths, so that all the berths satisfy the cylinder constraint. If some berths are still not satisfactory despite of this shifting, some ships are dropped from service, so that berth-windows for all ships to be served are placed within the cylinder.

The Lagrangian heuristic employs the similar procedure to the one for the Bin-Packing Problem, which packs items of various sizes into a set of bins while minimizing the number of bins used. Many heuristics have been exploited for this problem since the early 1970s. See some details in Bramel and Simchi-Levi (1997). We employ one of the popular Bin-Packing heuristics, First-Fit Decreasing, which first sorts the items in non-increasing order of their size and place them in the lowest indexed bin at the moment of packing whose current content does not exceed the bin capacity. Items and bins correspond to ships and berths in our problem background.

A more formal description of the heuristic follows:

[LH]

Step 1. In the solution to [SUB-1] (or [SUB-1′]), if a ship selected not to be served is connected with other ships with $R_{jj'}=1$, those ships are discarded too. Perform a DBAP's Lagrangian heuristic (SIMPLE, INDIVIDUAL or INTERACT) with those ships to be served. If all berths satisfy the cylinder constraint, then STOP.

Step 2. Setting the end time of the cylinder as the k(=1) th ship's completion time of service, set services of all the preceding ships in time without berth idle between any adjacent two ships, as shown in Fig. 4. If all berths serve all ships assigned to them within the cylinder, go to Step 8.

Step 3. Mark those berths without cylinder satisfaction and refer to them as violating berths.

Step 4. For all violating berths, remove ships in ascending order of its handling time C_j until the total ship handling time is no more than CT. Register them on a ship list.

Step 5. Arrange ships in non-increasing order of their handling times on the ship list.
Step 6. Examine a ship from the ship list. Find a berth, from berth 1, with idle berth time within *CT* that lets the ship stay. If such a berth is found, place the ship at the time closest to its arrival time in the berth and delete it from the ship list. Otherwise, the ship is registered as an un-served ship. If the un-served ship is associated with other ships with $R_{jj'} = 1$ either in the ship list or being as placed in the berths, those connected ships are also registered as un-served ones and deleted from the list and the berths. Update the idle berth time for berths if a ship is inserted in a berths and/or ships are deleted from berths in the above process.
Step 7. Repeat Step 6 until all the ships in the ship list are examined.
Step 8. Arrange ships' services in respective berths so that they start their services as soon as arrival time within *CT*.

For an arrangement of ship services for each berth in Step 8 of [LH], the following procedure is used:

Step 8–1. The end time of the cylinder, *CTend*, is set as the k(=1) th ship's completion time of service. The start time of the cylinder, *CTstart*, is set as $CTstart = CTend - CT$.
Step 8–2. From the last ship to the first ship in service order k for the berth, compute the start time (*ST*) and completion time (*FN*) of ship's service as follows:
If ship j is the last ship, $ST = \text{Max}\,(CTstart, A_j)$, $FN = ST + C_j$; otherwise,

$ST = \text{Max}$ (*FN* of the previous ship, A_j), $FN = ST + C_j$.

The heuristic initially deals with all the served ships that are selected by solving [SUB-1] or [SUB-1′]. However, some of those ships may be discarded in the heuristic since the solution to [SUB-1] or [SUB-1′] does not satisfy the cylinder constraints (5).

5. Numerical experiments

5.1. Outline of the experiments

The solving algorithm for the BTPS is constructed with theoretical implications of the DBAP algorithm in Imai et al. (2001). From this viewpoint, we first perform preliminary experiments to examine how the modified subgradient procedure works by using the DBAP instances. Subsequently, we perform a wide variety of experiments for the BTPS. All the solution procedures for the DBAP and BTPS are coded in "C" language on a Panasonic Let's note CF-B11 computer.

5.2. Preliminary experiments for the modified subgradient procedure

All computational instances assume four berths with 100 calling ships. We prepared three patterns of interval of the ship arrival time, A_j, being generated by exponential random variable with an average of 1, 5 and 8 h. The cargo handling time of a ship, C_{ij}, varies on its potential berthing locations, but it was generated based on uniform random variables with different average times of 4 and 12 h. Also, we prepared three different fluctuations between the maximum and minimum amounts of the cargo handling time of a ship: 0%, 100% and 200% of the average time. Totally 18 different calling ship scenarios were prepared with those data factors. For each computation instance of the 18 ones, we set three different start times of the planning horizon, S_i, where all berths have the same value of S_i. The earliest time is S_i equivalent to the first quarter of the 100 ships (S1), the next is the half (S2) and the third is the three quarters (S3). The combination of three different times and 18 scenarios leads to 54 problem instances in total.

We used three different weighted handling times for the modified subgradient procedure: C_{ij} (=original), C_{ij}/j and $C_{ij}/\log(j)$. As we utilize three Lagrangian heuristics of SIMPLE, INDIVIDUAL and INTERACT, we totally have nine algorithms to be compared. For every single instance, the procedure was terminated with 200 iterations.

Table 1 shows the average values (over the 18 instances) of UB (upper bound), LB (lower bound), GAP (=(UB-LB)/LB * 100) and CPU time, and the total values (of the 18 instances) of count of the best solutions among other methods for different S_i s. The bottom line in Table 1 shows the grand average UB and count of optimal solutions over cases S1 to S3. A value for each line in bold italic is the best among the nine solution methods, while figures in bold non-italic are the second best ones. Note that the GAP is not computed with average UB and LB, but it is the average over GAPs for all individual problem instances.

First of all, the solution quality in terms of GAP is improving with larger S_i. Typically, in case of S1, LB is negative (LBs of some individual problem instances are positive); this implies that the DBAP with an early start time of planning horizon is hard to solve. The GAP is not shown for S1 instances because a correct indication of solution quality is not provided by the negative LB. For S2, one problem instance out of the 18 has a negative LB, so its GAP is an average of other 17 instances. This trend of worse solution quality with smaller S_i is also reported in Imai et al. (2001). The reason why the GAP improves with larger S_i is that the DBAP with larger S_i reduces to the SBAP, which easily finds an optimal solution.

Generally speaking, SIMPLE is the worst among the three Lagrangian heuristics. Both INDIVIDUAL and INTERACT result in almost same performance. For comparisons among the modified subgradient procedure with different weighted handling time, C_{ij}/j and $C_{ij}/\log(j)$ are very effective for SIMPLE in terms of UB and GAP. They are also promising for INDIVIDUAL, but

Table 1
Average values by S_i for DBAP.

