PERGAMON

Transportation Research Part B 35 (2001) 401–417

TRANSPORTATION RESEARCH PART B

www.elsevier.com/locate/trb

The dynamic berth allocation problem for a container port

Akio Imai [a,*], Etsuko Nishimura [a], Stratos Papadimitriou [b]

[a] *Department of Transportation and Information Systems Engineering, Kobe University of Mercantile Marine, Fukae, Higashinada, Kobe 658-0022, Japan*

[b] *Department of Maritime Studies, University of Piraeus, 18532 Piraeus, Greece*

Received 14 August 1999; received in revised form 13 September 1999; accepted 4 October 1999

Abstract

This paper addresses the problem of determining a dynamic berth assignment to ships in the public berth system. While the public berth system may not be suitable for most container ports in major countries, it is desired for higher cost-effectiveness in Japan's ports. The berth allocation to calling ships is a key factor for efficient public berthing. However, it is not calculated in polynomially-bounded time. To obtain a good solution with considerably small computational effort, we developed a heuristic procedure based on the Lagrangian relaxation of the original problem. We conducted a large amount of computational experiments which showed that the proposed algorithm is adaptable to real world applications.

Keywords: Container transportation; Port operations; Heuristics

1. Introduction

Most container berths in major ports are leased by ship operators in order for them to be directly involved and responsible for the processing of containers and thus achieve higher productivity. Whereas this is justified in the case of a firm handling a large volume of containers with a large number of ship calls, it may not result in cost savings if these quantities are not sufficient. Charges in Japan's ports have been consistently higher than those in other major hubs over several years. Part of the increased cost is the result of overcapitalization of the port for the relatively small cargo volume.

* Corresponding author. Tel.: +81-78-431-6261; fax: +81-78-431-6365.
E-mail address: imai@bun.ti.kshosen.ac.jp (A. Imai).

0191-2615/01/$ - see front matter
PII: S0191-2615(99)00057-0

In the above context, it is of interest to limit the number of berths by introducing a public berth system. The berth allocation in this system, i.e., the assignment of berths to incoming ships for their cargo handling, plays an important role in minimizing the turnaround time; this is because the handling time for a specific ship is not necessarily the same at every berth.

The goal of this paper is to propose an algorithm for solving the berth allocation problem (BAP). A solution technique for the so-called static berth allocation problem (SBAP), that, given a collection of ships, finds a set of assignments to minimize the sum of the time they spend waiting for berths and handling cargo, has already been proposed (Imai et al., 1997). We consider in this paper the dynamic version of the problem (DBAP), which is similar to the SBAP but with the difference that ships arrive while work is in progress.

The SBAP may be formulated as a three dimensional assignment problem and can be reduced to a two-dimensional (or classical) assignment problem that is easily solved with the Hungarian method (Papadimitriou and Steiglitz, 1982). However, the DBAP is represented by a three dimensional assignment problem with some additional constraints, and it is not solved in polynomially-bounded time. Consequently we present a Lagrangian relaxation-based heuristic algorithm for the DBAP.

The paper is organized as follows. The next section gives the motivation for introducing a public berth system in major container ports of Japan and further provides a literature review on public berthing systems. A mixed integer programming formulation of the DBAP and Lagrangian relaxation of that formulation are presented in Section 3. In Section 4 we discuss a solution procedure employing the subgradient method based on the Lagrangian relaxation. Section 5 shows a large number of computational experiments and the final section discusses conclusions.

2. Importance of public berth allocation

We propose the public berth system for container port management. We here discuss issues in port management to show the motivation for the proposed public berthing. Further we describe existing fields of study relating to the public berthing.

2.1. Desired public berthing

By its geographical nature Japan's ports have been recognized as hubs connecting deepsea vessel traffic and local feeders for Asian countries. However, as the volume of container traffic to and from eastern Asia increases, most of the deepsea vessels have begun to call directly at some ports in that area. As a result, facilities and equipment in Japan's ports became redundant due to the decreasing cargo demand, thus making the cost of the private berthing system relatively high.

Table 1 shows the length of time each berth was occupied in port of Kobe during the month of February in 1996. The time is categorized for the number of container berths occupied at same time. It is noteworthy that in only nine hours out of twenty nine days were all the berth occupied simultaneously. The low usage of berths coupled with small amount of cargo handled results in relatively high port charges per container. The public berthing system proposed in this paper aims at reducing the number of required berths by appropriate ship-berth assignments, while maintaining at least the same level of service.

Table 1
Time lengths of berth occupation

Number of berths	0	1	2	3	4	5	6	7	Total
Time (hours)	19	76	138	159	136	90	62	9	689
%	3	11	20	23	20	13	9	1	100

Table 2
Waiting time for berthing

Number of berths	3	4	5	6	7
Time (minutes)	221	36	10	2	0

Generally, ships are serviced in a port on a First-Come-First-Served (FCFS) basis, which however, does not necessarily minimize the total staying time of ships. If the ship arrival order is ignored, the total cumulative staying time for all ships can be reduced. This, however, may yield longer waiting time for some ships.

In a previous research (Nishimura, 1998), a simple simulation for the public berthing system was carried out in order to examine the average time a ship spends for service, with the number of berths ranging from three to seven. Table 2 reports the average waiting time, showing that even if four berths are available then average waiting time is half an hour. The result strongly supports the public berthing system.

The public berth system may yield inequitable waiting time for each ship. This may not make the port attractive for carriers even if the port charges are low. However, the issue of equity is less important than port charges as ships had inequitable waiting times at every private berth in Kobe (Nishimura, 1998).

2.2. Related literature

Most port studies focus their attention to the strategic and tactical problems. As many container berths are privately operated by specific shipping companies, very few studies have been conducted on berth allocation in the public berth system.

Lai and Shih (1992) proposed some heuristic algorithms for a BAP which is motivated by more efficient berth usage in the HIT terminal of Hong Kong. Their problem assumes the FCFS allocation strategy, while our problem does not. Therefore, their solution may not be as good as ours.

