Les Cahiers du GERAD ISSN: 0711–2440

A Note on Formulations of Static and Dynamic Berth Allocation Problems

Pierre Hansen
Ceyda Oğuz

G–2003–30

May 2003
Revised: November 2003

A Note on Formulations of Static and Dynamic Berth Allocation Problems

Pierre Hansen

GERAD and Department of Quantitative Methods in Management
HEC Montréal, Canada

Ceyda Oğuz

Department of Logistics
The Hong Kong Polytechnic University
Kowloon, Hong Kong SAR

May, 2003
Revised: September, 2003

Les Cahiers du GERAD

G–2003–30

Abstract

The berth allocation problem is to allocate berths (i.e., sections of the quayside) to ships arriving in a container port in order to minimize the sum of their waiting and cargo handling times. In the static case, ships are assumed to arrive before the berths become available; in the dynamic case they can arrive before or after. We discuss recent models for both problems, correct one formulation, and provide a compact reformulation for the dynamic case. The computational results for problems that are moderate in size but realistic, with up to 10 berths and 50 ships, using both formulations and CPLEX, are reported.

Keywords: berth allocation, berthing, total completion time, unrelated parallel machines, release date.

Résumé

Le problème d'ordonnancement de l'accostage consiste à allouer des sections de quais aux bateaux arrivants en un port de conteneurs de façon à minimiser la somme des temps d'attente, de déchargement et de chargement. Pour le cas statique, les bateaux sont supposés arriver avant que les sections de quais ne deviennent disponibles; dans le cas dynamique ils peuvent arriver avant ou après. On discute des modèles récents pour les deux problèmes, corrige une formulation et présente une reformulation compacte pour le cas dynamique. Des résultats de calculs sont présentés pour des instances de taille modérée mais néanmoins réaliste, avec jusqu'à 10 sections de quais et 50 bateaux; on utilise les deux formulations et CPLEX.

Mots clés : ordonnancement, accostage, temps total d'exécution, machines parallèles non corrélées, date de début.

1 Introduction

The berth allocation problem is to allocate berths (i.e., sections of the quayside) to ships arriving in a container port in order to minimize the sum of their waiting and cargo handling times. As container transportation grows rapidly, and ships spend a large amount of their time in port at the cost of several thousand US dollars per hour (Peterkovsky and Daganzo, 1990; Ward, 2002), this problem is of major importance in port operations (Meersmans and Dekker, 2001). It has recently been studied using mathematical programming and genetic search by Imai *et al.* (1997, 2001), Lim (1998), Nishimura *et al.* (2001), and Park and Kim (2002). With the exception of the subgradient method of this last paper, applied to small problems with up to seven ships, only heuristics are available for the dynamic case in which ships arrive before or after berths become available.

The paper is organized as follows. The model for the static berth allocation problem of Imai *et al.* (1997, 2001), where all ships are assumed to have reached the port before the berths become available, is recalled in the next section. A correction is then made in the expression of the objective function of Imai *et al.* (2001). An extension of the dynamic berth allocation problem (Imai *et al.*, 2001) is discussed in Section 3. A compact reformulation is proposed in Section 4, together with some further extensions: taking into account safety constraints due to a ship's length or draft, the due dates for the departures of some or all ships and a bounded planning horizon. Computational results for problems that are moderate in size but realistic, with up to 10 berths and 50 ships, using both formulations of the dynamic berth allocation problem and CPLEX, are presented in Section 5. Brief conclusions are given in the last section.

2 The static berth allocation problem

Let us assume that:

(a) A set B of berths, indexed by $i = 1, 2, \ldots, I$, is available in a container port for receiving ships, from time S_i, $i = 1, 2, \ldots, I$, respectively onward;

(b) A set V of ships, indexed by $j = 1, 2, \ldots, T$, gets to the port at arrival times A_j, $j = 1, 2, \ldots, T$, respectively, and may have to wait before being handled;

(c) Each berth can handle one ship at a time or remain idle for some time;

(d) Any ship j can be handled at any berth i, with a given handling time c_{ij}, $i = 1, 2, \ldots, I$, $j = 1, 2, \ldots, T$, depending on both the ship and the berth;

(e) The total completion time, i.e., waiting and handling time for all ships is to be minimized;

(f) All ships arrive before any berth becomes available, i.e.,

$$\max_j A_j \leq \min_i S_i. \tag{1}$$

Due to this last assumption, all berths will handle ships consecutively; i.e., without idle time between ships. If this assumption does not hold, the possible idle time must be taken into account and the dynamic berth allocation problem is obtained. Observe that the model is short-term and deterministic. Moreover, fractional assignments are excluded; i.e., the uncommon practice of changing a ship's berth while unloading or loading it is prohibited.

Consider, then, the reverse order in which ships j are handled at berth i. To that effect, introduce binary variable x_{ijk}, such that

$$x_{ijk} = \begin{cases} 1 & \text{if ship } j \text{ is the } k\text{th last to be handled at berth } i, \\ 0 & \text{otherwise.} \end{cases}$$

At each berth, let us assign the reverse order at which ships will be handled with an index k such that $k \in O = \{1, 2, \dots, T\}$. Notice that this reverse order is unknown. It is then easy to see that the total completion time of all ships handled at berth i will be

$$\sum_{j \in V} \sum_{k \in O} (S_i - A_j) x_{ijk} + \sum_{j \in V} \sum_{k \in O} k c_{ij} x_{ijk} \tag{2}$$

where the first term corresponds to the waiting time for all ships handled there before berth i becomes available, and the second term is the sum for all such ships of the handling time and waiting time of the $k-1$ ships still to be handled while the kth ship is being attended to. This is illustrated in Figure 1, where W_j denotes the waiting period and H_j the handling period for ship j, $j = 1, 2, \dots, T$, and TCT denotes total completion times. Note that ships may be indexed in the non-decreasing order of their arrival times, but this is not mandatory.