		SIMPLE			INDIVIDUAL			INTERACT		
		C_{ij}	C_{ij}/j	$C_{ij}/\log(j)$	C_{ij}	C_{ij}/j	$C_{ij}/\log(j)$	C_{ij}	C_{ij}/j	$C_{ij}/\log(j)$
S1	UB (h)	16915.8	5666.1	8997.8	4007.9	3369.9	3341.5	**3218.9**	3291.7	***3213.3***
	LB (h)	−5853.0	−5845.7	−5851.3	−5845.3	−5845.4	−5845.2	***−5844.9***	**−5845.1**	−5845.3
	GAP (%)	–	–	–	–	–	–	–	–	–
	# of b. sol.	0	**5**	**5**	0	**5**	**5**	0	***10***	***10***
	CPU (s)	**30.7**	95.3	86.6	***30.1***	94.3	98.8	31.5	94.7	85.9
S2	UB (h)	18661.8	9682.4	12605.8	8908.8	8639.3	8618.9	***8606.9***	8623.3	**8612.4**
	LB (h)	5004.5	5006.1	5005.5	5005.8	5006.3	5005.9	**5006.2**	5006.1	***5007.4***
	GAP (%)*	1289.1	517.4	815.5	405.4	396.1	397.2	396.8	***395.5***	**396.0**
	# of b. sol.	0	6	6	0	7	7	0	9	***11***
	CPU(s)	28.5	84.2	76.6	**28.2**	83.5	87.9	***25.9***	83.6	76.0
S3	UB (h)	21817.6	17180.5	18151.9	16752.3	16763.1	***16642.2***	16726.6	16803.4	**16665.2**
	LB (h)	15664.0	15665.2	15665.2	***15666.0***	15665.4	**15665.8**	**15665.8**	15665.3	15665.7
	GAP (%)	39.1	10.3	16.6	7.5	7.7	***6.5***	7.4	8.2	**6.8**
	# of b. sol.	1	6	6	1	7	**11**	1	6	***12***
	CPU (s)	**25.1**	75.3	66.5	***22.0***	74.9	79.7	***22.0***	74.9	65.9
Ave	UB (h)	19131.7	10843.0	13251.9	9889.7	9590.8	9534.2	9517.5	**9572.8**	***9497.0***
	# of best sol.	1	17	17	1	19	23	1	**25**	***33***

UB (h): upper bound.
LB (h): lower bound.
GAP (%) = (UB − LB)/LB ∗ 100.
Figure in bold italic: the best among the nine algorithms for a specific problem case with 18 instances.
Figure in bold non-italic: the second best among the nine algorithms for a specific problem case with 18 instances.
* One out of 18 instances for S2 has a negative LB. So, GAP is average of 17 instances.

their superiority is quite minor compared to SIMPLE cases. For S3, they do not outperform. For S1, INTERACT with $C_{ij}/\log(j)$ generates the best in an average UB value among others, but the ones with C_{ij}/j and $C_{ij}/\log(j)$ provide the most count of the best solutions. For S2, INTERACT with C_{ij} is the best in UB while INTERACT with $C_{ij}/\log(j)$ is the best in the most count of the best solutions. For S3, INDIVIDUAL with $C_{ij}/\log(j)$ is best in UB whilst INTERACT with $C_{ij}/\log(j)$ provides the most count. Comparing C_{ij}/j and $C_{ij}/\log(j)$, the former is better in UB for SIMPLE, while the latter slightly outperforms the former for INDIVIDUAL and INTERACT. However, in terms of the number of best solutions, $C_{ij}/\log(j)$ is the best for all three start times. As for computation time, procedures with C_{ij} is the best.

As mentioned above, both INDIVIDUAL and INTERACT result in almost same performance. However, regarding UB, INTERACT with $C_{ij}/\log(j)$ is the best for S1 and the second best for S2 and S3. Furthermore, it outperforms in the most count. If we take into account CPU time, the procedures with C_{ij} would be the most preferable. Looking at the bottom line in Table 1 that shows the average UB and count of optimal solutions over S1 to S3, INTERACT with $C_{ij}/\log(j)$ provides the best average UB as well as optimal count.

Since the subgradient procedures with C_{ij} that were developed in Imai et al. (2001) are less complicated than the modified ones implemented in this paper, the CPU times for the former, with less solution quality, are much shorter than the latter. So, we run the former with more iterations that could result in the almost same CPU times in order to examine the solution quality of the former in almost same CPU time as the one the modified subgradient procedures spend. To do so, we run the three subgradient procedures of C_{ij} with 800 iterations. Table 2 shows comparisons of solution quality by those

Table 2
C_{ij} results with more iterations.

		SIMPLE		INDIVIDUAL		INTERACT	
	# Of iterations	200	800	200	800	200	800
S1	UB (h)	16915.8	15440.3	4007.9	3948.2	3218.9	3183.7
	LB (h)	−5809.1	−5809.1	−5845.3	−5809.6	−5844.9	−5808.9
	GAP (%)	–	–	–	–	–	–
	CPU (s)	30.7	119.6	30.1	116.6	30.5	118.0
S2	UB (h)	18661.8	17527.1	8908.8	8886.2	8606.9	8584.8
	LB (h)	5004.5	5022.3	5005.8	5021.9	5006.2	5021.9
	GAP (%)	1289.1	1073.9	405.4	378.7	396.8	371.4
	CPU s)	28.5	109.8	28.2	107.2	25.9	98.5
S3	UB (h)	21817.6	21456.1	16752.3	16719.4	16726.6	16705.5
	LB (h)	15664.0	15667.9	15666.0	15667.7	15665.8	15667.7
	GAP (%)	39.1	36.6	7.5	7.3	7.4	7.2
	CPU (s)	25.1	96.7	22.0	82.4	22.0	82.5

Table 3
BTPS results.