Brown et al. (1994, 1997) treat ship berthing in naval ports. They identify the optimal set of ship-to-berth assignments that maximizes the sum of benefits for ships while in port. Berth planning in naval ports has important differences from berth planning in commercial ports. In the former, a berth shift occurs when for proper services, a newly arriving ship must be assigned to a berth where another ship is already moored. This treatment is unlikely in commercial ports. Berth shifting as well as other factors less relevant to commercial ports are taken into account in Brown et al., thus making their problem inappropriate for commercial ports.

Imai et al. (1997) consider a BAP for commercial ports. Most service queues are in general processed on a FCFS basis. They conclude that for high port throughput, optimal ship-to-berth assignments should be found without considering the FCFS basis. However, this may result in some ships' dissatisfaction regarding order of service. In order to deal with the two criteria to evaluate, i.e., berth performance and dissatisfaction on order of service, they developed a heuristic algorithm to find a set of non-inferior solutions while maximizing the former and minimizing the latter. Their berthing principle, however, cannot treat the dynamic allocation.

The BAP is one of the parallel machine scheduling problems. A job and a machine can be treated as a ship and a berth, respectively. The SBAP reduces to a classical assignment problem (or a weighted bipartite matching problem) that is known to be polynomially-solvable (Pinedo, 1995). Bean et al. (1991) and Norman and Bean (1999) deal with the machine scheduling problems where each job has a release time that corresponds to a ship arrival time in the DBAP. The problem in Bean et al. is different from our problem because theirs is a scheduling problem in the job shop environment. Norman and Bean treat the parallel machine scheduling; however they assume identical machines in parallel whereas ours deals with unrelated machines in parallel.

3. Problem formulation and Lagrangian relaxation

In this section, a mixed integer programming formulation of the DBAP and Lagrangian relaxation of that formulation are presented. First the formulation for the simple version of berth allocation, the SBAP, is discussed and then extended to the DBAP.

3.1. Formulation of the static berth allocation

In the SBAP, all the ships are already in port when the berthing plan is determined. This guarantees every potential ship-berth-order assignment is feasible in the berth allocation.

We assume that each berth can service one ship at a time and that there are no physical or technical restrictions such as a relationship between ship draft and water depth. Further, for generalizing the berth allocation, the ship handling time is assumed dependent on the berth where it is assigned. The second assumption is justified by the following reasoning: at a private container berth, loaded containers are stored in appropriate locations in a terminal alongside the berth where the ship is moored. For public berthing, ship-to-berth assignments should be, in general, determined in advance of ship arrivals; however containers for the ships may arrive at the terminal for loading after the berthing decision. Therefore, the distance between a ship and its container location depends on the berth assignment. Although examination of the handling systems in terminals is beyond the scope of this paper, it is obvious that the handling time may be also dependent on the geographical relationship between the ship and the container location. Note that the handling time is assumed deterministic in this study, although a berthing plan may be determined before all the containers arrive at the port.

The objective is to minimize the sum of waiting time for the availability of the berth assigned to each ship, plus the handling time it spends at the berth.

In formulating the BAP we define binary variables x_{ijk} to specify if ship j is to be serviced as the kth ship at berth i. Other related studies (i.e., berth assignment in naval ports as discussed in the

previous section) use actual unit time for decision variables to index assignment sequences. In the berth allocation planning, it is considered that a ship spends up to 24 hours for cargo handling. Considering the prospective planning horizon of our model (say, at least a few days), we will have an explosively large number of decision variables by describing them with actual time. Further, this model guarantees consecutive service for all ships without disruptions such as berth shifting. A model using decision variables indexed by actual time must become considerably complicated to ensure this constraint.

As the x_{ijk} variable is restricted to 0–1 values, the SBAP may be formulated as an integer three-dimensional assignment problem as follows:

$$\text{[PS]} \qquad \text{Minimize} \quad \sum_{i\in B}\sum_{j\in V}\sum_{k\in O}\{(T-k+1)C_{ij}+S_i-A_j\}x_{ijk} \tag{1}$$

Subject to

$$\sum_{i\in B}\sum_{k\in O}x_{ijk}=1 \quad \forall j\in V, \tag{2}$$

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

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

where

$i\ (=1,\ldots,I)\in B$	set of berths
$j\ (=1,\ldots,T)\in V$	set of ships
$k\ (=1,\ldots,T)\in O$	set of service orders
S_i	time when berth i becomes idle for the berth allocation planning
A_j	arrival time of ship j
C_{ij}	handling time spent by ship j at berth i
x_{ijk}	1 if ship j is serviced as the kth ship at berth i
	0 otherwise

Notice that both sets of ships and service orders have the same number of elements T because a feasible solution may have all the ships serviced at a particular berth.

The objective (1) minimizes the sum of waiting and handling times for every ship. Note that $S_i\geqslant A_j$ for all i, j from the problem definition. Constraint set (2) ensures that every ship must be serviced at some berth in any order of service. Constraint set (3) enforces that every berth services up to one ship at any time. Since this is the static problem, we assume $S_i\geqslant A_j$.

In the objective function, a handling time C_{ij} is weighted by $(T-k+1)$. This results from the observation that the handling time C_{ij} of a specific ship serviced at berth i contributes waiting time to the ships to be serviced at the same berth after it. In other words, the waiting time of a particular ship is represented by the cumulative handling time of its predecessors.

