Transportation Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

An Exact Algorithm for the Integrated Planning of Berth Allocation and Quay Crane Assignment

Ilaria Vacca, Matteo Salani, Michel Bierlaire,

TRANSPORTATION SCIENCE
Vol. 47, No. 2, May 2013, pp. 148–161
ISSN 0041-1655 (print) | ISSN 1526-5447 (online)

http://dx.doi.org/10.1287/trsc.1120.0428
© 2013 INFORMS

An Exact Algorithm for the Integrated Planning of Berth Allocation and Quay Crane Assignment

Ilaria Vacca
Transport and Mobility Laboratory, School of Architecture, Civil and Environmental Engineering, École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland, ilaria.vacca@epfl.ch

Matteo Salani
Dalle Molle Institute for Artificial Intelligence (IDSIA), CH-6928 Manno-Lugano, Switzerland, matteo.salani@idsia.ch

Michel Bierlaire
Transport and Mobility Laboratory, School of Architecture, Civil and Environmental Engineering, École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland, michel.bierlaire@epfl.ch

In this paper we study the simultaneous optimization of berth allocation and quay crane assignment in seaport container terminals. We propose a model based on an exponential number of variables that is solved via column generation. An exact branch and price algorithm is implemented to produce optimal integer solutions to the problem. In particular, we present several accelerating techniques for the master and the pricing problem that can be generalized to other branch and price schemes. Computational results show that the proposed approach outperforms commercial solvers. Furthermore, the developed algorithm allows for a comparative analysis between the hierarchical and the integrated solution approach that confirms the added value of integration in terms of cost reduction and efficient use of resources. To the best of our knowledge, this is the first exact branch and price algorithm for both the berth allocation problem and the berth allocation problem with quay crane assignment.

Key words: container terminal management; berth allocation; quay crane assignment; integrated planning; column generation; branch and price
History: Received: March 2011; revisions received: January 2012, March 2012; accepted: March 2012. Published online in *Articles in Advance* August 2, 2012.

Introduction

Containerized sea-freight transportation has grown dramatically over the last two decades, much faster than other sea transportation modes. Container traffic increased about 9.5% per year between 2000 and 2008, while the average annual rate for cargo traffic increased by 5.3% (ISL 2009). The share of containerized trade in the world's total dry cargo increased from 5.1% in 1980 to 25.4% in 2008 (UNCTAD 2009).

Similar to air transport and the airline industry, which have greatly benefited from operations research (OR) since the 1950s (Barnhart, Belobaba, and Odoni 2003), maritime transport and seaport logistics represent a more recent OR research field. It has mainly been pushed forward by the competitive environment, forcing operators to optimize their cost to maintain margins. The optimization of container terminal operations has received increasing interest in the scientific literature over the last few years and current research directions in container terminal management point toward the integrated planning of operations. This could potentially yield significant improvements in terms of efficiency and productivity for the terminal. From a mathematical point of view, the resulting integrated problems are very complex. Therefore, large-scale optimization techniques need to be designed in order to cope with this complexity and provide exact solutions. In particular, column generation and branch and price schemes represent nowadays the most successful tool to solve such complex problems (Lübbecke and Desrosiers 2005). They have been recently applied to maritime transportation (Grønhaug et al. 2010) and container terminal management (Choo, Klabjan, and Simchi-Levi 2010).

In this paper we present an exact algorithm for the integrated planning of berth allocation and quay crane assignment in the context of container terminal management. The problem aims at assigning vessels to berthing positions, performing the scheduling of vessels in each berth and allocating quay cranes (QC) to vessels over a given time horizon, taking into account the quay crane capacity of the terminal. Although different mathematical models and heuristic procedures have been designed for the joint optimization problem (cf. the survey by Bierwirth and Meisel 2010), no exact approach has been proposed so far.

We propose a model for the integrated optimization problem based on an exponential number of variables. The model exploits the problem structure and it is solved by column generation. Furthermore, we design and implement an exact branch and price algorithm with the purpose of proving good-quality bounds and optimal solutions to the problem. We propose a specific branching scheme and several accelerating techniques both for the pricing and the master problem. Compared to state-of-the-art techniques such as bidirectional dynamic programming and dual stabilization, we reduce the number of states in the pricing problem by defining a new dominance rule and we speed up the dynamic programming (DP) algorithm by introducing an incremental heuristic DP step. These techniques, specifically designed for our problem, proved to be very useful and can be easily generalized to other branch and price schemes. Furthermore, our algorithm can be adapted to solve the berth allocation problem only. Therefore, the exact branch and price enables us to perform a comparative analysis between the hierarchical and the integrated optimization approach that confirms the added value of integration in terms of cost reduction and efficient use of resources. To the best of our knowledge, this is the first exact branch and price algorithm for both the berth allocation problem and the berth allocation problem with quay crane assignment. We intend to compare our exact approach with existing exact algorithms, i.e., general purpose branch and cut. The comparison against other heuristic algorithms is not within the scope of this paper.

The paper is organized as follows. Recent contributions concerning the integrated planning of berth allocation and quay crane assignment are reviewed in §1. The joint problem is described in §2, where the mathematical model is presented. The column generation scheme is illustrated in §3 while §4 provides details on the implementation of the exact branch and price algorithm. Computational results are discussed in §5, including some insights on the comparison between hierarchical versus integrated solution approaches. Section 6 concludes the paper.