T	Interval (h)		C_j	C_j/j	$C_{ij}/\log(j)$
SIMPLE					
50	2.8	OBJ	29460.6	27779.7	28517.4
		TDT (h)	2793.9	1113.0	1850.7
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
		CPU (s)	**5.4**	***5.3***	8.6
55	2.7	OBJ	111134.6	123321.8	123115.3
		TDT (h)	3356.9	1470.0	2374.5
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
		CPU time (s)	7.7	***7.5***	12.4
60	2.5	OBJ	319374.4	329555.4	331353.7
		TDT (h)	3448.5	1777.6	2835.2
		# of best sol.	0	3	0
		# of opt sol.	0	0	0
		CPU (s)	***10.5***	**11.5**	17.9
65	2.3	OBJ	475158.8	486282.4	488601.6
		TDT (h)	4019.3	1838.0	3046.1
		# of best sol.	2	3	0
		# of opt sol.	0	0	0
		CPU (s)	**13.6**	14.7	23.5
100	1.4	OBJ	59075.2	55928.9	65733.3
		TDT (h)	4462.7	2580.9	3511.1
		# of best sol.	0	1	1
		# of opt sol.	0	0	0
		CPU (s)	153.9	***132.0***	297.8
Average		OBJ	198679.0	204573.6	207464.3
		TDT (h)	3616.3	1755.9	2723.5
		# of best sol.	0.4	1.4	0.2
		# of opt sol.	0.0	0.0	0.0
		CPU (s)	38.2	***34.2***	72.1
INDIVIDUAL					
50	2.8	OBJ	28480.6	27165.9	27723.7
		TDT (h)	1813.9	499.3	1057.1
		# of best sol.	0	12	9
		# of opt sol.	0	3	3
		CPU (s)	***5.3***	14.4	15.8
55	2.7	OBJ	**109997.8**	122281.7	123003.9
		TDT (h)	2220.0	**800.3**	1522.4
		# of best sol.	0	6	6
		# of opt sol.	0	3	3
		CPU (s)	**7.6**	21.3	23.7
60	2.5	OBJ	***316879.8***	344567.3	345566.1
		TDT (h)	2065.0	**863.6**	1492.1
		# of best sol.	3	7	6
		# of opt sol.	0	2	2
		CPU (s)	***10.2***	32.0	34.5
65	2.3	OBJ	***473390.1***	488588.3	504081.3
		TDT (h)	2279.0	**810.6**	1118.3
		# of best sol.	1	4	5
		# of opt sol.	0	0	0
		CPU (s)	***13.5***	42.7	46.3
100	1.4	OBJ	***55869.6***	67338.1	76454.6
		TDT (h)	2536.3	**671.4**	899.0
		# of best sol.	0	8	6
		# of opt sol.	0	3	3
		CPU (s)	**150.6**	535.6	644.7
Average		OBJ	***196923.6***	209988.3	215365.9
		TDT (h)	2182.8	**729.0**	1217.8
		# of best sol.	1.4	7.4	6.4
		# of opt sol.	0.0	2.2	2.2
		CPU (s)	**37.4**	129.2	153.0
INTERACT					
50	2.8	OBJ	28473.0	***27015.7***	**27159.2**
		TDT (h)	1806.3	***349.0***	**492.5**
		# of best sol.	0	***23***	**16**
		# of opt sol.	0	***9***	***9***
		CPU (s)	**5.4**	26.6	28.7

(*continued on next page*)

Table 3 (*continued*)

T	Interval (h)		C_j	C_j/j	$C_{ij}/\log(j)$
55	2.7	OBJ	***109970.2***	121994.4	122292.7
		TDT (h)	2192.4	***512.9***	811.3
		# of best sol.	1	***22***	**13**
		# of opt sol.	0	***9***	***9***
		CPU (s)	7.7	36.3	39.2
60	2.5	OBJ	**316953.1**	344401.1	345019.4
		TDT (h)	2138.3	***697.4***	945.3
		# of best sol.	3	***19***	**12**
		# of opt sol.	0	***8***	***8***
		CPU (s)	10.5	50.0	53.8
65	2.3	OBJ	**473574.6**	488135.6	495005.6
		TDT (h)	2463.5	***728.2***	931.6
		# of best sol.	4	***18***	**12**
		# of opt sol.	0	***2***	***2***
		CPU (s)	**13.6**	65.5	69.5
100	1.4	OBJ	**55915.3**	62776.3	62942.9
		TDT (h)	2581.9	***554.1***	720.7
		# of best sol.	0	***23***	**15**
		# of opt sol.	0	***9***	***9***
		CPU (s)	151.7	596.2	703.6
Average		OBJ	**196977.2**	208864.6	210484.0
		TDT (h)	2236.5	***568.3***	780.3
		# of best sol.	0.0	***21.0***	**13.6**
		# of opt sol.	0.0	***7.4***	***7.4***
		CPU (s)	37.8	154.9	178.9

Figure in bold italic: the best among the nine algorithms for a specific problem case with 18 instances.
Figure in bold non-italic: the second best among the nine algorithms for a specific problem case with 18 instances.

procedures with 200 and 800 iterations. While the objective function values slightly improve (roughly decreases by 1–10%) with more iterations, they are not as good as the ones by the modified subgradient procedures. The CPU times for 800 iterations are almost same as the ones for the modified subgradient procedures with 200 iterations.

As a result from the above discussion, the modified subgradient methods with weighted handling times are effective for the three Lagrangian heuristics from the viewpoint of the UB and the number of best solutions. In particular, INTERACT with $C_{ij}/\log(j)$ is the most promising and can be expected to perform well for the BTPS.

5.3. BTPS experiments

Next, we perform numerical experiments for the BTPS. Providing four berths, we have four scenarios that serve the number of calling ships (T) ranging from 50 to 65. Also, to examine the BTPS procedures with large problem instances, we prepare instances of 100 ships with eight berths. Thus, we have five scenarios in total. Each of all the problem instances has ten mother-feeder connections, where two of the ships in each connection are served or neither of them is.

The tactical BTP (BTPT) is a hard problem to be solved because of the cylinder length. It is likely that we do not find a feasible solution of the BTPT with an enormous number of calling ships. The BTPS overcomes this drawback by eliminating less important ships from service. However, the BTPS should be examined in two cases: one when all ships are served and the other when all of them are not due to the cylinder. Therefore, we carefully design problem instances. To do so, for each scenario the given ships are spread during the cylinder length (CT = 150 h, which is equivalent to almost one week) in terms of arrival time whose interval follows an exponential distribution. For the arrival time we created three sets with different seeds for random numbers. The ship handling time, C_j, was generated based on uniform random variables with different average times of 4, 8 and 12 h. Also, we prepared three different fluctuations between the maximum and minimum amounts of the cargo handling time of a ship: 0%, 100% and 200% of the average time. In total, we have 27 problem instances for a scenario with a specific number of calling ships.

For BTPS experiments, penalty costs are carefully designed; otherwise, experimental results may be meaningless. The BTPS minimizes the total of delay time and penalty (in terms of time). If the value of ship penalty is not so large compared to the ship handling time, a great number of ships may not be selected to be served. More concretely speaking, if much delay is expected by serving lots of ships while the berthing capacity is not large enough to cover all of them, the BTPS model refuses many of them to reduce the delay time. To avoid such an unrealistic solution, the penalty is set much larger value than the ship handling time. In our experiments, the penalty value is 10000 times as much as the handling time.