According to the formulation, it is possible that ship berthing is not necessarily scheduled in consecutive order; for example, given three ships with two berths, x_{ijk} may be assigned as $x_{111}=0,\ x_{121}=1,\ x_{131}=0,\ x_{112}=0,\ x_{122}=0,\ x_{132}=0,\ x_{113}=1,\ x_{123}=0,$ and $x_{133}=0$ for berth 1. The objective value of this solution is not correctly computed. The handling time of ship 2

is added to the waiting time of ships berthed as both the second and third ships to be serviced, while correctly only for the third. However, in an optimal solution this never happens due to the following lemma:

Lemma 1. *At optimality, N ships assigned to a specific berth are scheduled to be serviced consecutively as the* $(T-N+1)$*th to the Tth ships.*

Proof. Assume that a solution where ships are not serviced consecutively is optimal. Suppose ship j and another ship are serviced as the kth and $k+2$th ships, respectively, while no ship is serviced as the $k+1$th ship. In the objective function, the handling time of ship j is added $T-k+1$ times for the potential ships serviced after it. If the $k+2$th ship is changed to be serviced as the $k+1$th ship, then its handling time is added $T-k$ times, resulting in smaller objective function value. This contradicts the assumption. □

Problem [PS] can be reformulated as follows by substituting indices $i \in B$ and $k \in O$ by $n \in N$, where letting $|N|$ be the cardinality of set N, $|N| = |B| \times |O|$.

$$[\mathrm{PS}'] \qquad \text{Minimize} \quad \sum_{j\in V}\sum_{n\in N} D_{jn}x_{jn} \tag{5}$$

Subject to

$$\sum_{n\in N} x_{jn} = 1 \quad \forall j \in V, \tag{6}$$

$$\sum_{j\in V} x_{jn} \leqslant 1 \quad \forall n \in N, \tag{7}$$

$$x_{jn} \in \{0,1\} \quad \forall j \in V,\ n \in N, \tag{8}$$

where D_{jn} is an equivalent of parameter $(T-k+1)C_{ij}+S_i-A_j$. This is the classical two-dimensional assignment problem which is efficiently solved by the Hungarian method.

Without loss of generality, the first ship assigned to a specific berth in a solution of [PS] starts to be serviced immediately after the departure of the last ship in the previous planning.

3.2. Formulation of the dynamic berth allocation

The SBAP is restrictive in its use as some ships may arrive at the port during the planning horizon. If they are considered for the next horizon, overall berth performance might be worse. Thus, we consider the generalization of berth arrangement, i.e., the dynamic berth allocation problem (DBAP).

We assume in the DBAP that all the ships, while their arrival time is known in advance, do not arrive at the port before S_i of the assigned berth. Except for that all the assumptions for the SBAP hold.

The DBAP may be formulated as follows:

$$[\mathrm{PD}] \qquad \text{Minimize} \quad \sum_{i\in B}\sum_{j\in V}\sum_{k\in O}\{(T-k+1)C_{ij}+S_i-A_j\}x_{ijk} + \sum_{i\in B}\sum_{j\in W_i}\sum_{k\in O}(T-k+1)y_{ijk} \tag{9}$$

Subject to

$$\sum_{i\in B}\sum_{k\in O} x_{ijk} = 1 \quad \forall j \in V, \tag{10}$$

$$\sum_{j\in V} x_{ijk} \leqslant 1 \quad \forall i \in B,\ k \in O, \tag{11}$$

$$\sum_{i\in V}\sum_{m\in P_k} (C_{il}x_{ilm} + y_{ilm}) + y_{ijk} - (A_j - S_i)x_{ijk} \geqslant 0 \quad \forall i \in B,\ j \in W_i,\ k \in O, \tag{12}$$

$$x_{ijk} \in \{0,1\} \quad \forall i \in B,\ j \in V,\ k \in O, \tag{13}$$

$$y_{ijk} \geqslant 0 \quad \forall i \in B,\ j \in V,\ k \in O, \tag{14}$$

where

P_k subset of O such that $P_k = \{p | p < k \in O\}$
W_i subset of ships with $A_j \geqslant S_i$
y_{ijk} idle time of berth i between the departure of the $k-1$th ship and the arrival of the kth ship when ship j is serviced as the kth ship

The objective function (9) minimizes the total of waiting and handling times for every ship. Constraints (12) assure that ships must be serviced after their arrival. We will discuss the derivation of (9) and (12) in the next section.

As will be stated in Section 4.1, integrality of the objective function saves computational time when the problem finds the optimal solution using the subgradient method. Therefore, we assume C_{ij}, S_i and A_j all have integer values. For this, the y_{ijk} variables can technically have integer values to indicate precedence relationship between a ship arrival at port and the service for it. To account for service after arrival, y_{ijk} is simply defined as a time difference between the start of service for ship j and the departure of its immediate predecessor, both of which are integer. This naturally induces the integrality restriction when combined with the objective function. Thus, y_{ijk} is stated and implemented as a continuous variable.

3.3. Derivation of the objective function and constraints (12)

Given an example of ship assignment to order of service at a particular berth, we derive the objective function of the DBAP.

Fig. 1 shows a set of ship assignments to berth i. Thin lines represent ships' wait while thick lines imply the service for them. Dotted lines reveal berths in idle status. Ship 1 arriving at the port before S_i is the first ship to be serviced while ships 2 and 3 arriving after S_i are the fourth and fifth ships, respectively. Berth i is already idle for ships 4 and 5; therefore they are serviced as the second and third ships as soon as they arrive.

In general the handling time C_{ij} of each ship contributes to the waiting times for all of its successors. For instance, the handling time of ship 4 being serviced as the second ship may be part of the waiting time for ships 5, 2, and 3 (without consideration of the scheduled service sequence

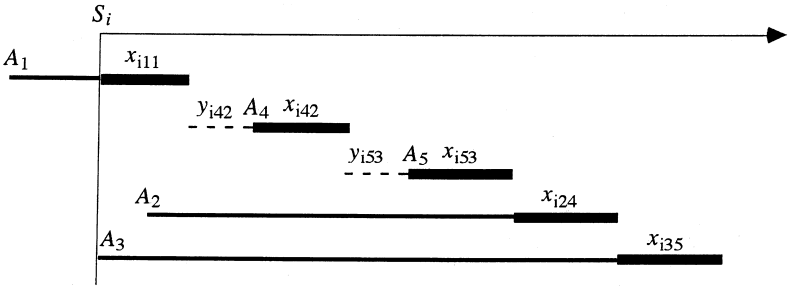

Fig. 1. Ship assignments to berth i.