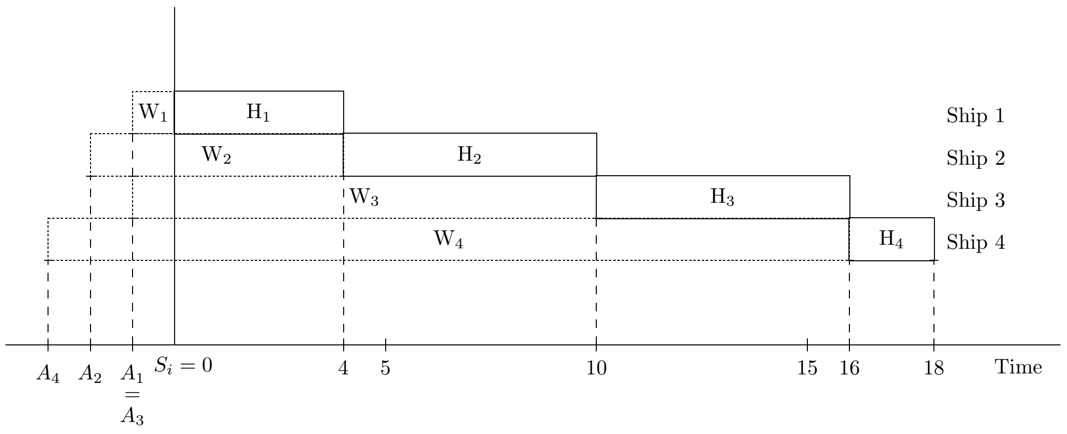

Figure 1: Example of ships handled at berth i for SBAP: $A_1 = -1$, $A_2 = -2$, $A_3 = -1$, $A_4 = -3$, $W_1 = 1$, $H_1 = 4$, $W_2 = 6$, $H_2 = 6$, $W_3 = 11$, $H_3 = 6$, $W_4 = 19$, $H_4 = 2$, TCT=55.

The static berth allocation problem can be expressed as follows, which is done as part of a more general model in Imai *et al.* (1997).

$$\text{(SBAP)} \qquad \text{Minimize} \sum_{i \in B} \sum_{j \in V} \sum_{k \in O} (k c_{ij} + S_i - A_j) x_{ijk} \tag{3}$$

subject to

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

$$\sum_{j\in V} x_{ijk} \leq \quad 1 \qquad \forall i \in B, k \in O, \tag{5}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in B, j \in V, k \in O. \tag{6}$$

Observe that as the coefficients of the variables x_{ijk} in the objective function (3) decrease when k increases, the ships will be assigned to consecutive positions at each berth. Note also that ships will be processed at each berth in increasing order of the c_{ij} in the optimal solution. Replacing the pairs of indices i and k by the new indices $n = 1, 2, \ldots, I \times T$, noting by N the resulting index set and setting

$$D_{jn} = kc_{ij} + S_i - A_j \quad \forall j \in V, n = (i-1)T + k \in N \qquad \forall i \in B,\ k \in O \tag{7}$$

gives the new expression

$$(\text{SBAP}') \qquad \text{Minimize} \sum_{j\in V}\sum_{n\in N} D_{jn}x_{jn} \tag{8}$$

subject to

$$\sum_{n\in N} x_{jn} = \quad 1 \qquad \forall j \in V, \tag{9}$$

$$\sum_{j\in V} x_{jn} \leq \quad 1 \qquad \forall n \in N, \tag{10}$$

$$x_{jn} \in \quad \{0,1\} \qquad \forall j \in V, n \in N, \tag{11}$$

which is a two-dimensional assignment problem and easily solved by for example, the polynomial algorithm of Jonker and Volgenant (1987) or the "auction" algorithm of Bertsekas (1992).

As already noted by Imai *et al.* (1997), SBAP is a particular case of a known scheduling problem; i.e., minimizing the mean completion time for independent tasks on unrelated parallel machines. A solution similar to the one given above is presented in Bruno *et al.* (1974).

In Imai *et al.* (2001) a variant of the SBAP model is proposed: instead of interpreting variables x_{ijk} equal to 1 as indicating that the ship j is handled at berth i in the kth last position, they are considered to indicate that this ship is handled as the kth ship at berth i. We will use x'_{ijk} in this case.

The following model is presented:

$$\text{Minimize} \sum_{i\in B}\sum_{j\in V}\sum_{k\in O}\{(T-k+1)c_{ij} + S_i - A_j\}x'_{ijk} \tag{12}$$

subject to constraints (4) to (6) with x'_{ijk} instead of x_{ijk}. This return to the order in time instead of the reverse order of handling ships at berths presents two difficulties. First, the number T_i of ships handled at berth i should be used in (12) instead of T for all i, and is not known beforehand. It can be determined by using further equations, i.e.,

$$T_i = \sum_{j \in V} \sum_{k \in O} x'_{ijk} \qquad \forall i \in B, \tag{13}$$

but then the model becomes non-linear in objective.