Like DBAP experiments, we have nine solution procedures with a combination of the three heuristics SIMPLE, INDIVIDUAL and INTERACT, and the three weighted handling times C_j (original), C_j/j and $C_j/\log(j)$. All the procedures run with 200 iterations. Also, the procedures with C_j run with 800 iterations like the preliminary experiments with the DBAP instances; however, the results of these runs are almost same as the runs with 200 iterations. So, these results are not reported in the following analyses.

Table 4
BTPS results for BTPT-feasible cases.

T	Interval (h)		C_j	C_j/j	$C_j/\log(j)$
SIMPLE					
50	2.8	OBJ	2820.0	1159.7	1977.5
		TDT (h)	2820.0	1159.7	1977.5
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
55	2.7	OBJ	3147.3	1397.0	2312.4
		TDT (h)	3147.3	1397.0	2312.4
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
60	2.5	OBJ	3421.8	1202.2	2381.2
		TDT (h)	3421.8	1202.2	2381.2
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
65	2.3	OBJ	4019.3	1321.1	2516.3
		TDT (h)	4019.3	1321.1	2516.3
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
100	1.4	OBJ	4998.9	2601.4	3690.8
		TDT (h)	4998.9	2601.4	3690.8
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
Average		OBJ	3681.5	1536.3	2575.7
		TDT (h)	3681.5	1536.3	2575.7
		# of best sol.	0.0	0.0	0.0
		# of opt sol.	0.0	0.0	0.0
INDIVIDUAL					
50	2.8	OBJ	1758.8	495.3	1122.9
		TDT (h)	1758.8	495.3	1122.9
		# of best sol.	0	9	6
		# of opt sol.	0	3	3
55	2.7	OBJ	2078.0	751.7	1411.7
		TDT (h)	2078.0	751.7	1411.7
		# of best sol.	0	6	5
		# of opt sol.	0	3	3
60	2.5	OBJ	1742.0	298.9	879.4
		TDT (h)	1742.0	298.9	879.4
		# of best sol.	0	6	6
		# of opt sol.	0	2	2
65	2.3	OBJ	1548.5	229.3	511.5
		TDT (h)	1548.5	229.3	511.5
		# of best sol.	0	4	5
		# of opt sol.	0	0	0
100	1.4	OBJ	2344.1	**653.0**	909.8
		TDT (h)	2344.1	**653.0**	909.8
		# of best sol.	0	8	6
		# of opt sol.	0	3	3
Average		OBJ	1894.3	485.7	967.1
		TDT (h)	1894.3	485.7	967.1
		# of best sol.	0.0	6.6	5.6
		# of opt sol.	0.0	2.2	2.2
INTERACT					
50	2.8	OBJ	1785.1	***326.3***	**487.8**
		TDT (h)	1785.1	***326.3***	**487.8**
		# of best sol.	0	***20***	**13**
		# of opt sol.	0	***9***	***9***
55	2.7	OBJ	2097.7	***408.8***	**533.3**
		TDT (h)	2097.7	***408.8***	**533.3**
		# of best sol.	0	***22***	**12**
		# of opt sol.	0	***9***	***9***
60	2.5	OBJ	1772.4	***49.6***	**69.7**
		TDT (h)	1772.4	***49.6***	**69.7**
		# of best sol.	0	***18***	**12**
		# of opt sol.	0	***8***	***8***
65	2.3	OBJ	1645.1	***106.3***	**222.4**
		TDT (h)	1645.1	***106.3***	**222.4**
		# of best sol.	0	***18***	**12**
		# of opt sol.	0	***2***	***2***

(*continued on next page*)

Table 4 (*continued*)

T	Interval (h)		C_j	C_j/j	$C_j/\log(j)$
100	1.4	OBJ	2395.7	***514.1***	705.3
		TDT (h)	2395.7	***514.1***	705.3
		# of best sol.	0	***23***	**14**
		# of opt sol.	0	***9***	***9***
Average		OBJ	1939.2	***281.0***	**403.7**
		TDT (h)	1939.2	***281.0***	**403.7**
		# of best sol.	0.0	***20.2***	**12.6**
		# of opt sol.	0.0	**7.4**	**7.4**

Figure in bold italic: the best among the nine algorithms for a specific problem case with 18 instances.
Figure in bold non-italic: the second best among the nine algorithms for a specific problem case with 18 instances.

Table 5
BTPS results for BTPT-infeasible cases.

T	Interval (h)		C_j	C_j/j	$C_j/\log(j)$
SIMPLE					
50	2.8	OBJ	242585.0	**240739.7**	240836.3
		TDT (h)	2585.0	**739.7**	836.3
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
55	2.7	OBJ	586279.0	659791.2	654648.0
		TDT (h)	4279.0	1791.2	2648.0
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
60	2.5	OBJ	951279.8	986261.9	989298.8
		TDT (h)	3502.0	2928.6	3743.2
		# of best sol.	0	***3***	0
		# of opt sol.	0	0	0
65	2.3	OBJ	1417437.9	1456205.0	1460772.2
		TDT (h)	4104.6	2871.7	4105.6
		# of best sol.	2	3	0
		# of opt sol.	0	0	0
100	1.4	OBJ	484408.0	482548.7	562073.3
		TDT (h)	4408.0	2548.7	2073.3
		# of best sol.	0	1	1
		# of opt sol.	0	0	0
Average		OBJ	736397.9	765109.3	781525.7
		TDT (h)	3775.7	2176.0	2681.3
		# of best sol.	0.4	**1.4**	0.2
		# of opt sol.	0.0	0.0	0.0
INDIVIDUAL					
50	2.8	OBJ	242255.0	***240530.7***	***240530.7***
		TDT (h)	2255.0	***530.7***	***530.7***
		# of best sol.	0	***3***	***3***
		# of opt sol.	0	0	0
55	2.7	OBJ	**584845.2**	657014.0	658009.6
		TDT (h)	2845.2	***1014.0***	2009.6
		# of best sol.	***3***	0	1
		# of opt sol.	0	0	0
60	2.5	OBJ	**947155.4**	1033104.0	1034939.6
		TDT (h)	2711.0	***1992.9***	2717.3
		# of best sol.	***3***	1	0
		# of opt sol.	0	0	0
65	2.3	OBJ	***1417073.2***	1465306.3	1511220.9
		TDT (h)	3739.9	**1973.0**	2332.0
		# of best sol.	1	0	0
		# of opt sol.	0	0	0
100	1.4	OBJ	**484073.7**	600818.3	680813.0
		TDT(h)	4073.7	818.3	**813.0**
		# of best sol.	0	0	0
		# of opt sol.	0	0	0
Average		OBJ	***735080.5***	799354.7	825102.7
		TDT (h)	3125.0	***1265.8***	1680.5
		# of best sol.	**1.4**	0.8	0.8
		# of opt sol.	0.0	0.0	0.0