in Fig. 1). Similarly, the idle time of a berth prior to service for a ship, y_{ijk}, must be summed up for the time its successors spend waiting for service. If a ship arrives before its immediate predecessor's departure, by definition $y_{ijk} = 0$ for it. For those ships that arrive before S_i, the time they spend waiting before S_i, i.e., $S_i - A_j$, is added. Oppositely, if ship j arrives after S_i (i.e., $S_i - A_j$ is negative) regardless of their $y_{ijk} > 0$ or $= 0$, then waiting times summed by C_{ij}s and y_{ijk}s of their predecessors are subtracted by time duration of $A_j - S_i$. Notice that the objective function contains $(T - k + 1)y_{ijk}$, i.e., y_{ijk} is computed for ship j as the kth ship as well as its successors, although it should not be for itself. As stated above, the waiting time of a certain ship given as the sum of C_{ij} and y_{ijk} for its predecessors is subtracted by $A_j - S_i$ if $y_{ijk} > 0$ for it; therefore its y_{ijk} must be added for its own waiting time.

Constraint (12) for ship j as the kth ship at berth i yields the following inequality by moving the third term of the left-hand side to the right:

$$\sum_{l\in V}\sum_{m\in P_k}(C_{ij}x_{ilm} + y_{ilm}) + y_{ijk} \geqslant (A_j - S_i)x_{ijk}. \tag{15}$$

The first term in the left-hand side is the time duration between S_i and the time when the last of its predecessors leaves the port. Thus, the left-hand side, i.e., the time duration between S_i and the start of the service for ship j, must be no less than $A_j - S_i$ if $x_{ijk} = 1$.

3.4. Lagrangian relaxation of the DBAP

The formulation of the DBAP is a mixed integer program which is not known to be solved in polynomially-bounded time. This may be solved by a branch and bound algorithm but that would be time-consuming for problems of practical size. The DBAP may be solved frequently to obtain a new berth allocation due to the changes of estimated ship arrival times. Consequently the branch and bound algorithm does not seem suitable for the DBAP. This encourages us to develop a heuristic for the problem. This heuristic procedure employs the subgradient optimization procedure based on the following Lagrangian relaxation of the original problem [PD]:

$$\begin{aligned}[P1] \qquad \text{Minimize} \quad & \sum_{i\in B}\sum_{j\in V}\sum_{k\in O}\{(T-k+1)C_{ij} + S_i - A_j\}x_{ijk} + \sum_{i\in B}\sum_{j\in W_i}\sum_{k\in O}(T-k+1)y_{ijk} \\ & - \sum_{i\in B}\sum_{j\in W_i}\sum_{k\in O}\lambda_{ijk}\left\{\sum_{j\in V}\sum_{m\in P_k}(C_{il}x_{ilm} + y_{ilm}) + y_{ijk} - (A_j - S_i)x_{ijk}\right\}\end{aligned} \tag{16}$$

Subject to

$$\sum_{i\in B}\sum_{k\in O} x_{ijk} = 1 \quad \forall j \in V, \tag{17}$$

$$\sum_{j\in V} x_{ijk} \leqslant 1 \quad \forall i \in B,\ k \in O, \tag{18}$$

$$x_{ijk} \in \{0,1\} \quad \forall i \in B,\ j \in V,\ k \in O, \tag{19}$$

$$y_{ijk} \geqslant 0 \quad \forall i \in B,\ j \in V,\ k \in O, \tag{20}$$

where λ_{ijk} is a Lagrangian multiplier for berth i, ship j, and the kth position of the service sequence and has non-negative value.

This formulation can be rewritten as follows, because y_{ijk}s are redundant as they are not in any constraints.

$$\text{[P2]} \qquad \text{Minimize} \quad \sum_{i\in B}\sum_{j\in W_i}\sum_{k\in O}\{(T-k+1)C_{ij} + S_i - A_j\}x_{ijk} - \sum_{i\in B}\sum_{j\in W_i}\sum_{k\in O}\lambda_{ijk}(S_i - A_j)x_{ijk} - \sum_{i\in B}\sum_{j\in W_i}\sum_{k\in O}\lambda_{ijk}\sum_{l\in V}\sum_{m\in P_k} C_{il}x_{ilm} \tag{21}$$

subject to (17)–(19).

Problem [P2], is further reformulated by introducing representative cost E_{ijk} in the objective function.

$$\text{[P3]} \qquad \text{Minimize} \quad \sum_{i\in B}\sum_{j\in V}\sum_{k\in O} E_{ijk}x_{ijk} \tag{22}$$

subject to (17)–(19).

With relaxing constraint set (12), formulation [PD] becomes a three-dimensional assignment problem that is reduced to the classical two-dimensional assignment problem and is therefore easy to solve.

4. Solution procedure

4.1. Subgradient method

The quality of the feasible solution obtained using the above procedure is strongly dependent on one's ability to determine good Lagrangian multipliers λ_{ijk}s. Essentially, with a zero multiplier in [P3] for each combination of (i,j,k), we assume that ships can be allocated to berths in order while minimizing the total of waiting and handling times. This may lead to an infeasible solution to [PD] as some ships may be serviced before their arrivals. As the multipliers corresponding to the ships with infeasible berthing (i.e., the scheduled services are performed while they are not yet in port) are increased, the "cost" of these ships are increased in the Lagrangian problem, [P3], probably resulting in later service for them. In other words, the solution to [P3] gets closer to a feasible solution to [PD].

Good multipliers are also important as the quality of the lower bound, i.e., the objective function value of [P3], is a function of these multipliers. The best lower bound corresponding to the optimal multiplier vector λ^* is determined as

$$Z_{P3}(\lambda^*) = \mathrm{Max}(Z_{P3}(\lambda)), \tag{23}$$

where $(Z_{P3}(\lambda))$ is the value of the Lagrangian function with a multiplier set (vector) λ. There are a number of different approaches to finding a good, if not optimal, set of Lagrangian multipliers (Fisher, 1981). We have selected to use the subgradient optimization procedure. The subgradient method is an adaptation of the gradient method in which the subgradients replace the gradients (Bazaara and Goode, 1979; Held et al., 1974). Since the approach has been widely utilized and well understood we will not repeat details here. It suffices to note that as a termination criterion we fixed the maximum number of iterations at 200. The procedure is also terminated if the gap between the feasible solution value and the Lagrangian bound becomes less than 1. Given integer objective function coefficients, this condition is sufficient to detect an optimal solution.