1. Literature Review

The need for an efficient management of logistic activities at modern container terminals is well recognized, and there exists a rich literature of optimization models and algorithms designed for specific operational problems.

Vis and de Koster (2003) illustrate the main logistic processes in a container terminal, reporting about 50 references up to 2001. Steenken, Voss, and Stahlbock (2004) present an exhaustive overview of optimization methods in container terminal management, reviewing more than 200 references up to 2004; the survey has been recently updated by Stahlbock and Voss (2008). A review of operations research methods in maritime logistics by Christiansen, Fagerholt, and Ronen (2004) focuses more generally on ship routing and scheduling.

A promising research track is represented by the integrated optimization of decision problems that are highly interdependent, yet usually solved hierarchically by terminal planners, as pointed out by Vacca, Bierlaire, and Salani (2007), Vacca, Salani, and Bierlaire (2010). In particular, a recent survey by Bierwirth and Meisel (2010) reviews contributions on integrated solution approaches for the berth allocation problem (BAP) and the quay crane assignment problem (QCAP).

The integrated planning of berth allocation and quay crane assignment, introduced by Park and Kim (2003) has been further investigated by Imai et al. (2008) and Meisel and Bierwirth (2009). The resulting models represent a good starting point for tackling such a complex problem; however, they still present some unrealistic assumptions and limits. In fact, the relationship between the number of quay cranes and the handling time is ignored (Imai et al. 2008) or the crane productivity is assumed to be proportional to the number of QCs (Park and Kim 2003), although it is known that quay crane interference reduces the marginal productivity. Furthermore, all the authors, including Meisel and Bierwirth (2009), assign quay cranes hour by hour in their models, without any control on the final outcome; in particular, feasible assignments may include quay cranes with idle times within the same service.

These unrealistic assumptions are addressed by the model proposed by Giallombardo et al. (2010). The joint problem, called the tactical berth allocation problem (TBAP), makes use of the concept of quay crane assignment profiles to capture real-world issues. A quay crane profile encodes the number of quay cranes assigned to a vessel along the time steps that the vessel is berthed at the port. The new modeling feature is designed to include some requirements that terminals impose on the quay crane assignment process: quay cranes can be moved from one vessel to another only at the end of a working shift; productivity losses due to crane interference are taken into account in the profile definition as well as the vessels' priorities in terms of the number of quay cranes and handling time. The authors propose two mixed integer programming (MIP) formulations and a heuristic algorithm based on tabu search and mathematical programming techniques, and provide computational results based on real-world instances.

From an algorithmic point of view, all the mentioned approaches rely on heuristic methods to provide good and fast solutions. To the best of our knowledge, the only attempt to solve exactly the berth allocation problem (without quay crane assignment) is the one by Buhrkal et al. (2011): the authors present a generalized set-partitioning model for the discrete BAP where all columns are enumerated a priori. The formulation outperforms existing models (e.g., Imai, Nishimura, and Papadimitriou 2001; Cordeau et al. 2005) and guarantees optimality; however, authors recognize that a branch and price algorithm should be implemented in order to solve larger instances.

From the reviewed contributions we remark that there exists no exact algorithm for the integrated planning of berth allocation and quay crane assignment. However, we believe that it is important to characterize optimal solutions, and in this paper we intend to exploit the problem structure in order to design an exact algorithm able to cope with the problem complexity.

2. The Mathematical Model

We consider the problem defined by Giallombardo et al. (2010) for the tactical berth allocation problem. We refer the readers interested in the motivations related to all modeling assumptions to the mentioned publication. Here, we provide a brief discussion of problem settings.

Given a set of vessels and a set of berths, the joint optimization problem aims at assigning a berthing position and a quay crane assignment profile to every vessel over a given time horizon as well as at scheduling incoming vessels according to their time windows. A quay crane profile is defined by its monetary value, i.e., the price charged by the terminal to the shipping companies for the provided service, its time duration, and its number of quay cranes per shift. Variable profiles in terms of assigned quay cranes permit a better exploitation of a terminal's resources as quay cranes can be shared among vessels. Planners must ensure the so-called end-of-shift constraints, i.e., that quay cranes have to move from one vessel to another at the end of a working shift. We refer the reader to Giallombardo et al. (2010) for any further details related to quay crane profiles. The objective is to maximize the difference between the revenue associated with the chosen quay crane profiles and the housekeeping costs generated by transshipment flows exchanged by the vessels.

The key point is that the handling time associated with the vessel directly depends on the number of assigned quay cranes and thus on the assigned profile. Furthermore, the total capacity of the terminal in terms of quay cranes cannot be exceeded.

Time is discretized in time steps of constant length.

The input data of the tactical berth allocation problem are:

N	set of vessels;
M	set of berths;
H	set of time steps;
P_i	set of feasible quay crane assignment profiles for vessel $i \in N$;
t_i^p	handling time associated with profile $p \in P_i$, $i \in N$, expressed as number of time steps;
v_i^p	monetary value associated with the quay crane profile $p \in P_i$;
q_i^{pu}	number of quay cranes used by profile $p \in P_i$, $i \in N$, at time step $u \in (1, \ldots, t_i^p)$;
Q^h	maximum number of quay cranes available at time step $h \in H$;
f_{ij}	number of containers exchanged between vessels $i, j \in N$;
g_{ij}	binary coefficient equal to 1 if vessel i exchanges containers with vessel j, i.e., if $f_{ij} > 0$;
d_{kw}	unit housekeeping cost between yard slots corresponding to berths $k, w \in M$;
$[a_i, b_i]$	[earliest, latest] feasible service time step of ship $i \in N$;
$[a^k, b^k]$	[start, end] of availability time step of berth $k \in M$.