Second, while in model (3)–(6) the coefficients of variables x_{ijk} *increase* with k for a fixed i and j, which ensures that a ship cannot be handled at a given berth in the kth last position unless another ship is handled in the $(k-1)$st position at that berth, in model (12), (4)–(6) the coefficients of variables x'_{ijk} *decrease* when k increases. This implies that constraints (4) to (6) are not sufficient to ensure that a ship will not be handled at a given berth in the kth first position while no ship is handled in the $(k-1)$st first position at that berth in the solution. Additional constraints are then needed and can be written as

$$\sum_{j \in V} x'_{ijk} \leq \sum_{j \in V} x'_{ij,k-1} \qquad \forall i \in B, k \in O \setminus \{1\}. \tag{14}$$

Then if no ship is handled in the $(k-1)$st position at berth i, the right-hand side is equal to zero, which implies the left-hand side is also null, and that no ship is handled in the kth position at that berth. Again, it is possible to reduce the problem to a two-dimensional one. Let n be defined as above. Then, taking

$$D'_{jk} = (T-k+1)c_{ij} + S_i - A_j \quad \forall j \in V, n = (i-1)T + k \in N \tag{15}$$

one gets

$$\text{(SBAP}''\text{)} \qquad \text{Minimize} \sum_{j \in V} \sum_{n \in N} D'_{jn} x'_{jn} \tag{16}$$

subject to

$$\sum_{n \in N} x'_{jn} = 1 \qquad \forall j \in V, \tag{17}$$

$$\sum_{j \in V} x'_{jn} \leq 1 \qquad \forall n \in N, \tag{18}$$

$$\sum_{j \in V} x'_{jn} \leq \sum_{j \in V} x'_{j,n-1} \qquad \forall n \in N, n \bmod (T) \notin \{1\}, \tag{19}$$

$$x'_{jn} \in \{0,1\} \qquad \forall j \in V, n \in N. \tag{20}$$

where $n \bmod (T)$ denotes the remainder of n after division by T.

However, the first difficulty remains. It is not innocuous, as it may affect the optimal solution and not only its value. To illustrate, consider the following three ships and two

Table 1: Handling times

	Ship		
Berth	1	2	3
1	10	10	10
2	11	11	11

berths problem, with $S_i = 0$ for $i = 1, 2$ and $A_j = 0$ for $j = 1, 2, 3$. Handling times are given in Table 1.

Coefficients D_{jn} of model (8)–(11) are given in Table 2. The optimal solution will be $x^*_{111} = x^*_{122} = x^*_{231} = 1$, and the other $x^*_{ijk} = 0$ (or equivalently $x^*_{11} = x^*_{22} = x^*_{34} = 1$, and the other $x^*_{jn} = 0$), with a total completion time of $10 + 2 \times 10 + 11 = 41$.

Table 2: Coefficients D_{jn} of variables x_{jn} in model (8)–(11)

	Berth		1			2	
Ship	Berth×position (reverse)	1	2	3	4	5	6
1		$\boxed{10}$	20	30	11	22	33
2		10	$\boxed{20}$	30	11	22	33
3		10	20	30	$\boxed{11}$	22	33

Coefficients D'_{jn} of model (16)–(20) are given in Table 3. With this model, the optimal solution will be $x'^*_{111} = x'^*_{122} = x'^*_{133} = 1$, and the other $x'^*_{ijk} = 0$ (or equivalently $x'^*_{11} = x'^*_{22} = x'^*_{33} = 1$, and the other $x'^*_{jn} = 0$). The value of this solution is $3 \times 10 + 2 \times 10 + 10 = 60$, which is not optimal.

Table 3: Coefficients D'_{jn} of variables x'_{jn} in model (16)–(20)

	Berth		1			2	
Ship	Berth×position (direct)	1	2	3	4	5	6
1		$\boxed{30}$	20	10	33	22	11
2		30	$\boxed{20}$	10	33	22	11
3		30	20	$\boxed{10}$	33	22	11

3 The dynamic berth allocation problem

If assumption (f) of the SBAP is relaxed, i.e., some ships may arrive after some or all berths become available, the *dynamic berth allocation problem* (DBAP) arises. This problem is NP-hard even if there is a single berth. Indeed, it then reduces to *minimizing the total* *completion time with release dates on a single machine.* This last problem has been shown to be NP-hard by Lenstra *et al.* (1977). As a consequence, the *first come, first served* rule, which may be adopted for reasons of fairness, i.e., to avoid overpassing between ships, is not optimal for the criterion of total completion time (contrary to what was stated in a recent survey on the transshipment of containers, by Vis and de Koster, 2003, p3). This is easily seen with the following one berth, two ships example: Let $A_1 = 0$, $A_2 = 1$, $c_{11} = 10$, $c_{12} = 1$ and $S_1 = 0$. If ship 1 is handled first, from $s_{11} = 0$ onwards, ship 2 is handled from $s_{12} = 10$, and the total completion time is $10 + 9 + 1 = 20$. If the second ship is handled first at $s_{11} = 1$, and the first one at $s_{12} = 2$, the total completion time is $1 + 2 + 10 = 13$ (see Figure 2). This example also shows another counter-intuitive fact: even if a ship has already arrived at the port, it may be better to keep the berth idle for some time in order to minimize the total completion time, than to handle it immediately (when a small ship will arrive soon).

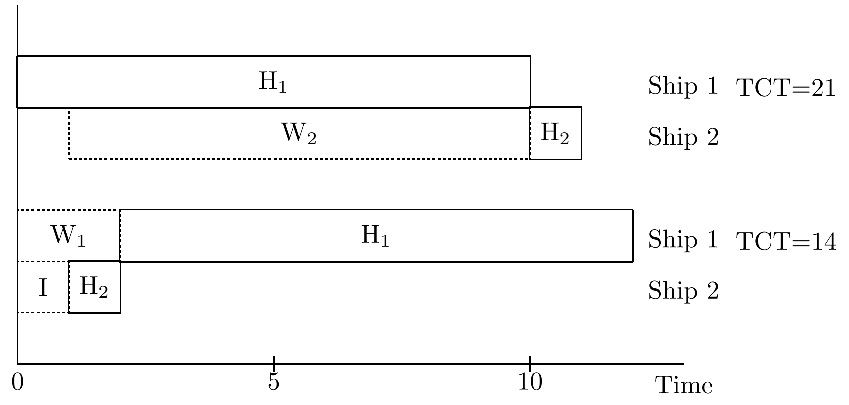

Figure 2: An example of DBAP, where overpassing is beneficial and the berth may be kept idle even if a ship is waiting (which is noted by I).