During each iteration of the procedure, [P3] is solved to obtain a lower bound for [PD]. Note that the objective function value of [P1] is equal to the objective function value of [P3] plus the value related to y_{ijk}. As the objective function of [P1] is a lower bound, y_{ijk} values are fixed to zero. One of SIMPLE, INDIVIDUAL, or INTERACT processes (discussed in Section 4.2) is performed at each iteration of the heuristic to determine a feasible solution to [PD]. At the time of termination, the subgradient optimization procedure reports the best feasible solution and the best lower bound generated in all of the iterations. The procedure is detailed as follows:

In the subgradient optimization procedure, given a set of starting multipliers λ^0, a sequence of multipliers is generated using the following rule:

$$\lambda^{k+1} = \lambda^k + t_k(Ax^k - b), \tag{24}$$

where x^k is an optimal solution to the Lagrangian problem P3(λ^k) and t_k is a positive scalar step size and $(Ax \leqslant b)$ is the set of constraints being relaxed (i.e., constraint set 12). We use the following step size that has been used frequently in the past:

$$t_k = d_k(\bar{Z} - Z_{P3}(\lambda^k))/\|Ax^k - b\|^2, \tag{25}$$

where $\bar{Z}$ is the best known feasible solution value, $\|Ax^k - b\|$ is the Euclidean norm of $Ax^k - b$, and d_k is a scalar satisfying the relation $0 \leqslant d_k \leqslant 2$. This scalar is set to 2 at the start of the procedure and is halved whenever the bound does not improve in 20 consecutive iterations. The following is a formal statement of the procedure:

Step 1. Maxiter $= 200,\ d = 2,\ \bar{Z} = 1 \times 10^8,\ \mathrm{Iter} = 1, k = 1,\ \mathrm{BestLB} = 0,\ \lambda = \lambda^* = \{0\}$.

Step 2. Solve problem [P1], and calculate the objective function of [P3]. Let Z_{P3} be the solution value of [P3]. If $Z_{P3} >$ BestLB, let BestLB $= Z_{P3}$, Iter $= 1,\ \lambda = \lambda^*$, otherwise Iter $=$ Iter $+ 1$.

Step 3. Perform one of SIMPLE, INDIVIDUAL, and INTERACT processes. 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 $k = k + 1$. If $k >$ Maxiter, STOP; otherwise continue.

Step 5. If Iter > 20 let Iter $= 1, \lambda = \lambda^*,\ d = d/2$, otherwise calculate step size t_k and update multipliers λ_{ijk}.

Step 6. If $\lambda_{ijk} < 0$, then set $\lambda_{ijk} = 0$. Go to Step 2.

4.2. Obtaining a feasible solution

We propose the following three different processes to find the feasible solution: SIMPLE, INDIVIDUAL, and INTERACT.

- SIMPLE process makes a straightforward modification to the solution of problem [P3], resulting in no changes in ship-to-berth assignments as well as ship-to-order assignments at each berth. It attempts to find the first "unsatisfied ship" in ascending order of service, that is serviced before its arrival in the solution to [P3]. If an unsatisfied ship is found, then the service is postponed until its arrival time, thereby resulting in the corresponding delay in service of its successors. This process is repeated until all the ships at each berth satisfy the relaxed constraints. In essence, this process finds a feasible solution but without any changes in order of service in the solution to problem [P3].
- INDIVIDUAL process first constructs a feasible assignment for each berth by changing the starting time of service for unsatisfied ships as done in SIMPLE process. Then, if there is an idle time of the berth between two adjacent services of ships, a ship being scheduled for later service is shifted in the idle time as long as it is serviced after its arrival. In this process no ships are swapped between berths.

  For the above procedure, we select a ship (ship j) with the minimum value of W_j, where W_j given by Eq. (26) is potential time of wait for ship j. We will discuss this selection principle later.

$$W_j = \sum_{j' \in L^U} (D_j - A_{j'}), \tag{26}$$

  where L^U is a set of ships consisting of unsatisfied ship U and its successors except for ship j, and D_j is the scheduled departure time of ship j.

  A formal process is as follows:

  Step 1. Select a berth (berth i). If all berths were examined, STOP.

  Step 2. Select in ascending order of position in the service sequence a ship (ship U) being serviced at berth i earlier than its arrival. If the previous ship U is the last ship serviced at berth i, go to Step 1.

  Step 3. Change the start time of service for ship U such that its service begins no earlier than its arrival.

  Step 4. Select a ship (ship j) among a set (set R) consisting of the successors of ship U in ascending order of position in the sequence of service. If the previous ship j is the last ship serviced at berth i or there exists no idle time between services of ship U and its immediate predecessor, go to Step 10.

  Step 5. Compute $W_j = \sum_{j' \in L^U} (D_j - A_{j'})$.

  Step 6. Select the next ship j among set R. If the previous ship j is the last ship serviced at the berth, then go to Step 7, otherwise go to Step 5.

  Step 7. Identify a ship ($\in L^U$) with minimum W_j that can be shifted immediately before ship U while its service begins after arrival. If no such a ship exists, go to Step 10.

  Step 8. Shift the ship just before ship U.

  Step 9. Go to Step 4.

  Step 10. Compute new start times of service for all the successors of ship U. Go to Step 2.