Table 5 (*continued*)

T	Interval (h)		C_j	C_j/j	$C_j/\log(j)$
INTERACT					
50	2.8	OBJ	241976.0	***240530.7***	***240530.7***
		TDT (h)	1976.0	***530.7***	***530.7***
		# of best sol.	0	***3***	***3***
		# of opt sol.	0	0	0
55	2.7	OBJ	***584609.2***	656971.0	658034.4
		TDT (h)	2609.2	**971.0**	2034.4
		# of best sol.	1	0	1
		# of opt sol.	0	0	0
60	2.5	OBJ	947314.4	1033104.0	1034918.7
		TDT (h)	2870.0	***1992.9***	2696.4
		# of best sol.	***3***	0	0
		# of opt sol.	0	0	0
65	2.3	OBJ	**1417433.7**	1464194.3	1484572.0
		TDT (h)	4100.3	***1972.1***	2349.8
		# of best sol.	0	18	12
		# of opt sol.	0	0	0
100	1.4	OBJ	***484071.7***	560874.0	560844.3
		TDT (h)	4071.7	874.0	***844.3***
		# of best sol.	0	2	0
		# of opt sol.	0	0	0
Average		OBJ	**735081.0**	791134.8	795780.0
		TDT (h)	3125.4	**12368.1**	1691.1
		# of best sol.	***1.6***	0.8	1.0
		# of opt sol.	0.0	0.0	0.0

Figure in bold italic: the best among the nine algorithms for a specific problem case with 18 instances.
Figure in bold non-italic: the second best among the nine algorithms for a specific problem case with 18 instances.

Table 3 illustrates computation results for the nine subgradient procedures. The results are summarized as the average values (over the 27 problem instances) of OBJ (the objective function value, i.e., the total of penalty cost and TDT (the total delay time)), TDT (h) and CPU time (s), and the total values (of the 27 instances) of the count of best solutions among the nine procedures and the count of optimal solutions. Like the DBAP analysis, for each line, figures in bold italic are the best while ones in bold non-italic are the second best.

Like the DBAP, the BTPS procedures produced negative LB since the BTPS procedures (the DBAP with the cylinder constraint) assume $S_i = 0$; for this reason, LB and GAP are not shown in Table 3. Therefore, it is impossible to judge if an obtained solution is optimal or not by using LB. However, the judgment is possible by the TDT. If the BTPS solution is BTPT-feasible (i.e., the BTPS solution equivalent to a feasible one for the BTPT where all the ships are to be served), the solution with TDT = 0 is optimal because of the unique ship handling time over berths. Of course, the optimal solutions could have a non-zero TDT, but by no means we can judge if those solutions are optimal or not. In case of BTPT-infeasible (i.e., the BTPS solution does not serve all the ships), the solution with TDT = 0 cannot be judged to be optimal to the BTPS problem.

The five scenarios with different calling ships correspond to the range of different arrival intervals from 1.4 to 2.8 h, as indicated in the table. At the bottom are the average values of those evaluation terms over the five scenarios.

First, we examine the average values over the five scenarios. Comparing SIMPLE, INDIVIDUAL and INTERACT, we have an observation that INDIVIDUAL with C_j is the best among the three procedures according to the average OBJ over the five scenarios. The second best is INTERACT with C_j. This tendency is against our expectation that algorithms with weighted handling time are superior to the ones with original handling time. In fact, this trend is completely different from the one for the DBAP, since INTERACT is better than INDIVIDUAL for the DBAP. However, as a common feature between the DBAP and BTPS, SIMPLE is the worst. With other weighted handling times, INDIVIDUAL is better than INTERACT.

Looking at the results for each scenario, except for the 50-ship case, all three procedures of SIMPLE, INDIVIDUAL or INTERACT with C_j perform well. INTERACT outperforms with fewer ships, while INDIVIDUAL does with more ships. Nevertheless, the results of both procedures are almost the same.

Regarding the count of best and optimal solutions, INTERACT with C_j/j and $C_j/\log(j)$ are the best and the second best, respectively. It is envisaged that the low average value of OBJ is attributed to large OBJ values due to the large value of penalty costs for BTPT-infeasible cases where too many ships request calling compared to the berthing capacity within the cylinder. In fact, 10–30% of the 27 instances are BTPT-infeasible.

Table 4 shows results for BTPT-feasible cases, while Table 5 illustrates ones for BTPT-infeasible. According to Table 4, INTERACTs with C_j/j and with $C_j/\log(j)$ are the best and the second best in OBJ, respectively. So, as an overall trend, INTERACTs with weighted handling time are superior like the DBAP trend.

Meanwhile, as Table 5 indicates, INDIVIDUAL and INTERACT with C_j outperform like overall results of the BTPS as shown in Table 3. Note that the TDT is small with INDIVIDUAL and INTERACT with C_j/j. This tendency is quite similar to the ones for the overall BTPS results and for the BTPT-feasible cases. Thus, INDIVIDUAL and INTERACT with C_j/j work well to minimize the TDT, since the BTPS for the BTPT-feasible cases is equivalent to the minimization of $(\mathrm{TST} - \sum_j C_j)$ as the DBAP. In the

Table 6
BTPS results for BTPT-infeasible cases (T = 100).

T	Interval (h)		C_j	C_j/j	$C_j/\log(j)$
SIMPLE					
100	1.4	OBJ	3036087.7	**3033864.3**	3167882.9
		TDT (h)	6921.0	3864.3	4549.6
		# of best sol.	**3**	***5***	1
		# of opt sol.	0	0	0
INDIVIDUAL					
100	1.4	OBJ	3034748.2	3185148.1	3233968.0
		TDT (h)	3914.8	**2648.1**	3134.7
		# of best sol.	2	0	0
		# of opt sol.	0	0	0
INTERACT					
100	1.4	OBJ	***3029935.0***	3177727.1	3203976.2
		TDT (h)	4101.7	***2643.1***	3142.8
		# of best sol.	0	0	1
		# of opt sol.	0	0	0

Figure in bold italic: the best among the nine algorithms for a specific problem case with 18 instances.
Figure in bold non-italic: the second best among the nine algorithms for a specific problem case with 18 instances.