Our model is based on the concept of *berth sequence*, a sequentially ordered subset of ships in a berth with an assigned quay crane profile and a start of service time. We define the set Ω^k for every $k \in M$ that represents the set of all feasible sequences of vessels moored at berth k—feasible with respect to time windows constraints, end-of-shift constraints and with a unique quay crane profile assigned to each vessel.

To model sequences of vessels in berth $k \in M$, the graph $G^k = (V^k, A^k)$ $\forall k \in M$ is defined, where $V^k = N \cup \{o(k), d(k)\}$ and $o(k)$ and $d(k)$ are additional vertices representing berth k. The set of arcs $A^k \subseteq V^k \times V^k$ represents feasible precedence relationships between vessels: an arc (i, j) exists if vessel i can precede vessel j according to their time windows.

The decision variables of our optimization problem are:

$x_{ij}^k \in \{0, 1\}$	$\forall k \in M$, $\forall (i, j) \in A^k$, equal to 1 if ship j is scheduled after ship i at berth k, and 0 otherwise;
$y_i^k \in \{0, 1\}$	$\forall k \in M$, $\forall i \in N$, equal to 1 if ship i is assigned to berth k, and 0 otherwise;
$\lambda_i^p \in \{0, 1\}$	$\forall p \in P_i$, $\forall i \in N$, equal to 1 if ship i is served by the profile p, and 0 otherwise;
$s_{r_k} \in \{0, 1\}$	$\forall r_k \in \Omega^k$, associated with the selection of sequence r_k. It equals 1 if sequence r_k is chosen in the solution and 0 otherwise;
$z_{ij}^{kw} \geq 0$	$\forall i, j \in N$, $\forall k, w \in M$, auxiliary variables resulting from the linearization of the model proposed by Giallombardo et al. (2010), $z_{ij}^{kw} = y_i^k y_j^w$, equal to 1 if ship i moors at berth k and ship j moors at berth w, and 0 otherwise.

Every berth sequence $r_k \in \Omega^k$ is described by the following coefficients:

x_{ijr_k}	binary coefficient equal to 1 if vessel j follows vessel i in sequence r_k;
y_{ir_k}	binary coefficient equal to 1 if vessel i is moored at berth k in sequence r_k;
$\lambda^p_{ir_k}$	binary coefficient equal to 1 if profile p is assigned to vessel i in sequence r_k;
$q^h_{r_k}$	coefficient that counts the number of quay cranes used by sequence r_k at time step h;
v_{r_k}	monetary value associated with sequence r_k, defined as $v_{r_k} = \sum_{i \in N} \sum_{p \in P_i} \lambda^p_{ir_k} v^p_i$.

The tactical berth allocation problem is formulated as follows:

$$\max \left\{ \sum_{k \in M} \sum_{r_k \in \Omega^k} v_{r_k} s_{r_k} - \sum_{i,\,j \in N} \sum_{k,\,w \in M} f_{ij} d_{kw} z^{kw}_{ij} \right\} \tag{1}$$

$$\text{s.t.} \quad \sum_{k \in M} \sum_{r_k \in \Omega^k} y_{ir_k} s_{r_k} = 1 \quad \forall\, i \in N, \tag{2}$$

$$\sum_{k \in M} \sum_{r_k \in \Omega^k} q^h_{r_k} s_{r_k} \le Q^h \quad \forall\, h \in H, \tag{3}$$

$$\sum_{r_k \in \Omega^k} s_{r_k} \le 1 \quad \forall\, k \in M, \tag{4}$$

$$\sum_{k \in M} \sum_{w \in M} z^{kw}_{ij} = g_{ij} \quad \forall\, i, j \in N, \tag{5}$$

$$\sum_{r_k \in \Omega^k} y_{ir_k} s_{r_k} - z^{kw}_{ij} \ge 0 \quad \forall\, i, j \in N,\ \forall\, k, w \in M, \tag{6}$$

$$\sum_{r_w \in \Omega^w} y_{jr_w} s_{r_w} - z^{kw}_{ij} \ge 0 \quad \forall\, i, j \in N, \forall\, k, w \in M, \tag{7}$$

$$\sum_{r_k \in \Omega^k} x_{ijr_k} s_{r_k} = x^k_{ij} \quad \forall\, i, j \in N,\ \forall\, k \in M, \tag{8}$$

$$\sum_{r_k \in \Omega^k} y_{ir_k} s_{r_k} = y^k_i \quad \forall\, i \in N,\ \forall\, k \in M, \tag{9}$$

$$\sum_{k \in M} \sum_{r_k \in \Omega^k} \lambda^p_{ir_k} s_{r_k} = \lambda^p_i \quad \forall\, p \in P_i,\ \forall\, i \in N, \tag{10}$$

$$s_{r_k} \in \{0, 1\} \quad \forall\, r_k \in \Omega^k,\ \forall\, k \in M, \tag{11}$$

$$x^k_{ij} \in \{0, 1\} \quad \forall\, (i, j) \in A^k,\ \forall\, k \in M, \tag{12}$$

$$y^k_i \in \{0, 1\} \quad \forall\, i \in N,\ \forall\, k \in M, \tag{13}$$

$$\lambda^p_i \in \{0, 1\} \quad \forall\, p \in P_i,\ \forall\, i \in N. \tag{14}$$

$$z^{kw}_{ij} \ge 0 \quad \forall\, i, j \in N,\ \forall\, k, w \in M. \tag{15}$$