When changing the position of the service sequence for ship j, the final assignments for the successors of ship U are unknown. However, maintaining their present position is desirable. This is because the present position is part of the optimal solution to [P3] and good assignment if the successors arrive before their scheduled service. If their arrivals are later than the departure of ship j in the revised sequence of service, its W_j is small. The selection of ship j with minimum W_j likely yields another idle time of the berth in the new sequence of service. We could lower the objective function value by shifting as many ships as possible in the idle time of the berth. If the successors of ship U arrive before the departure of ship j, W_j should be large. This situation will likely make them wait rather long for service. The above reasoning justifies choosing a ship with minimum W_j.

We examine the unsatisfied ships in ascending order of position in the service sequence. If we do this in reverse order, after processing one unsatisfied ship all preceding unsatisfied ships still remain unsatisfied. However, in our process every time we process an unsatisfied ship, we recalculate the start times of service for all of its successors. This may make unsatisfied ships among them become "satisfied ships".

- INTERACT process is the same as INDIVIDUAL but with an additional process. If there are still some idle times of a berth after the INDIVIDUAL process, they are retried, by the additional process, to be used for the service of ships scheduled at other berths. The way to find a candidate for berth shifting is different from INDIVIDUAL for significant savings in computational time. That is, selecting ships from the set V (the set of ships) in their arrival order, a candidate is the ship first encountered in the selecting process while it is satisfied with the following condition: the total of its staying time in port and the resulting waiting time of its successors at the new berth is less than that at the originally assigned berth. A more formal statement is as follows where E is the set of periods when the berth is idle:

Step 1. Select a berth (berth i). If all the berth were examined, STOP.

Step 2. Select in ascending order of position in the service sequence a ship (ship U) being serviced at berth i earlier than its arrival. If the previous ship U is the last ship serviced at berth i, go to Step 1.

Step 3. Perform same as Steps 3–8 in INDIVIDUAL.

Step 4. Make a set E with a period when berth i is idle between the services for ship U and its immediate predecessor in the sequence of service before shifting the ship in Step 8 of INDIVIDUAL.

Step 5. Select a ship from V in ascending order of arrival time. If all the ships are examined, go to Step 10.

Step 6. If it is a satisfied ship and assigned to berth i, go to Step 5.

Step 7. Find an idle time in E when it can be serviced after its arrival. If there exists no such an idle time, go to Step 5.

Step 8. Calculate the total of staying time for it and its successors at berth i when it is moved to berth i, and the total time at the berth where it is currently scheduled to be serviced. If the latter is no larger than the former, go to Step 5.

Step 9. Shift it to berth i and go to Step 5.

Step 10. Compute new start times of service for all the successors of ship U. Replace the idle time in E examined in Step 7 by the resulting periods split in idle condition. Go to Step 2.

5. Computational experiments

The solution procedure was coded in FORTRAN 77 on a Sun S-4/2000E workstation. Computation times reported here are in CPU seconds on this computer. Problems used in these experiments were generated randomly, but systematically. We solve problems of 5, 7, and 10 berths, each of which contains data on 25 and 50 ships, yielding 6 prototype problems. For each of these prototypes, 10 problems were generated with different seed sets for random numbers from an exponential interval of ship arrivals and those from a 2-Erlangian handling time (these distributions were obtained in our survey in the port of Kobe). Note here that the distribution of arrival time is independent from that of handling time.

The objective function of the relaxed problem [P3] is the total of waiting and handling times spent by incoming ships. Thus, in an optimal solution of [P3], ships are serviced in ascending order of their handling times at each berth. This implies that if ship handling time increases with the order of their arrivals, the feasible solution to [PD] likely gets close to optimal. From this insight, we examined two additional variations from each of the 10 problems. We hereafter call these three

Table 3
Computational results for problems A

Problem size (berths × ships)	S_i	SIMPLE			INDIVIDUAL			INTERACT		
		Gap[a] (%)	CPU (s)	Iteration[b]	Gap (%)	CPU (s)	Iteration	Gap (%)	CPU (s)	Iteration
5 × 25	1	37.9	82.2	200.0	27.1	50.2	200.0	34.5	51.0	200.0
	2	12.3	38.4	200.0	7.9	41.3	200.0	8.5	39.0	200.0
	3	3.4	27.5	200.0	1.7	28.3	200.0	1.7	27.1	200.0
	4	0.7	18.0	160.3	0.4	18.7	160.3	0.4	17.7	160.3
7 × 25	1	49.2	115.0	200.0	43.1	67.1	200.0	54.2	62.2	200.0
	2	13.6	56.9	200.0	11.7	50.6	200.0	14.4	64.1	200.0
	3	3.3	38.9	200.0	2.8	35.6	200.0	3.4	37.3	200.0
	4	0.5	26.7	200.0	0.3	24.2	168.1	0.3	24.4	168.1
10 × 25	1	60.3	80.1	200.0	48.3	80.0	200.0	65.5	80.0	200.0
	2	14.6	65.4	200.0	11.4	66.1	200.0	18.5	67.8	200.0
	3	3.6	45.6	200.0	3.3	45.9	200.0	5.6	46.1	200.0
	4	0.7	32.6	200.0	0.4	32.0	184.0	0.6	35.0	200.0
5 × 50	1	117.8	718.1	200.0	65.1	733.7	200.0	96.7	755.9	200.0
	2	38.5	593.4	200.0	15.1	591.4	200.0	24.4	617.7	200.0
	3	12.4	470.9	200.0	3.5	470.9	200.0	5.5	491.0	200.0
	4	2.4	291.8	200.0	0.6	292.7	200.0	0.8	306.4	200.0
7 × 50	1	139.6	979.5	200.0	99.0	981.7	200.0	141.6	1019.1	200.0
	2	33.4	798.9	200.0	21.1	804.2	200.0	30.1	841.6	200.0
	3	9.9	636.2	200.0	5.0	641.1	200.0	7.1	659.8	200.0
	4	2.0	388.4	200.0	0.8	392.5	200.0	0.9	402.1	200.0
10 × 50	1	166.3	1357.4	200.0	145.5	1373.7	200.0	219.7	1428.7	200.0
	2	32.7	1105.9	200.0	27.5	1109.3	200.0	41.6	1153.7	200.0
	3	7.9	877.1	200.0	6.5	909.0	200.0	10.3	911.3	200.0
	4	1.5	525.5	200.0	1.0	539.0	200.0	1.7	547.3	200.0

[a] Gap = (feasible solution value − lower bound) × 100/lower bound.
[b] Iteration number at which the process terminates.