BTPT-feasible cases, the minimization of TDT is substantially the objective for the BTPS; consequently INDIVIDUAL and INTERACT with C_j/j perform well for the BTPS problem. On the other hand, for the BTPS-infeasible cases, the penalty cost overwhelms the TDT in the objective function. Thus, the algorithms attempt to serve as many ships as possible. This result is unlikely obtained by minimizing the TDT, for which INDIVIDUAL and INTERACT with C_j/j outperform. As a conclusion, INDIVIDUAL and INTERACT with C_j are the best for the BTPT-infeasible cases. Overall results for the BTPS are attributed to the BTPT-infeasible cases due to the large value of the penalty cost. In terms of the count of best solution, INTERACT with C_j/j is the best as the overall performance. This is because there are only few BTPT-infeasible cases out of the 27 instances where INTERACT with C_j outperforms. For all the other instances (i.e., BTPT-feasible), INTERACT with C_j/j is the best.

Lastly, having created nine more BTPT-infeasible instances, we found a BTPS solution for totally 12 BTPT-infeasible instances with 100 ships and 8 berths. Table 6 shows the average values of OBJ and TDT and total counts of best and optimal solutions over the 12 instances. This also indicates the superiority of INTERACT with C_j like Table 5.

6. Conclusions

This paper addressed a relatively long-term decision making of berth schedule, the berth template problem in discrete berthing locations. Two problems, namely the strategic and tactical berth template problems, deal with the situations with different planning intervals and strategic importance. The strategic one (BTPS) chooses ships to be served and those not to be served for the case with excessive calling requests when compared to the berthing capacity. For such as a case, the tactical one (BTPT) usually cannot find a feasible solution. Therefore, the former is in general more applicable to congested berthing situations since it always finds a solution. In particular, we incorporate in the model the practical consideration about the simultaneous treatment of mother and feeder ships under the hub-and-spoke operation. Although some BTPT studies have been done for the last few years, we have not found a BTP study addressing these strategic issues as the BTPS in this paper does.

Regarding the solution algorithm, we developed a subgradient procedure with Lagrangian relaxation to find an approximate solution to the BTPS. In a previous work by the authors, the solution algorithms of the berth allocation problem (DBAP) for operational scheduling were developed based on the subgradient procedure with the Lagrangian relaxation. Since the BTPS shares the key structure of problem formulation with the DBAP, the subgradient approach has been applied to the BTPS. However, the developed solution algorithms for the BTPS are not a straightforward extension of the subgradient procedure for the DBAP and we have exploited a modification of the procedure. Based on the numerical experiments, the modified algorithm works very well for the DBAP. In particular, INTERACT with $C_j/\log(j)$ is superior. As for the BTPS, according to the experiments, INDIVIDUAL with C_j is the best among others regarding the UB performance while INTERACT with C_j/j is superior in terms of the best and optimal solution counts. In summary, the modified algorithms work well for both the DBAP and BTPS as an overall evaluation.

Finally, the modeling framework and the solution algorithms that are developed in this paper should be useful for the terminal operators to better manage their valuable resources. In particular, since the BTPS model incorporates the decision of selecting the calling ships/shipping lines strategically, it is very suitable for the situations with excessive demand or the cases of capacity expansion. Thus, we believe this study can serve as a useful decision support to terminal operators from a practical point of view.

Acknowledgement

The authors would like to thank the three anonymous referees for their constructive comments. This research is supported by the JSPS Grant-in-Aid for Scientific Research-B Grant 25282093.

Appendix A. Transformation of the objective function

We derive $\sum_{i\in B}\sum_{j\in W_i}\sum_{k\in U\setminus\{T\}}\alpha_{ijk}\sum_{l\in V}\sum_{m\in P_k}x_{ilm}=\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in W_i}\sum_{m<k}\alpha_{ilm}x_{ijk}$.

The formulation can simply be transformed as $\sum_{i\in B}\sum_{j\in W_i}\sum_{k\in U\setminus\{T\}}\alpha_{ijk}\sum_{l\in V}\sum_{m\in P_k}x_{ilm}=\sum_{i\in B}\sum_{j\in W_i}\sum_{k\in U\setminus\{T\}}\alpha_{ijk}\sum_{l\in V}\sum_{m>k}x_{ilm}$. We substitute k' for m and m' for k, then we have $\sum_{i\in B}\sum_{j\in W_i}\sum_{m'\in U\setminus\{T\}}\alpha_{ijm'}\sum_{l\in V}\sum_{k'>m'}x_{ilk'}$. Since element k' is more than m', k' belongs to a subset $U\setminus\{1\}$. Reversely, m' is less than k'. Consequently, we have $\sum_{i\in B}\sum_{j\in W_i}\sum_{m'\in U\setminus\{T\}}\alpha_{ijm'}\sum_{l\in V}\sum_{k'>m'}x_{ilk'}=\sum_{i\in B}\sum_{j\in W_i}\sum_{k'\in U\setminus\{1\}}\sum_{l\in V}\sum_{m'<k'}\alpha_{ijm'}x_{ilk'}$. Once again substituting k and m for k' and m', respectively, we have $\sum_{i\in B}\sum_{j\in V}\sum_{k\in U\setminus\{1\}}\sum_{l\in W_i}\sum_{m<k}\alpha_{ilm}x_{ijk}$.

Appendix B. Subgradient procedure for BTPS

Step 1. Maxiter = 200, $d=2$, $\bar{Z}=1\times10^8$, Iter = 1, $n=1$, BestLB = 0, $(\alpha=\alpha^*=0,\ \beta=\beta^*=0,\ \gamma=\gamma^*=0,\ \delta=\delta^*=0)$.

Step 2. Solve problem [RBTPT], and calculate its objective function. Let Z_{RBTPD} be the solution value of [RBTPT]. If Z_{RBTPD} > BestLB, let BestLB = Z_{RBTPD}, Iter = 1, $(\alpha^*=\alpha,\ \beta^*=\beta,\ \gamma^*=\gamma,\ \delta^*=\delta)$, otherwise Iter = Iter + 1.

Step 3. Perform a heuristic by using an optimal solution to [SUB-1] to find a feasible solution to [BTPT]. If the feasible solution is not found, STOP; otherwise, let FEAS be the objective function value of the feasible solution. If FEAS<$\bar{Z}$, let $\bar{Z}$=FEAS, If $\bar{Z}-$ BestLB < 1, STOP.