The objective function (1) maximizes the difference between the total monetary value associated with the selected sequences, i.e., the total value of selected profiles, and the total housekeeping cost generated by the berth allocation plan. Constraints (2) ensure that every ship is assigned to exactly one sequence, and thus to one berth, while constraints (3) ensure that the quay crane capacity is not violated. Constraints (4) select at most one sequence for each berth. Constraints (5)–(7) support the linearization of the model proposed by Giallombardo et al. (2010). Constraints (8)–(10) link decision variables x^k_{ij}, y^k_i, and λ^p_i to s_{r_k}. Finally, the integrality of the solution is ensured by constraints (11)–(14).

Furthermore, we remark that model (1)–(15) when variables s_{r_k} are relaxed, corresponds to the Dantzig-Wolfe reformulation of the linearization of the TBAP model by Giallombardo et al. (2010).

3. Column Generation

If we relax the integrality requirements (11)–(14), constraints (8)–(10) become redundant and can be removed from the formulation. The resulting model is referred to as the *master problem*.

The resulting linear program involves an exponential number of variables (columns). Therefore, the column generation scheme starts solving a *restricted master problem*, defined on a subset of columns and, at each iteration, it generates new profitable columns to be added to the formulation, if any, by solving the pricing problem.

3.1. Pricing Subproblem

Let π, μ, ξ, θ, and η be the dual vectors associated with constraints (2)–(4), (6), and (7), respectively. Given an optimal solution of the restricted master problem, the reduced cost of sequence $r \in \Omega^k$ is given by:

$$\tilde{v}_{r_k} = v_{r_k} - \sum_{i \in N} \pi_i y_{ir_k} - \sum_{h \in H} \mu^h q^h_{r_k} - \xi_k - \sum_{i,\,j \in N} \sum_{w \in M} \theta^{kw}_{ij} y_{ir_k} - \sum_{i,\,j \in N} \sum_{w \in M} \eta^{kw}_{ij} y_{jr_w}.$$

The pricing subproblem identifies, for every berth $k \in M$, the column r^*_k with the maximum reduced cost. Additional decision variables are:

- $T_i \ge 0$, representing the start-of-service time step of ship $i \in N$;
- $T_o \ge 0$, representing the time step when operations start in the berth;
- $T_d \ge 0$, representing the time step when operations end in the berth;
- $\gamma^h_i \in \{0, 1\}$, equal 1 if ship i starts operations at time step h.

Additional parameters to support end-of-shift constraints:

- S is the set of time step indexes relative to a working shift (the time step is normally a submultiple of a working shift);
- H^s is the subset of H containing all time steps corresponding to time step $s \in S$ within a working shift;
- P^s_i is the subset of P_i containing all quay crane profiles of ship i compatible with time step $s \in S$ within a working shift.

For example, if the working shift is six hours and the time step is set to two hours, then $S=\{1,2,3\}$ and for every vessel we have three (possibly empty) sets of quay crane profiles compatible with time steps in S.

The pricing subproblem is formulated as follows:

$$\max \left\{ \sum_{i\in N}(v_i^p\lambda_i^p-\pi_i y_i)-\sum_{i\in N}\sum_{p\in P_i}\sum_{h=1\ldots t_i^p}\mu^{T_i+h-1}q_i^{ph}\lambda_i^p-\xi_k - \sum_{i,j\in N}\sum_{w\in M}(\theta_{ij}^{kw}y_i+\eta_{ij}^{kw}y_j)\right\} \tag{16}$$

$$\text{s.t.} \quad \sum_{j\in N\cup\{d\}} x_{o,j}=1, \tag{17}$$

$$\sum_{i\in N\cup\{o\}} x_{i,d}=1, \tag{18}$$

$$\sum_{j\in N\cup\{d\}} x_{ij}-\sum_{j\in N\cup\{o\}} x_{ji}=0 \quad \forall i\in N, \tag{19}$$

$$\sum_{j\in N\cup\{d\}} x_{ij}=y_i \quad \forall i\in N, \tag{20}$$

$$T_i+\sum_{p\in P_i} t_i^p\lambda_i^p-T_j\le(1-x_{ij})M1 \quad \forall i\in N, \quad \forall j\in N\cup\{d(k)\}, \tag{21}$$

$$T_o-T_j\le(1-x_{o,j})M2 \quad \forall j\in N, \tag{22}$$

$$a_i y_i\le T_i \quad \forall i\in N, \tag{23}$$

$$T_i\le b_i y_i \quad \forall i\in N, \tag{24}$$

$$a^k\le T_o, \tag{25}$$

$$T_d\le b^k, \tag{26}$$

$$\sum_{p\in P_i}\lambda_i^p=y_i \quad \forall i\in N, \tag{27}$$

$$\sum_{h\in H^s}\gamma_i^h=\sum_{p\in P_i^s}\lambda_i^p \quad \forall i\in N,\ \forall s\in S, \tag{28}$$