If $A_j > S_i$ for some ships and berths, the possibility of idle periods at some berths must be taken into account. This is done in Imai *et al.* (2001) through the introduction of additional continuous variables y_{ijk} representing the length of the idle period at berth i before the arrival of ship j, which will be handled in kth position. Constraints for giving the right values to these variables are also added.

As the same error as in the SBAP model is made in the DBAP model of Imai *et al.* (2001) (and also in a recent paper by the same authors, i.e., Imai *et al.*, 2003), we present this model after due corrections have been made. Therefore, we again use the reverse order for the position of the ships at a berth in contrast to Imai *et al.* (2001), who attempted to use the direct order. The continuous variables y_{ijk} then represent the length of the idle period at berth i before the arrival of ship j, which will be handled in the kth last position.

The corrected DBAP model is the following:

$$\text{(DBAP)} \qquad \text{Minimize} \sum_{i\in B}\sum_{j\in V}\sum_{k\in O}(kc_{ij}+S_i-A_j)x_{ijk}+\sum_{i\in B}\sum_{j\in V}\sum_{k\in O}ky_{ijk} \tag{21}$$

subject to

$$\sum_{i\in B}\sum_{k\in O}x_{ijk} = 1 \qquad \forall j\in V, \tag{22}$$

$$\sum_{j\in V}x_{ijk} \le 1 \qquad \forall i\in B, k\in O, \tag{23}$$

$$\sum_{l\in V}\sum_{m\in P_k}(c_{il}x_{ilm}+y_{ilm})+y_{ijk}-(A_j-S_i)x_{ijk} \ge 0 \qquad \forall i\in B, j\in W_i, k\in O, \tag{24}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i\in B, j\in V, k\in O, \tag{25}$$

$$y_{ijk} \ge 0 \qquad \forall i\in B, j\in V, k\in O. \tag{26}$$

where $P_k=\{m\in O: m>k\}$ and $W_i=\{j\in V: A_j>S_i\}$.

The total completion time of all ships handled at berth i is

$$\sum_{j\in V}\sum_{k\in O}\{(kc_{ij}+S_i-A_j)x_{ijk}+ky_{ijk}\}. \tag{27}$$

This is illustrated in Figure 3.

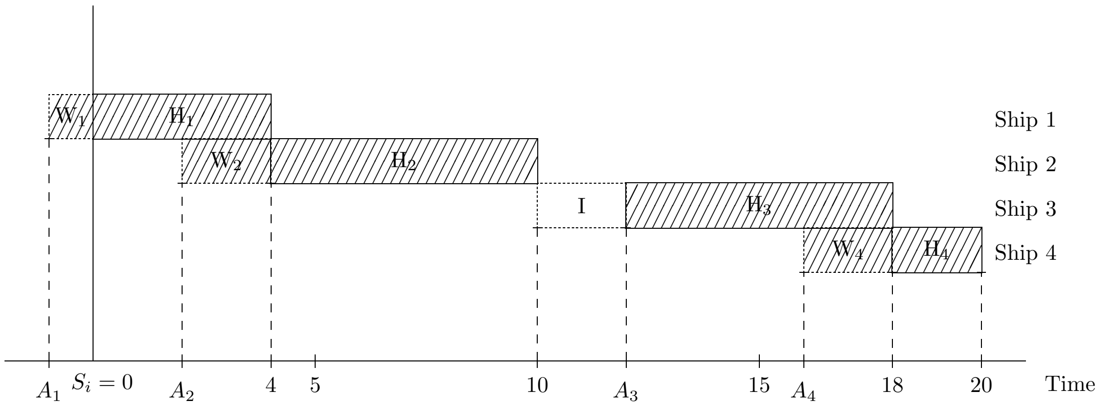

Figure 3: Example of ship handling at berth i for DBAP: $A_1=-1$, $A_2=2$, $A_3=12$, $A_4=16$, $W_1=1$, $H_1=4$, $W_2=2$, $H_2=6$, $W_3=0$, $H_3=6$, $W_4=2$, $H_4=2$, TCT=23.

When a ship arrives at this berth, it may either (i) have to wait, which is noted by W_j, as $A_j<S_i$ (e.g., for $j=1$ in Figure 3) or as the berth is occupied (e.g., for $j=2$ in Figure 3) or because a ship with a short handling time will be arriving soon, or (ii) be

processed immediately as the berth is unoccupied. In this last case, an idle period for the berth (which is noted by I and may be equal to 0) immediately precedes the handling of the ship (e.g., for $j = 3$ in Figure 3). The total completion time for the ships handled at berth i, represented by the shaded area in Figure 3, is given by (27) and may be explained as follows:

Let

$$j_i^*(k) = j \in V | x_{ijk} = 1 \tag{28}$$

denote the index of the ship handled in the kth last position at berth i for $k = 1, 2, \ldots, T_i$. Moreover, let $O_i = \{1, 2, \ldots, T_i\}$ for all $i \in B$. Consider first the sum for all such ships that arrive before S_i of their waiting time until berth i becomes available, where $\bar{W}_i = \{j \in V : A_j > S_i\}$:

$$\sum_{k \in O_i} \sum_{j_i^*(k) \in V \setminus \bar{W}_i} (S_i - A_{j_i^*(k)}). \tag{29}$$

This sum is represented by the areas of all shaded rectangles to the left of the vertical line passing through S_i in Figure 3. Then, focus on the sum for all ships handled at i of the difference between their departure time and S_i (which includes waiting, handling, idle and those periods before a ship arrives):

$$\sum_{k \in O_i} \sum_{j_i^*(k) \in V} (d_{j_i^*(k)} - S_i) \tag{30}$$

where the departure time $d_{j_i^*(k)}$ corresponds to the abscissa of the end of a horizontal segment in the upper staircase line of Figure 3, and (30) corresponds to the area below that line and to the right of S_i in that figure. Partitioning this area into rectangles bounded by the abscissae axis, the upper staircase line and the vertical lines at the abscissae corresponding to the successive arrival and departure times shows that (30) is equal to

$$\sum_{k \in O_i} k(y_{ij_i^*(k)k} + c_{ij_i^*(k)}). \tag{31}$$

Summing (29) and (31) gives an overestimate of the total completion time, as ships with $j \in \bar{W}_i$ arrive after S_i. A correction of

$$\sum_{k \in O_i} (A_{j_i^*(k)} - S_i) \tag{32}$$

must therefore be subtracted from this sum. It corresponds to the area bounded by the vertical line through S_i, the abscissae axis and a staircase line immediately below and to the right of each of the shaded rectangles corresponding to handling a ship at berth i in Figure 3.