Step 4. Let $n=n+1$. If $n>$ Maxiter, STOP; otherwise continue.

Step 5. If Iter > 20 let Iter = 1, $(\alpha^*=\alpha,\ \beta^*=\beta,\ \gamma^*=\gamma,\ \delta^*=\delta)$, $d_n=d_n/2$, otherwise calculate step size t_n and update multipliers α_{ijk}, β_i, γ_{ik} and $\delta_{jj'}$.

Step 6. If α_{ijk}, β_i, γ_{ik} or $\delta_{jj'}<0$, then set it = zero. Go to Step 2.

Appendix C. SBAP and DBAP formulations

[SBAP]

$$\text{Minimize}\ \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}(kC_{ij}+S_i-A_j)x_{ijk}+\sum_{i\in B}\sum_{j\in W_i}\sum_{k\in U}ky_{ijk} \tag{A.1}$$

$$\text{subject to}\ \sum_{i\in B}\sum_{k\in U}x_{ijk}=1\quad\forall j\in V, \tag{A.2}$$

$$\sum_{j\in V}x_{ijk}\leqslant 1\quad\forall i\in B,k\in U, \tag{A.3}$$

$$x_{ijk}\in\{0,1\}\quad\forall i\in B,j\in V,k\in U, \tag{A.4}$$

[DBAP]

$$\text{Minimize}\ \sum_{i\in B}\sum_{j\in V}\sum_{k\in U}(kC_{ij}+S_i-A_j)x_{ijk}+\sum_{i\in B}\sum_{j\in W_i}\sum_{k\in U}ky_{ijk} \tag{A.1}$$

$$\text{subject to}\ (A.2-A.4)$$

$$\sum_{l\in V}\sum_{m\in P_k}(C_{il}x_{ilm}+y_{ilm})+y_{ijk}-(A_j-S_i)x_{ijk}\geqslant 0\quad\forall i\in B,j\in W_i,\ k\in U, \tag{A.5}$$

$$y_{ijk}\ \geqslant 0\ \forall i\in B,j\in V,k\in U, \tag{A.6}$$

where S_i is the start time of the availability of berth i and C_{ij} is the handling time being spent by ship j at berth i. Other parameters and variables are the same as the ones for [BTPS]. Constraints (A.2–A.6) are basically the same as (2–4), (9), (10). All the calling ships are already in port in the SBAP, while they are not all so in the DBAP.

Note here that formulations above are slightly different from the one in Imai et al. (2001). The difference arises from the structure of parameters in the objective function due to the service order scheme. That is, Imai et al. (2001) has $\{(T-k'+1)C_{ij}+S_i-A_j\}x_{ijk}$ in the objective function where the service order k' is numbered in ascending order from the first one to be served. However, the x_{ijk} variable-associated coefficient in Imai et al. and the one in this paper are completely equivalent, since $(T-k'+1)$ for k' increasing from 1 to T turns to be k that decreases from T to 1. The objective (A.1) is used in this paper because of the formulation simplicity.

Appendix D. Modified subgradient procedure for BTPS

Step 1. Maxiter = 200, $d=2$, $\bar{Z}=1\times10^8$, Iter = 1, $n=1$, BestLB = 0, $(\alpha=\alpha^*=0,\ \beta=\beta^*=0,\ \gamma=\gamma^*=0,\ \delta=\delta^*=0)$.

Step 2. Solve problem [RBTPT], and calculate its objective function. Let Z_{RBTPD} be the solution value of [RBTPT]. If Z_{RBTPD} > BestLB, let BestLB = Z_{RBTPD}, Iter = 1, $(\alpha^*=\alpha,\ \beta^*=\beta,\ \gamma^*=\gamma,\ \delta^*=\delta)$, otherwise Iter = Iter + 1.

Step 3–1. Solve problem [MBTPT], which is the same as [RBTPT] but with C_j/j instead of C_j.

Step 3–2. Perform a heuristic by using an optimal solution to [SUB-3'] to find a feasible solution to [BTPT]. If the feasible solution is not found, STOP; otherwise, let FEAS be the objective function value of the feasible solution. If FEAS< $\bar{Z}$, let $\bar{Z}$=FEAS, If $\bar{Z}-$ BestLB < 1, STOP.

Step 4. Let $n = n + 1$. If $n >$ Maxiter, STOP; otherwise continue.

Step 5. If Iter > 20 let Iter = 1, $(\alpha^* = \alpha,\ \beta^* = \beta,\ \gamma^* = \gamma,\ \delta^* = \delta)$, $d_n = d_n/2$, otherwise calculate step size t_n and update multipliers α_{ijk}, β_i, γ_{ik} and $\delta_{jj'}$.

Step 6. If α_{ijk}, β_i, γ_{ik} or $\delta_{jj'} < 0$, then set it = zero. Go to Step 2.

References

Bramel, J., Simchi-Levi, D., 1997. The Logic of Logistics. Springer-Verlag, New York.

Brucker, P., Kampmeyer, T., 2008. A general model for cyclic machine scheduling problems. Discrete Appl. Math. 156, 2561–2572.

Buhrkal, K., Zuglian, S., Ropke, S., Larsen, J., Lusby, R., 2011. Models for the discrete berth allocation problem: a computational comparison. Transp. Res. Part E 47, 461–473.

Cheong, C.Y., Tan, K.C., Liu, D.K., Lin, C.J., 2010. Multi-objective and prioritized berth allocation in container ports. Ann. Oper. Res. 180, 63–103.

Cordeau, J.-F., Laporte, G., Legato, P., Moccia, L., 2005. Models and tabu search heuristics for the berth-allocation problem. Transp. Sci. 39, 526–538.

Crama, Y., Van de Klundert, J., 1997. Cyclic scheduling of identical parts in a robotic cell. Oper. Res. 45, 952–965.

Crama, Y., Kats, V., Van de Klundert, J., Levner, E., 2000. Cyclic scheduling in robotic flowshops. Ann. Oper. Res. 96, 97–124.

De Oliveira, R.M., Mauri, G.R., Lorena, L.A.N., 2012. Clustering search for the berth allocation problem. Expert Syst. Appl. 39, 5499–5505.

Giallombardo, G., Moccia, L., Salani, M., Vacca, I., 2010. Modeling and solving the tactical berth allocation problem. Transp. Res. Part B 44, 232–245.