$$T_i-b^h\le(1-\gamma_i^h)M3 \quad \forall i\in N,\ \forall h\in H, \tag{29}$$

$$a^h-T_i\le(1-\gamma_i^h)M4 \quad \forall i\in N,\ \forall h\in H, \tag{30}$$

$$x_{ij}\in\{0,1\} \quad \forall (i,j)\in A^k, \tag{31}$$

$$y_i\in\{0,1\} \quad \forall i\in N, \tag{32}$$

$$\lambda_i^p\in\{0,1\} \quad \forall p\in P_i,\ \forall i\in N, \tag{33}$$

$$\gamma_i^h\in\{0,1\} \quad \forall i\in N,\ \forall h\in H, \tag{34}$$

$$T_i\ge 0 \quad \forall i\in N\cup\{o,d\}, \tag{35}$$

where $M1$, $M2$, $M3$, and $M4$ are large positive constants.

The objective function (16) maximizes the reduced cost of a column in berth k. Constraints (17)–(19) ensure flow conservation, while variables y_i are linked to variables x_{ij} by constraints (20). Precedence constraints (21)–(22) and time window constraints (23)–(26) ensure the correct scheduling of vessels over time while the profile assignment is controlled by the set of constraints (27). Constraints (28)–(30) ensure that only compatible quay crane profiles can be selected for time step h. When the length of a time step coincides with the length of a shift, all these constraints and supporting variables and sets are redundant. We adopted this formulation for two reasons: first, we decouple the choice of time step length from that of a shift length; second, we permit modeling common practices in container terminals in which operations can be started or finished earlier or later in a working shift preserving the constraint of not moving quay cranes within a working shift. Finally, constraints (31)–(35) define the domain of variables.

We remark that index k has disappeared from decision variables x and y with respect to the notation introduced in §2, since the pricing subproblem is solved for a fixed berth k.

We remark that the above formulation is provided for the mathematical correctness of the model. The implemented algorithm to solve the pricing problem makes use of a dynamic programming algorithm which does not directly use this formulation.

At each iteration of column generation, we solve $m=|M|$ subproblems, one for each berth $k\in M$. If $\tilde{v}_{r_k^*}>0$ for some k, we add column $s_{r_k^*}$ to the restricted master problem and we iterate the process; otherwise, the current solution of the master problem is proven to be optimal and we stop.

3.1.1. Dynamic Programming. The pricing subproblem (16)–(35) can be cast to a resource constrained elementary shortest path problem (RCESPP), where the resource is represented by time, and it is solved by means of dynamic programming. The underlying network $G(V,A)$ has one vertex for every vessel $i\in N$, for every profile $p\in P_i$ and for every time step $h\in H$. Transit time on arcs equals the handling time of profile $p\in P_i$ assigned to vessel i. Vertex (i,h,p) represents vessel i berthed at time step h and operated by quay crane profile p. The graph has two additional vertices, o, d, associated with the specific berth $k\in M$ for which the pricing is solved, representing the origin and the destination of the path. Feasibility constraints related to the starting time of a quay crane profile are encoded in the network structure: arcs are present only if profile p can start at time step h.

The RCESPP aims at finding a maximum-cost elementary path from o to d that satisfies the constraints on resources: the objective function of the RCESPP associated with the pricing subproblem corresponds to (16), while the resource constraint requires that it does not exceed the given time horizon.

The dynamic programming (DP) algorithm iteratively extends states. A *state* for vertex (i,h,p) represents a path from o to (i,h,p); many states are associated with the same vertex (i,h,p), representing different paths. Each state is encoded by a *label* of the form (S, τ, C, i, h, p), that is a path from o to (i, h, p) with time consumption τ and cost C; furthermore, to ensure elementarity, set S keeps track of vessels visited along the path (Beasley and Christofides 1989). The optimal solution is given by the maximum cost state associated with the destination vertex d.

At vertex o, time consumption τ is initialized at 0 and $S = \{o\}$; cost C is initialized to $-\xi_k$, where k is the berth for which the pricing problem is being solved. When extending state $(S, \tau, C, i, h_i, p_i)$ to another feasible state $(S', \tau', C', j, h_j, p_j)$, the label is updated according to the formula:

$$S' = S + \{j\}, \tag{36}$$

$$\tau' = h_j + t_j^{p_j}, \tag{37}$$

$$C' = C + v_{p_j} - \pi_j - \sum_{h=h_j}^{h_j+t_j^{p_j}} \mu^h q_j^{p_j(h-h_j)} - \sum_{n\in N}\sum_{w\in M} (\theta_{jn}^{kw} + \eta_{nj}^{wk}). \tag{38}$$

The extension is feasible if $j \notin S$ (elementarity), $\tau' < |H|$ (total duration), and h_j satisfies time windows $[a_j, b_j]$.

The efficiency of dynamic programming strongly depends on the effectiveness of *dominance* rules that are used to fathom feasible, yet nonoptimal states. In particular, dominated states are not extended further. According to Beasley and Christofides (1989), state $(S', \tau', C', j, h_j, p_j)$ dominates $(S'', \tau'', C'', j, h_j, p_j)$ if:

$$S' \le S'', \tag{39}$$

$$\tau' \le \tau'', \tag{40}$$

$$C' \ge C'', \tag{41}$$

and at least one of these inequalities is strictly satisfied.