Summing then gives

$$\sum_{k\in O_i}\left\{\sum_{j\in V}(S_i - A_{j_i^*(k)}) + k(y_{ij_i^*(k)k} + c_{ij_i^*(k)})\right\} \tag{33}$$

which is the same as (27) after the decision variables are fixed.

The constraints of the DBAP model will be the same as those of the SBAP model, except that we have to define idle periods at berth i as a constraint in DBAP. The constraints (24) that define idle periods at berth i are easier to explain. When the kth last ship to be handled arrives, the time during which berth i has been available is

$$A_{j^*(k)} - S_i \tag{34}$$

and it has been used or idle until the departure of the ship handled at the $(k+1)$st last position, for a time of 0 if $k = T_i$ and

$$\sum_{m\in O_i, m>k}(y_{ij_i^*(m)m} + c_{ij_i^*(m)}) \tag{35}$$

otherwise.

Subtracting (35) from (34) corresponds to the constraints (24) on the length y_{ijk} of the idle period before handling the kth last ship, after the decision variables have been fixed.

Unfortunately, the size of problem (21)–(26) rapidly increases with the number of berths and ships considered. There are $IJT = |B||V||O|$ binary variables x_{ijk} and as many continuous variables y_{ijk}, J constraints (22), IT constraints (23) and IJT constraints (24) (at most, as some $j \in V \setminus W_i$ for some i and then no idle time y_{ijk} occurs). For a realistic example with $I = 10$ and $J = 50$, as used in Imai *et al.* (2001), there are 25,000 0-1 variables, 25,000 continuous ones, 50 constraints (22), 500 constraints (23) and 25,000 constraints (24).

Such numbers are large for a mixed-integer program, even if constraints (22) and (23) are multiple-choice ones, which can be efficiently exploited in algorithms for mixed-integer programming with *special ordered sets* (Forrest *et al.*, 1974) as implemented in most standard packages. In the next section we will present an equivalent, more compact, formulation. In order to take advantage of the assignment-type structure of constraints (22) and (23), as in the SBAP, Imai *et al.* (2001) moved constraints (24) into the objective function using Lagrangian relaxation. Then, for the y_{ijk} only the lower bounds of 0 remained in the constraint set. This implies that these variables will be equal to 0 at the optimum if their coefficient in the Lagrangian relaxation is positive or to ∞ if it is negative (indicating infeasibility). Imai *et al.* (2001) set all y_{ijk} at 0. The resulting bounds, obtained with subgradient optimization, are therefore not tight unless there is no idle period at any berth. This happens in rare cases, in which it is assumed that the vast majority of ships arrive at the port before some or all of the berths become available. Most of the constraints (24) then disappear.

Some further modifications can be brought to DBAP in order to take additional realistic restrictions into account.

1. *Ship size constraints:* Some berths i may not be able to handle some ships j because of the excessive length or draft of the ships; it suffices to set $x_{ijk} = 0$ for such pairs (i, j) and all $k \in O$ to express that. Alternately, one could use very large c_{ij} to impose $x_{ijk} = 0$.
2. *Due dates:* It may be unwise to have some ships wait for long periods of time while others that arrive later are processed, even if this is required by the schedule to minimize the total completion time. Due dates may also be imposed by contracts. This can be taken into account by introducing due date constraints, i.e., times L_j before or at which the ships j must have left or leave the port. Such constraints take the following form:

$$S_i + \sum_{l \in V} \sum_{m \in P_k} (c_{il} x_{ilm} + y_{ilm}) + y_{ijk} + c_{ij} x_{ijk} \le L_j \quad \forall i \in B, j \in W_i, k \in O. \tag{36}$$

   Indeed, the time at berth i is the time S_i when it becomes available, plus the times for handling ships there, with an index larger than or equal to k in the reverse handling order, plus the waiting times before handling them, and it should not be larger than L_j. When it is the case, (36) imposes $x_{ijk} = 0$.
3. *Horizon:* As new information about arriving ships becomes available over time and there are some uncertainties, it appears to be realistic not to plan too much in advance, i.e., to impose a horizon H before or at which all ships considered will have been handled. It then suffices to take $L'_j = \min\{L_j, H\}$ instead of L_j in (36). Note that imposing a fairly close horizon will also spread the work load among berths more equally, which could be a desirable secondary objective. By the same token, one might wish to roughly equalize the number of ships handled at each berth; e.g., limit the maximum number of ships handled at one berth to two or three times the average. When this is the case, large values of k need not be considered anymore, and the size of the model is further reduced, both in terms of variables and of equations.