Table 4
Computational results for problems B

Problem size (berths × ships)	S_i	SIMPLE			INDIVIDUAL			INTERACT		
		Gap[a] (%)	CPU (s)	Iteration[b]	Gap (%)	CPU (s)	Iteration	Gap (%)	CPU (s)	Iteration
5 × 25	1	7.1	46.2	200.0	7.1	45.7	200.0	10.8	44.8	200.0
	2	0.7	34.4	180.1	0.7	35.3	180.1	1.3	34.0	180.1
	3	0.1	9.7	53.8	0.1	8.2	50.7	0.1	8.0	50.7
	4	0.0[c]	0.1	1.0	0.0[c]	0.1	1.0	0.0[c]	0.1	1.0
7 × 25	1	19.4	59.8	200.0	19.2	62.4	200.0	27.8	74.8	200.0
	2	4.2	51.4	200.0	4.2	52.4	200.0	6.6	50.5	200.0
	3	0.5	34.3	160.3	0.5	32.1	160.3	0.9	31.9	160.3
	4	0.0[c]	0.1	1.1	0.0[c]	0.1	1.1	0.0[c]	0.1	1.1
10 × 25	1	32.0	81.4	200.0	32.0	81.9	200.0	39.9	85.2	200.0
	2	9.2	68.0	200.0	9.2	67.7	200.0	13.1	70.6	200.0
	3	2.1	48.6	200.0	2.1	48.9	200.0	4.2	51.3	200.0
	4	0.1	30.0	140.8	0.1	30.2	140.8	0.2	31.1	140.5
5 × 50	1	8.1	718.2	200.0	8.0	719.1	200.0	12.3	748.4	200.0
	2	0.2	188.6	60.7	0.2	190.5	60.7	0.2	196.1	60.7
	3	0.0[c]	1.9	1.0	0.0[c]	1.9	1.0	0.0[c]	2.0	1.0
	4	0.0[c]	1.4	1.0	0.0[c]	1.4	1.0	0.0[c]	1.5	1.0
7 × 50	1	25.6	983.0	200.0	25.4	983.3	200.0	46.4	1024.1	200.0
	2	2.9	807.2	200.0	2.9	800.9	200.0	7.5	834.9	200.0
	3	0.0	67.5	20.9	0.0	67.1	20.9	0.0	69.6	20.9
	4	0.0[c]	1.8	1.0	0.0[c]	1.8	1.0	0.0[c]	1.9	1.0
10 × 50	1	53.1	1380.2	200.0	53.0	1438.3	200.0	76.1	1500.5	200.0
	2	10.9	1146.8	200.0	10.9	1139.7	200.0	19.9	1209.1	200.0
	3	1.2	1113.3	200.0	1.1	947.5	200.0	4.4	1059.0	200.0
	4	0.0	122.4	40.8	0.0	123.3	40.8	0.0	130.2	40.8

[a] Gap = (feasible solution value − lower bound) × 100/lower bound.
[b] Iteration number at which the process terminates.
[c] 10 solutions are all optimal.

variations problems A, B, and C. In problem B, the generated handling time for a particular ship in problem A is reassigned to another ship such that it increases with ship arrival time. On the other hand, it decreases in problem C. The former may be easy to solve in a sense that a "good" solution is likely obtained while the latter is difficult.

Over all, 180 problems were tested for the experiments. The DBAP with all the ships arriving before S_i reduces to the SBAP; a solution of [P3] corresponds to an optimal solution to [PD]. This suggests that better solutions are likely obtained, when all S_is get closer to the arrival time of the last ship. Assuming S_i for each i has a unique value in the experiments, we performed four computations for each of the 180 problems with differing values of S_i.

Tables 3–5 report average gap, CPU time, and iteration number when the computation terminates for problems A, B, and C with different seed sets. In the experiments S_i was set to 1/2, 5/8, 3/4, and 7/8 of time duration between the first and last arriving ships that are indexed by 1, 2, 3, and 4 in the tables. Problems reported in Table 3 are generated with exponential ship arrival times and Erlangian handling times. Those in Table 4 are the same as ones in Table 3 but with the

Table 5
Computational results for problems C

Problem size (berths × ships)	S_i	SIMPLE			INDIVIDUAL			INTERACT		
		Gap[a] (%)	CPU (s)	Iteration[b]	Gap (%)	CPU (s)	Iteration	Gap (%)	CPU (s)	Iteration
5 × 25	1	51.3	52.6	200.0	28.0	48.8	200.0	27.2	50.6	200.0
	2	23.5	43.4	200.0	12.0	43.1	200.0	11.4	44.1	200.0
	3	8.9	32.7	200.0	4.5	33.2	200.0	4.3	32.4	200.0
	4	2.2	24.1	200.0	1.6	22.9	200.0	1.6	23.4	200.0
7 × 25	1	53.7	79.1	200.0	28.4	67.4	200.0	28.7	65.5	200.0
	2	21.4	56.1	200.0	11.6	57.2	200.0	12.0	56.4	200.0
	3	7.5	42.1	200.0	5.3	43.2	200.0	4.9	42.8	200.0
	4	2.0	32.9	200.0	1.9	33.5	200.0	1.9	33.4	200.0
10 × 25	1	54.3	87.7	200.0	30.5	88.5	200.0	33.7	88.1	200.0
	2	21.6	75.3	200.0	11.6	76.5	200.0	13.5	76.0	200.0
	3	7.7	55.1	200.0	4.1	56.4	200.0	4.4	56.5	200.0
	4	1.7	42.5	200.0	2.1	43.1	200.0	1.8	43.8	200.0
5 × 50	1	120.2	767.5	200.0	61.1	772.4	200.0	66.5	796.4	200.0
	2	52.2	646.8	200.0	25.9	650.9	200.0	30.3	668.2	200.0
	3	21.5	536.4	200.0	10.5	537.1	200.0	11.4	553.6	200.0
	4	6.2	362.1	200.0	2.9	359.7	200.0	3.2	372.5	200.0
7 × 50	1	142.5	1049.8	200.0	58.0	1050.5	200.0	73.5	1081.2	200.0
	2	57.5	874.2	200.0	27.1	884.1	200.0	31.2	897.1	200.0
	3	21.2	715.4	200.0	11.7	720.8	200.0	12.5	738.0	200.0
	4	5.4	473.1	200.0	3.1	477.7	200.0	3.4	494.7	200.0
10 × 50	1	171.2	1570.3	200.0	67.5	1508.5	200.0	88.1	1544.3	200.0
	2	57.8	1221.8	200.0	25.8	1246.4	200.0	33.6	1320.0	200.0
	3	19.1	999.2	200.0	11.7	1031.7	200.0	13.2	1091.5	200.0
	4	4.6	698.3	200.0	3.4	675.9	200.0	3.5	669.2	200.0