4. Branch and Price Algorithm

To obtain integer solutions, we implement a branch and price algorithm where column generation is applied at every node of the search tree. The search tree is explored according to a best-first strategy with respect to the upper bound associated with the node. The algorithm makes use of a column pool that keeps track of all columns generated in different nodes of the search tree.

In the remainder of this section we illustrate the branching rules as well as accelerating techniques both for the pricing and the master problem.

4.1. Branching Scheme

In the search tree, branching is required when the master problem is solved at optimality and the corresponding solution in terms of the original formulation's variables is not integer. We implement a branching scheme consisting of four hierarchical levels.

1. If the total number of selected sequences $\tilde{K} = \sum_{k\in M}\sum_{r_k\in\Omega^k} s_{r_k}$ is fractional, then branching requires an additional constraint to be added in the master problem:
   - $\sum_{k\in M}\sum_{r_k\in\Omega^k} s_{r_k} \le \lfloor \tilde{K} \rfloor$ on the first child node;
   - $\sum_{k\in M}\sum_{r_k\in\Omega^k} s_{r_k} \ge \lceil \tilde{K} \rceil$ on the second child node.

This branching requires the dual value associated with the additional constraint, denoted by π_0, to be collected and accounted for in the pricing subproblem. We remark that π_0 is a constant, regardless of the berth. In particular, the additional constraint in the master problem modifies the objective function of the pricing subproblem as follows:

$$\max\left\{ v_{r_k} - \sum_{i\in N} \pi_i y_{ir_k} - \sum_{h\in H} \mu^h q_{r_k}^h - \xi_k - \sum_{i,j\in N}\sum_{w\in M} (\theta_{ij}^{kw} y_{ir_k} + \eta_{ij}^{kw} y_{jr_w}) - \pi_0 \right\}.$$

2. If some vessel $i \in N$ is assigned fractionally to some berth $k \in M$, i.e., quantity $\tilde{Y}_i^k = \sum_{r_k\in\Omega^k} y_i^{r_k} s_{r_k}$ is fractional, then the branching requires an additional constraint to be added to the master problem for the given vessel i and berth k:
   - $\sum_{r_k\in\Omega^k} y_i^{r_k} s_{r_k} = 0$ on the first child node;
   - $\sum_{r_k\in\Omega^k} y_i^{r_k} s_{r_k} = 1$ on the second child node.

This branching requires the dual value associated with the additional constraint, denoted by φ_i^k, to be taken into account in the pricing subproblem. We remark that φ_i^k is collected in the pricing for berth k if vessel i is visited by the sequence. In particular, the additional constraints in the master problem modify the objective function of the pricing subproblem as follows:

$$\max\left\{ v_{r_k} - \sum_{i\in N} \pi_i y_{ir_k} - \sum_{h\in H} \mu^h q_{r_k}^h - \xi_k - \sum_{i,j\in N}\sum_{w\in M} (\theta_{ij}^{kw} y_{ir_k} + \eta_{ij}^{kw} y_{jr_w}) - \pi_0 - \sum_{i\in N} \varphi_i^k y_{ir_k} \right\}.$$

3. If some profile $p \in P_i$ is assigned fractionally to some vessel $i \in N$, i.e., quantity $\tilde{\Lambda}_i^p = \sum_{k\in M}\sum_{r_k\in\Omega^k} \lambda_i^{pr_k} s_{r_k}$ is fractional, then branching is handled directly in the pricing subproblem by modifying the set P_i of feasible profiles for vessel i. On the first child node, we enforce profile p to be assigned to vessel i by removing all other feasible profiles from set P_i; this branching corresponds to enforcing $\lambda_i^p = 1$ in the original formulation. On the second child node, we prevent profile p from being used by removing it by set P_i; this branching corresponds to enforcing $\lambda_i^p = 0$ in the original formulation. We remark that neither the master nor the pricing formulation is modified by this branching in terms of objective function and

additional constraints. However, infeasible columns must be removed from the master problem, according to the branching decision associated with the analyzed node.

4. If none of the above conditions holds, then there exists some vessel $i \in N$ such that the quantity $\tilde{T}_i^h = \sum_{k\in M}\sum_{r_k\in\Omega^k} T_i^{hr_k} s_{r_k}$ is fractional for some $h^* \in H$, where $T_i^{hr_k}$ is a binary coefficient equal to 1 if vessel i starts being serviced at time step h in sequence r_k. In this case, branching is handled in the pricing subproblem, by modifying the time windows $[a_i, b_i]$ associated with vessel i, as similarly proposed by Gélinas et al. (1995) for the vehicle routing problem. We denote by t_i^* the service time associated with time step h^*. On the first child node, we enforce the vessel to be serviced before time step h^*: the new time windows for vessel i are therefore $[a_i, t_i^* - \epsilon]$ and this corresponds to enforce $\gamma_i^h = 0\ \forall\, h \geq h^*$ in the original formulation of the pricing. On the second child node, we enforce the vessel to be serviced at or after time step h^*: the new time windows for vessel i are therefore $[t_i^*, b_i]$ and this corresponds to enforce $\gamma_i^h = 0\ \forall\, h < h^*$ in the original formulation. We remark that neither the master nor the pricing formulation is modified by this branching in terms of objective function and additional constraints. However, infeasible columns must be removed from the master problem, according to the branching decision associated with the analyzed node.

The order of branching is determined by the increasing complexity of the branching rules due to additional constraints in the master problem or additional complexity of the pricing problem.