4 A compact reformulation of DBAP

A close look at constraints (24) and Figure 3 suggests a possible way of obtaining an equivalent formulation of DBAP more compact than (21)–(26). This requires new variables s_{ik}, defined as the starting time for handling the kth ship at berth i, for $i \in B$, $k \in O$; note that here we use the order of handling ships in time and not the reverse order. As above, we use variables x'_{ijk} instead of x_{ijk} to indicate this. Binary variables z_{ik} are also needed for $i \in B$, $k \in O$ to indicate whether a kth ship will be handled at berth i ($z_{ik} = 0$) or not ($z_{ik} = 1$). Finally, large constants M_{ik} for $i \in B$, $k \in O$ are introduced to allow s_{ik} to be equal to 0 if no kth ship is handled at the ith berth. The model is as follows:

$$(\text{DBAP}') \qquad \text{Minimize} \sum_{i \in B} \sum_{j \in V} \sum_{k \in O} (c_{ij} - A_j) x'_{ijk} + \sum_{i \in B} \sum_{k \in O} s_{ik} \tag{37}$$

subject to

$$\sum_{i\in B}\sum_{k\in O} x'_{ijk} = 1 \qquad \forall j \in V, \tag{38}$$

$$\sum_{j\in V} x'_{ijk} + z_{ik} = 1 \qquad \forall i \in B, k \in O, \tag{39}$$

$$z_{ik} \geq z_{i,k-1} \qquad \forall i \in B, k \in O \setminus \{1\}, \tag{40}$$

$$s_{ik} \geq \sum_{j\in V} A_j x'_{ijk} \qquad \forall i \in B, k \in O, \tag{41}$$

$$s_{ik} \geq s_{i,k-1} + \sum_{j\in V} c_{ij} x'_{ij,k-1} - M_{ik} z_{ik} \qquad \forall i \in B, k \in O \setminus \{1\}, \tag{42}$$

$$s_{i1} \geq S_i - S_i z_{i1} \qquad \forall i \in B, \tag{43}$$

$$s_{ik} \geq 0 \qquad \forall i \in B, k \in O \setminus \{1\}, \tag{44}$$

$$x'_{ijk} \in \{0,1\} \qquad \forall i \in B, j \in V, k \in O, \tag{45}$$

$$z_{ik} \in \{0,1\} \qquad \forall i \in B, k \in O, \tag{46}$$

where M_{ik} is equal to the sum of the $(k-1)$ largest c_{ij}, plus the largest of the maximum arrival time of the ships and the time berth i becomes available. Problem DBAP$'$ has IJT binary variables x'_{ijk} as DBAP, but only IT continuous variables s_{ik} instead of IJT variables y_{ijk}, and IT additional binary variables z_{ik}. Constraints (38) and (39) of DBAP$'$ are the same as (22) and (23) of DBAP, except that variables x'_{ijk} and x_{ijk} refer to the direct and reverse handling order of ships at berths, respectively. Constraints (40) express that one cannot handle a ship in the kth position for $k \geq 2$ at a berth i unless there is a ship handled in the $(k-1)$st position there. They are equivalent to constraints (19) of SBAP$''$, but more compact.

There are $2I(T-1)$ constraints (41) and (42) in DBAP$'$, which define variables s_{ik}, instead of the much more numerous constraints (24) of DBAP, which define variables y_{ijk}. Constraints (41) express that a ship cannot be handled, in the kth position, before it arrives at the port. Constraints (42) specify that if a kth ship is handled at berth i, this cannot begin at s_{ik}, before the time $s_{i,k-1}$ at which the handling of the previous ship began, plus the time c_{ij} required by that operation. For $I = 10$, $|V| = 50$, as considered in Imai et al. (2001) and above, DBAP$'$ has 25,500 binary variables x'_{ijk} and z_{ik}, 500 continuous variables s_{ik} and 50+500+980=1,530 constraints, so it is much more compact than DBAP in terms of constraints. Unfortunately, constraints (42) contain "big M" coefficients which are known to slow down solutions.

As in the DBAP model, additional constraints may be added to express further requirements. Ship size constraints are treated in the same way as above, setting some variables $x'_{ijk} = 0$. Alternately, one could use very large c_{ij} to impose $x'_{ijk} = 0$ (but then one should avoid using such c_{ij} in the computation of the M_{ik}'s). Due date constraints can be expressed in a more compact way than in DBAP. They take the following form:

$$\sum_{i \in B} \sum_{k \in O} (s_{ik} + c_{ij}) x'_{ijk} \leq L_j \quad \forall j \in V, \tag{47}$$

and there are J of them, which is small. Due to constraints (38), only one variable x'_{ijk} in (47) will be equal to 1 in the optimal solution. This equation then ensures that the corresponding departure date $s_{ik} + c_{ik}$ does not exceed L_j. Finally, the horizon constraints are treated again in DBAP$'$ as in DBAP.

5 Computational experience

The systematic testing of models DBAP and DBAP$'$ has been conducted using CPLEX-MIP on a 600MHz PC computer. First, problems have been generated as in Imai *et al.* (2001). They consist of three series with the following characteristics:

1. Number of berths, $|B| = 5, 10$.
2. Number of ships, $|V| = 10, 15, 20, 25, 30$.
3. Arrival times of the ships, A_j, are from a uniform distribution in the range of $[1, (7000/60)(|V|/|B|)]$.
4. Handling times, $c_{ij} = (2 * u_{ij} + 1.5) * 2000/60$, where u_{ij} is a random number from the uniform distribution between 0 and 1.
5. Availability times of the berths, S_i = 1/2, 3/5, 5/8, 7/8, of the time interval between the arrivals of the first and last ships.

We generated three problem instances for each combination of the data above and set, for the larger instances, a maximum of 1 hour of CPU time to stop the algorithm when solving both DBAP and DBAP$'$ during our computational experiments. The results are presented in Tables 4-7. If the 1 hour CPU time limit was exhausted for all three problem instances, we denoted that with the entry > 3600 in the tables. Furthermore, if only one or two out of three problem instances could be solved within the 1 hour CPU limit, we denoted it with (1) and (2), respectively, next to the CPU times.

It appears that DBAP is capable of solving all moderate-sized problems, with up to 10 berths and 30 ships. Furthermore, DBAP is able to solve most of the larger problems, with up to 10 berths and 50 ships, within reasonable CPU times. We can observe from Tables 4 and 5 that DBAP obtains the optimum solution in a shorter time when the number of ships waiting before a berth becomes available increases. We note that this becomes more apparent for larger problem instances; in a 5-berth case, when the number of ships is greater than or equal to 40, and in a 10-berth case when the number of ships is greater than or equal to 35.