Golias, M.M., Boile, M., Theofanis, S., 2009. Berth scheduling by customer service differentiation: a multi-objective approach. Transp. Res. Part E 45, 878–892.

Golias, M.M., Boile, M., Theofanis, S., 2010. A lamda-optimal based heuristic for the berth scheduling problem. Transp. Res. Part C 18, 794–806.

Hall, N.G., Kamoun, H., Sriskandarajah, C., 1997. Scheduling in robotic cells: classification, two and three machine cells. Oper. Res. 45, 421–439.

Hanen, C., 1994. Study of a NP-hard cyclic scheduling problem: the recurrent job-shop. Eur. J. Oper. Res. 72, 82–101.

Hansen, P., Oguz, C., Mladenovic, N., 2008. Variable neighborhood search for minimum cost berth allocation. Eur. J. Oper. Res. 191, 636–649.

Hendriks, M.P.M., Armbruster, D., Laumanns, M., Lefeber, E., Udding, J.T., 2012. Strategic allocation of cyclically calling vessels for multi-terminal container operators. Flex. Serv. Manuf. J. 24, 248–273.

Hendriks, M.P.M., Lefeber, E., Udding, J.T., 2013. Simultaneous berth allocation and yard planning at a tactical level. OR Spectr. 35, 441–456.

Imai, A., Nagaiwa, K., Chan, W.T., 1997. Efficient planning of berth allocation for container terminals in Asia. J. Adv. Transp. 31, 75–94.

Imai, A., Nishimura, E., Papadimitriou, S., 2001. The dynamic berth allocation problem for a container port. Transp. Res. Part B 35, 401–417.

Imai, A., Nishimura, E., Papadimitriou, S., 2003. Berth allocation with service priority. Transp. Res. Part B 37, 437–457.

Imai, A., Nishimura, E., Papadimitriou, S., 2005a. Corrigendum to "The dynamic berth allocation problem for a container port" [Transportation Research Part B 35 (2001) 401–417]. Transp. Res. Part B 39, 197.

Imai, A., Sun, X., Nishimura, E., Papadimitriou, S., 2005b. Berth allocation in a container port: using a continuous location space approach. Transp. Res. Part B 39, 199–221.

Imai, A., Nishimura, E., Hattori, M., Papadimitriou, S., 2007. Berth allocation at indented berths for mega-containerships. Eur. J. Oper. Res. 179, 579–593.

Imai, A., Nishimura, E., Papadimitriou, S., 2008. Berthing ships at a multi-user container terminal with a limited quay capacity. Transp. Res. Part E 44, 136–151.

Imai, A., Nishimura, E., Papadimitriou, S., 2013. Marine container terminal configurations for efficient handling of mega-containership. Transp. Res. Part E 49, 141–158.

JICT, 2014. Jakarta International Container Terminal, Berthing Contract. Available from: <http://jict.co.id/?x0=berthing%2Bcontract&x1=26&x2=article>.

Lalla-Ruiz, E., Melian-Batista, B., Moreno-Vega, J.M., 2012. Artificial intelligence hybrid heuristic based on tabu search for the dynamic berth allocation problem. Eng. Appl. Artif. Intell. 25, 1132–1141.

Lee, D.-H., Jin, J.G., 2013. Feeder vessel management at container transshipment terminals. Transp. Res. Part E 49, 201–216.

Lee, T.-E., Posner, M.E., 1997. Performance measures and schedules in periodic job shops. Oper. Res. 45, 72–91.

Lee, D.-H., Chen, J.H., Cao, J.X., 2010. The continuous berth allocation problem: a greedy randomized adaptive search solution. Transp. Res. Part E 46, 1017–1029.

Li, C.-L., Cai, X., Lee, C.-Y., 1998. Scheduling with multiple-job-on-one-processor pattern. IIE Trans. 30, 433–445.

Lim, A., 1998. The berth planning problem. Oper. Res. Lett. 22, 105–110.

Matsuo, H., Shand, J.S., Sullivan, R.S., 1991. A crane scheduling problem in a computer-integrated manufacturing environment. Manage. Sci. 37, 587–606.

Mauri, G.R., Oliveira, A.C.M., Lorena, L.A.N., 2008. A hybrid column generation approach for the berth allocation problem. Lect. Notes Comput. Sci. 4972, 110–122.

McCormick, S.T., Rao, U.S., 1994. Some complexity results in cyclic scheduling. Math. Comput. Modell. 20, 107–122.

Monaco, M.F., Sammarra, M., 2007. The dynamic berth allocation problem: a strong formulation solved by Lagrangean approach. Transp. Sci. 41, 265–280.

Mongelluzzo, B., 2013. Decline of the single-user terminal. J. Commerce 14, 36–38.

Moorthy, R., Teo, C.-P., 2006. Berth management in container terminal: the template design problem. OR Spectr. 28, 495–518.

Nishimura, E., Imai, A., Papadimitriou, S., 2001. Berth allocation planning in the public berth system by genetic algorithms. Eur. J. Oper. Res. 131, 282–292.

Norman, B.A., Bean, J.C., 1999. A genetic algorithm methodology for complex scheduling problems. Nav. Res. Logist. 46, 199–211.

Park, K.T., Kim, K.H., 2002. Berth scheduling for a container terminals by using a sub-gradient optimization technique. J. Oper. Res. Soc. 53, 1054–1062.

Pinedo, M., 2012. Scheduling: Theory, Algorithms, and Systems. Springer US, Boston, MA.

Roundy, R., 1992. Cyclic schedules for job-shops with identical jobs. Math. Oper. Res. 17, 842–865.

Saharidis, G.K.D., Golias, M.M., Boile, M., Theofanis, S., Ierapetritou, M.G., 2010. The berth scheduling problem with customer differentiation: a new methodological approach based on hierarchical optimization. Int. J. Adv. Manuf. Technol. 46, 377–393.

Ting, C.-J., Wu, K.-C., Chou, H., 2014. Particle swarm optimization algorithm for the berth allocation problem. Expert Syst. Appl. 41, 1543–1550.

Vacca, I., Salani, M., Bierlaire, M., 2013. An exact algorithm for the integrated planning of berth allocation and quay crane assignment. Transp. Sci. 47, 148–161.

Xu, D., Li, C.-L., Leung, J.Y.-T., 2012. Berth allocation with time-dependent physical limitations on vessels. Eur. J. Oper. Res. 216, 47–56.

Zhen, L., Chew, E.P., Lee, L.H., 2011. An integrated model for berth template and yard template planning in transshipment hubs. Transp. Sci. 45, 483–504.