4.2. Accelerating Techniques

4.2.1. Solving Exact Dynamic Programming. We implement state-of-the-art techniques for solving the RCESPP such as bounded bidirectional dynamic programming and decremental state space relaxation.

Bounded bidirectional DP (Righini and Salani 2006) consists of two steps: firstly, states are extended in forward and backward direction until half of the so-called *critical resource* (time, in our case) is consumed; secondly, forward and backward paths are joined to produce feasible sequences. Bounding is used to discard nondominated nonoptimal states.

The basic idea of decremental state space relaxation (Righini and Salani 2008) is to start checking elementarity only on a subset $\bar{S}$ of S. If the final solution is nonelementary, one or more vertices violating the constraint are added to $\bar{S}$ and DP is executed again.

The implemented search policy takes into account time windows (Liberatore, Righini, and Salani 2011). At every iteration of dynamic programming, states are explored according to the vertices (vessels) they are associated with. We decide to sort vessels according to the starting time a_i of their time windows; this search strategy proves to be important for the effectiveness of the algorithm in our tests.

Furthermore, we design an additional technique for accelerating the exact pricing, which is specifically designed for our pricing subproblem.

Domination of (h, p) *pairs.* Unlike RCESPP subproblems arising in vehicle routing, where customers are visited one right after the other, in our problem it may be convenient to wait some time between the departure of a vessel and the start of service of the next one. This is due to the quay crane capacity constraint that controls the interactions between berths at the master problem level; in particular, these interactions are captured by dual vectors θ and η. More specifically, when extending a label to the next vessel j, we have as many new states as the number of feasible service time steps h_j; furthermore, we may have more than one profile p_j associated with a single time step h_j and vice versa. To reduce the number of states, preprocessing is performed at the beginning of the DP algorithm: we populate a list of nondominated (h_j, p_j) pairs for every vessel j and we refer to this list when extending a label to vessel j. The domination criteria considers the contribution to the reduced cost of the selected profile executed at time h_j and its length.

We remark that in the special case of $\theta = 0$ and $\eta = 0$, the list has at most one pair (h_j, p_j) for every profile p_j. When the DP algorithm handles the feasibility check related to time resources for a specific time interval, it shifts the closest nondominated pair to match the actual service time if no pair exists for the specific time period. In this way, no potentially improving (h_j, p_j) pair is left untreated and optimality is guaranteed.

4.2.2. Heuristic Pricing. The pricing subproblem is first solved heuristically. An exact solution is computed only if needed.

The heuristic dynamic programming algorithm is based on a relaxed dominance rule that allows us to eliminate many more states during the comparison of labels (Dell'Amico, Righini, and Salani 2006). The final solution is an elementary shortest path that satisfies resource constraints; however, optimality is no longer guaranteed. When using relaxed dominance, we say that state $(S', \tau', C', j, h_j, p_j)$ dominates $(S'', \tau'', C'', j, h_j, p_j)$ if:

$$\tau' \leq \tau'', \tag{42}$$

$$C' \geq C'', \tag{43}$$

and at least one of these inequalities is strictly satisfied. In other words, we do not compare anymore the set of vertices S', S'' visited by the partial paths. As the dominance is weaker, the number of eliminated labels is greater. This results in a reduced computational effort to solve the pricing. In particular, we remove from the dominance criteria the set S of visited nodes as it encodes the combinatorial nature of the problem. After its removal, the solution can be found in pseudo-polynomial time.

Furthermore, the following accelerating techniques are implemented with the main purpose of avoiding the call to exact pricing as much as possible.

Multiple pricing strategy. At every iteration of column generation, we first solve a pricing subproblem for every berth $k \in M$ using the heuristic dynamic programming algorithm; exact DP is called only if heuristic pricing cannot provide a positive reduced cost column. As soon as we find a positive reduced cost column for some berth k^*, the pricing terminates. At this point, columns generated for berth k^* are evaluated for all berths $k \neq k^*$, $k \in M$: if a column is feasible for another berth, say $\bar{k}$, and its reduced cost re-computed for berth $\bar{k}$ is strictly positive, then the column is duplicated and added to the master problem for berth $\bar{k}$ also. This strategy provides a fast column generation, avoiding the time-consuming pricing subproblem for berth $\bar{k}$.

Incremental heuristic dynamic programming. The basic idea is to incrementally strengthen the relaxed dominance rule introduced for the heuristic pricing, in order to increase the probability of finding a positive reduced cost column and therefore avoid calling exact DP. We define the set of *critical vertices* $\tilde{N} \subset N$ for which exact dominance is required, similarly to decremental state space relaxation (Righini and Salani 2008); the set $\tilde{N}$ is initialized with the empty set, and it is iteratively incremented until a given percentage δ of vertices is reached. At each iteration, $\beta|N|$ critical vertices are chosen from the set $N \backslash \tilde{N}$. Vertices with the greatest associated nonnegative dual price are chosen. The dominance rule is the one described in §3.1, except for the definition of set S: a vertex j belongs to S if it is visited by the partial path and if $j \in \tilde{N}$. The first iteration, when $\tilde{N} = \{\varnothing\}$, corresponds to the heuristic DP algorithm outlined at the beginning of this subsection; the special case of $\delta = 1$ corresponds to exact dynamic programming. After some preliminary experiments, we fix $\beta = 0.2$ and $\delta = 0.4$ in our tests.