Imai *et al.* (2001) reported heuristic solutions for similar problems with times of up to about 1500 seconds on a SUN S4/2000E workstation. For 10 berths and 50 ships, the gap of these solutions defined as (feasible solution value – lower bound) / lower bound is in the range of 0 to 2% for $S_i = 7/8$, but 50 to 200% for $S_i = 1/2$.

Table 4: Minimum, average and maximum CPU times of DBAP for different numbers of berths and ships.

			CPU time (sec)		
Berth	Ship	S_i	Minimum	Average	Maximum
5	10	1/2	0.8900	2.4067	5.3200
		3/5	0.8900	4.5767	11.1100
		5/8	0.1900	2.2633	6.3000
		7/8	0.0700	0.0867	0.1200
5	15	1/2	2.3700	7.9433	12.3600
		3/5	1.0000	1.9300	2.6200
		5/8	1.1100	1.5800	1.8900
		7/8	0.5100	0.5200	0.5300
5	20	1/2	14.6700	56.1600	128.1800
		3/5	4.2100	14.0200	19.7999
		5/8	3.8500	10.5933	17.1500
		7/8	1.1800	1.9500	2.3400
5	25	1/2	109.3900	2330.1501	6474.1802
		3/5	14.5200	98.2700	224.8999
		5/8	11.1600	162.8533	451.4400
		7/8	5.3400	8.8500	13.1500
5	30	1/2	4123.6201	25445.6963	60419.5391
		3/5	157.3800	427.2233	799.8599
		5/8	41.3800	49.5633	55.4199
		7/8	103.7699	145.4533	215.4600
5	35	1/2	778.5300	2690.6033	3657.5400
		3/5	180.4500	1197.8600	3042.2300
		5/8	77.9500	520.3333	1324.1200
		7/8	28.5600	33.9633	37.7400
5	40	1/2	>3600	>3600	>3600
		3/5	2820.6799 (1)	2820.6799 (1)	2820.6799 (1)
		5/8	1201.2300 (2)	1357.8749 (2)	1514.5500 (2)
		7/8	31.7200	68.5167	136.1400
5	45	1/2	>3600	>3600	>3600
		3/5	1047.3000 (1)	1047.3000 (1)	1047.3000 (1)
		5/8	883.6500 (2)	2146.7700 (2)	3409.8899 (2)
		7/8	57.5900	117.3500	225.0300
5	50	1/2	>3600	>3600	>3600
		3/5	>3600	>3600	>3600
		5/8	1633.5100 (1)	1633.5100 (1)	1633.5100 (1)
		7/8	182.0000	241.2667	302.3600

Table 5: Minimum, average and maximum CPU times of DBAP for different numbers of berths and ships.

			CPU time (sec)		
Berth	Ship	S_i	Minimum	Average	Maximum
10	10	1/2	0.8400	1.4600	1.8700
		3/5	0.1700	0.5567	1.2400
		5/8	0.1800	0.4900	1.0600
		7/8	0.1000	0.1133	0.1300
10	15	1/2	1.2300	6654.8632	19955.5996
		3/5	0.7400	258.3533	770.7500
		5/8	0.7600	140.4467	417.8700
		7/8	0.4400	1.0867	2.0900
10	20	1/2	38.3900	79.3033	126.5100
		3/5	11.8900	28.2067	54.7100
		5/8	8.4100	17.87	31.6000
		7/8	2.4700	2.8733	3.6300
10	25	1/2	874.3000	1833.5366	3635.6800
		3/5	91.7700	234.2200	504.3100
		5/8	74.1900	156.1567	260.4800
		7/8	10.4100	13.18	15.2100
10	30	1/2	2517.5901	3061.6934	3650.7500
		3/5	172.7100	386.8533	627.0700
		5/8	153.9300	241.7600	324.9700
		7/8	17.4700	32.0267	53.3300
10	35	1/2	>3600	>3600	>3600
		3/5	1068.8800 (1)	1068.8800 (1)	1068.8800 (1)
		5/8	572.6600	1114.0867	1979.8001
		7/8	35.9300	61.6633	113.0000
10	40	1/2	>3600	>3600	>3600
		3/5	587.7000	708.1467	785.7100
		5/8	441.7700	748.7533	1008.1600
		7/8	154.8900	180.9400	213.5600
10	45	1/2	>3600	>3600	>3600
		3/5	927.6300 (1)	927.6300 (1)	927.6300 (1)
		5/8	1007.6300 (1)	1007.6300 (1)	1007.6300 (1)
		7/8	400.9500	509.8400	711.3000
10	50	1/2	>3600	>3600	>3600
		3/5	2206.7500 (2)	2893.0750 (2)	3579.4500 (2)
		5/8	1339.5700 (1)	1339.5700 (1)	1339.5700 (1)
		7/8	492.4900	740.1467	1224.5900

We see from Tables 6 and 7 that DBAP′ obtains the optimum solution within the 1 hour CPU time limit only for small problems. However, we notice that if there are large waiting times for the ships, then DBAP′ becomes more powerful in obtaining the optimal solution in a shorter time compared to DBAP.

Table 6: Minimum, average and maximum CPU times of DBAP′ for different numbers of berths and ships.

			CPU time (sec)		
Berth	Ship	S_i	Minimum	Average	Maximum
5	10	1/2	77.3000	366.9133	682.6500
		3/5	41.3000	98.4367	190.1100
		5/8	36.5800	77.0367	129.1500
		7/8	19.9300	142.5300	341.3800
5	15	1/2	3193.1899 (1)	3193.1899 (1)	3193.1899 (1)
		3/5	1861.9000 (1)	1861.9000 (1)	1861.9000 (1)
		5/8	>3600	>3600	>3600
		7/8	387.8000 (1)	387.8000 (1)	387.8000 (1)
5	20	1/2	>3600	>3600	>3600
		3/5	>3600	>3600	>3600
		5/8	>3600	>3600	>3600
		7/8	>3600	>3600	>3600

Table 7: Minimum, average and maximum CPU times of DBAP′ for different numbers of berths and ships.