[a] Gap = (feasible solution value − lower bound) × 100/lower bound.
[b] Iteration number at which the process terminates.

generated handling times reassigned to other ships. Problems in Table 5 have handling times reassigned in reverse order.

The first column in Tables 3–5 indicates the problem size (i.e., numbers of berths and ships). Column 2 presents the start of the planning horizon, that is also the time when the last ship at each berth leaves in the previous planning horizon (we assume S_i for each i has the same value in the experiments). In the subsequent columns shown are the gap between the best feasible solution value (i.e., $\bar{Z}$) and the lower bound, computation time, and iteration number obtained by solution procedures using SIMPLE, INDIVIDUAL, and INTERACT processes.

First, some problems were solved with the limit of 200 iterations; however no significant solution improvement was observed through the last 100 iterations. Hence we use the limit of 200 iterations for the computational experiments. Most problems reported in these tables required the maximum 200 iterations of the subgradient optimization algorithm. This was due to the fact that objective function values for the test problems were always in millions and the subgradient optimization procedure was terminated early only if the gap between the lower bound and feasible solution value was less than 1 (i.e., the optimal solution was found).

For all three problems, INDIVIDUAL yields the smallest gaps among the three procedures, probably identifying better solutions. Although the three procedures are different in complexity, there is no significant difference in overall CPU time. The gaps obtained by INDIVIDUAL are almost less than 10% and 20%, respectively for the 25 and 50 ship problems A with $S_i > 1$, and better for problems B.

As expected, good solutions are likely identified in problems B because their gaps are smaller than those for the corresponding problems A. For all problems with $S_i = 4$ each of the three procedures identifies the optimal solutions. Further, in the 5 and 7 berth problems with $S_i = 4$ all of the procedures yield the optimal solutions. The optimal solution were mostly found at iteration 1, i.e., the first relaxed problems found it.

Problems C do not necessarily seem difficult to obtain good solutions, since some gaps are smaller than those for problems A while the others are larger. INDIVIDUAL and INTERACT yield the gap less than 30% with 25 ships.

For each size of the problem in these tables, every procedure computes a smaller gap with increasing S_i. This is because large S_is are more likely the relaxed solutions satisfying arrival restrictions. The gap is generally increasing with problem size. Noticing that the gap refers to the worst case quality of the solution, the procedure we developed seems practical.

6. Conclusions

In this paper we present a heuristic procedure for obtaining near optimal ship-berth-order assignments for the dynamic berthing plan. As stated in Section 2, the public berthing system employing the dynamic berth allocation is desired for most of Japan's container ports. The throughput for these ports highly depends on "good" berth allocation plans. It is found from the experiments conducted that the proposed algorithm is adaptable for the practical size of problems.

This paper present, given the number of berths to operate, an algorithm for planning of the ship-berth-order assignment. However, it can also be useful for decision-making on how many berths to operate. For this, given observed statistics of ship calling, one computes the DBAP with several numbers of berths and then finds, as the optimal solution, the least number of berths with its objective function value no more than the deserved value.

Acknowledgements

The authors would like to thank the three anonymous referees for their time, support and insightful suggestions. The authors, of course, assume full responsibility for the contents.

References

Bazaara, M.S., Goode, J.J., 1979. A survey of various tactics for generating Lagrangian multipliers in the context of Lagrangian duality. European Journal of Operational Research 3, 322–338.

Bean, J.C., Birge, J.R., Mittenthal, J., Noon, C.E., 1991. Matchup scheduling with multiple resources, release dates and disruptions. Operations Research 39, 470–483.

Brown, G.G., Lawphongpanich, S., Thurman, K.P., 1994. Optimizing ship berthing. Naval Research Logistics 41, 1–15.

Brown, G.G., Cormican, K.J., Lawphongpanich, S., Widdis, D.B., 1997. Optimizing submarine berthing with a persistence incentive. Naval Research Logistics 44, 301–318.

Fisher, M.L., 1981. Lagrangian relaxation method for solving integer programming problems. Management Science 27, 1–18.

Held, M., Wolfe, P., Crowder, H.P., 1974. Validation of subgradient optimization. Mathematical Programming 6, 62–88.

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

Lai, K.K., Shih, K., 1992. A study of container berth allocation. Journal of Advanced Transportation 26, 45–60.

Nishimura, E., 1998. Public use effects on the performance of container berth. Master Thesis, Kobe University of Mercantile Marine, Japan (in Japanese).

Norman, B.A., Bean, J.C., 1999. A genetic algorithm methodology for complex scheduling problems. Naval Research Logistics 46, 199–211.

Papadimitriou, C.H., Steiglitz, K., 1982. Combinatorial Optimization: Algorithms and Complexity. Prentice-Hall, Englewood Cliffs, NJ.

Pinedo, M., 1995. Scheduling: Theory, Algorithms, and Systems. Prentice-Hall, Englewood Cliffs, NJ.