			CPU time (sec)		
Berth	Ship	S_i	Minimum	Average	Maximum
10	10	1/2	3.1700	4.6967	5.3100
		3/5	2.8000	3.5500	4.3400
		5/8	2.3900	2.5100	2.7000
		7/8	0.6900	0.7333	0.7900
10	15	1/2	>3600	>3600	>3600
		3/5	>3600	>3600	>3600
		5/8	>3600	>3600	>3600
		7/8	>3600	>3600	>3600

Then, larger problems were tested, assuming due dates L_j given by $\max\{A_j, S_i\} + r_j$, where r_j is a random number from a uniform distribution in the range of $[C(1-RDD/2)$, $C(1+RDD/2)]$, with C being the sum of all handling times (c_{ij}) divided by the square of the number of berths and RDD standing for the relative range of due dates. The values of *RDD* are taken as 0.2, 0.4, 0.6, 0.8, and 1.0. Preprocessing has first been made to avoid excessively large values of k in variables and equations. From the computational experiments, it appears that the due dates do not change computing times. The reason for this is probably that CPLEX can detect some variables that must be equal to 0 and fix them in a preprocessing phase.

6 Conclusions

The dynamic berth allocation problem is an important one in operations of container ports. Imai *et al.* (2001) recently introduced a model for it. After correcting it, a more compact equivalent model has been proposed, and extended to accommodate constraints on size, the due dates of ships and a bounded planning horizon.

Computational experiments with CPLEX on both models show that the first model, DBAP, is able to solve problems with up to 10 berths and 50 ships. However, the second, more compact model, DBAP$'$, is not efficient due to the big M's in the constraints. Although it appears that this paper is the first to report the exact solution of berth allocation problems of a realistic size, much work remains to be done, in several directions:

(i) enrich the DBAP model, to take costs into account, instead of times; in particular, to express differences in cost for waiting and handling, as well as for operations at different berths;

(ii) develop new heuristics to quickly solve large instances to near optimality, using recent metaheuristics such as Variable Neighborhood Search (Mladenović and Hansen, 1997), and compare them with previously proposed ones;

(iii) explore an exact solution for DBAP and extensions using advanced mixed integer programming techniques such as stabilized column generation (du Merle *et al.*, 1999) combined with branch-and-bound and adapted to their particular structure.

Acknowledgments This work was carried out during two visits of the first author at the Department of Management, the Hong Kong Polytechnic University, whose support is gratefully acknowledged. The work described in this paper was also partially supported by a grant from The Hong Kong Polytechnic University (Project No. G-T247). The authors are grateful to Dragan Urošević for running the computational experiments and to Nenad Mladenović for stimulating discussions.

References

1. Bertsekas, D.P. (1992) Auction algorithms for network flow problem: a tutorial introduction. *Computational Optimization and Applications*, 1, 7-66.
2. Bruno, J., Coffman, E.G. and Sethi, R. (1974) Scheduling independent tasks to reduce mean finishing time. *Communications of the ACM*, 17 (7), 382-387.
3. du Merle, O., Villeneuve, D., Desrosiers, J. and Hansen, P. (1999) Stabilized column generation. *Discrete Mathematics*, 194, 229–237.

4. Forrest, J.J.H., Hirst, J.P.H. and Tomlin, J.A. (1974) Practical solution of large mixed-integer programming problems with Umpire. *Management Science*, A20, 736-773.
5. Imai, A., Nagaiwa, K. and Tat, C.W. (1997) Efficient planning of berth allocation for container terminals in Asia. *Journal of Advanced Transportation*, 31, 75-94.
6. Imai, A., Nishimura, E. and Papadimitriou, S. (2001) The dynamic berth allocation problem for a container port. *Transportation Research Part B*, 35, 401-417.
7. Imai, A., Nishimura, E. and Papadimitriou, S. (2003) Berth allocation with service priority. *Transportation Research B*, 37, 437–457.
8. Jonker, R. and Volgenant A. (1987) A shortest augmenting path algorithm for dense and sparse linear assignment problem. *Computing*, 38, 325-340.
9. Lenstra, J.K., Rinnooy Kan, A.H.G. and Brucker, P. (1977) Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, 1, 343-362.
10. Lim, A. (1998) The berth planning problem. *Operations Research Letters*, 22, 105-110.
11. Meersmans, P.J.M. and Dekker, R. (2001) Operations Research supports container handling. *Economic Institute Report EI 2001-22*, Erasmus University, Rotterdam, Netherlands.
12. Mladenović, N. and Hansen, P. (1997) Variable neighborhood search. *Computers and Operations Research*, 24, 1097–1100.
13. Nishimura, E., Imai, A. and Papadimitriou, S. (2001) Berth allocation planning in the public berth system by genetic algorithms. *European Journal of Operational Research*, 131, 282-292.
14. Park, K.T. and Kim, K.H. (2002) Berth scheduling for container terminals by using sub-gradient optimization technique. *Journal of the Operational Research Society*, 53, 1054-1062.
15. Peterkofsky, R.I. and Daganzo, C.F. (1990) A branch-and-bound algorithm for the crane scheduling problem. *Transportation Research Part B*, 24B, 159-172.
16. Vis, I.F.A. and de Koster, R. (2003) Transshipment of containers at a container terminal: An overview. *European Journal of Operational Research*, 147, 1-16.
17. Ward, T. (2002) Container terminal capacity numbers and policy.
`http://www.jwdgroup.com/images/ppp/ctmeasuremant/`