4.2.3. Dual Stabilization. Column generation is known to suffer slow convergence (*tailing-off effect*) mainly due to a stability problem. Degeneracy of the master problem implies an infinite number of dual optimal solutions: the simplex method typically provides an extreme dual optimal vector, whereas interior dual vectors could be more suitable for generating good paths in the pricing subproblem. Stabilization methods try to overcome this issue by providing a better approximation of optimal dual values (du Merle et al. 1999; Rousseau, Gendreau, and Feillet 2007).

Our stabilized version of column generation is inspired by Addis, Carello, and Ceselli (2012). The basic idea is the following: a dual optimal solution π to the restricted master problem can be either feasible, and thus optimal, or infeasible for the dual of the complete master problem. We are mainly interested in pricing out with a dual vector close to the optimal dual, thus close to feasibility.

We define the *stability center* $\bar{\pi}$ that represents our current best guess for the optimal dual. At each iteration of column generation, we modify the dual vector provided by the restricted master problem and we obtain a new vector $\tilde{\pi}$ that we use in the pricing problem. The updated formula is clear and simple:

$$\tilde{\pi} = \alpha \pi + (1 - \alpha) \bar{\pi}, \tag{44}$$

where α is a parameter between 0 and 1. At every iteration of column generation the value of α is initialized to $\alpha_0 = 0.5$ and it is increased by a step of 0.1 until positive reduced cost columns are found. The process is repeated until $\alpha = 1$ and no positive reduced cost columns can be found.

4.2.4. Primal Heuristic. Integer feasible solutions are rarely produced in column generation, as the optimal solutions of restricted master problems are typically fractional. Therefore we implement a primal heuristic within the branch and price scheme to identify feasible integer solutions. The main purpose is to improve the primal bound, and thus increase the possibility of pruning the search tree.

The heuristic algorithm takes as input a fractional optimal solution to a restricted master problem and identifies the variable $s^*_{r_k}$ with the highest fractional value strictly lower than 1; variable $s^*_{r_k}$ is set equal to 1 and the linear program is solved again. The procedure is repeated until either an integer solution is found or the linear problem becomes infeasible.

Although very simple, the primal heuristic proves to be helpful in finding integer solutions especially for larger instances. This heuristic is not meant to be a stand-alone algorithm to solve the TBAP but rather a component of the entire exact approach.

4.2.5. Initialization. The master problem is initialized with a set of artificial variables that satisfy constraints (2), (5)–(7).

A second initialization makes use of the solution provided by the heuristic algorithm of Giallombardo et al. (2010): columns associated to the initial solution are added to the master problem at the root node.

5. Computational Results

The branch and price algorithm for the TBAP is implemented in C++ and compiled with gcc 4.1.2.

All restricted master problems are solved using ILOG CPLEX version 12. Computational experiments are run under a linux operating system on a 2 GHz Intel processor equipped with 2 GB of RAM. The time limit is set to three hours for all tests.

5.1. Instances

Computational experiments are performed on instances derived by the test set introduced by Giallombardo et al. (2010). We consider instances with 10 to 20 vessels and three to five berths over a time horizon of one week, resulting in classes 10×3, 15×3 and 20×5. In particular, class 15×3 is obtained by considering the first 15 vessels and the first three berths of the class 20×5.

For each class, four scenarios of traffic volume, denoted by $H1$, $H2$, $L1$, and $L2$, are tested: the traffic volume is given by the total number of containers exchanged between vessels (input parameters f_{ij}) and is mostly influenced by the proportion between mother vessels and feeders, since mother vessels present a number of loading/unloading containers on average higher than feeders. The name of the instance indicates the traffic volume, high ($H1$, $H2$) or low ($L1$, $L2$), and the number of feasible quay crane assignment profiles for each vessel ($p10$, $p20$, or $p30$). Time windows for the ships' service time are generated according to the historical data, and berths are assumed to be available for the whole time horizon.

We also consider an additional set of instances for class 20×5 where the time horizon is shortened from seven to four working days; this new class is denoted by the suffix $4d$ in the instance name.

5.2. Branch and Price Results

We compare our branch and price algorithm to the mixed integer linear programming (MILP) formulation of Giallombardo et al. (2010) solved by CPLEX. The results presented in Giallombardo et al. (2010) have been re-performed on our machines with CPLEX version 12.

Tables 1 and 2 report results for instances with 10 vessels and three berths over a time horizon of one week. Table 1 provides a comparison between the upper bound of the linear relaxation of the original MILP formulation (z_{LP}) and the upper bound obtained via Dantzig-Wolfe reformulation (z_{DW}), i.e., the optimal value of the master problem at the end of the root node. Computational times are not reported, as they are negligible for both cases. Column $\%z_{DW}$ reports the percentage of the bound improvement, which is always less than 0.5%, thus not very significant. However, the DW formulation proves to be much stronger than MILP when embedded into a branch and bound framework. Indeed, we notice that, after three hours of computation, the CPLEX bound is unchanged, despite of the depth of the search tree and the several branching decisions that have been made. On the contrary, it is often sufficient to perform a few branching decisions to see an improvement in the bound provided by the master problem.

Table 1 **Linear Relaxation Results for 10 Ships and Three Berths Over One Week**

	Root node			Search tree		
10×3 instance	z_{LP}	z_{DW}	$\%z_{DW}$ (%)	$z_{LP}^{B\&B}$	$z_{DW}^{B\&P}$	$\%z_{DW}^{B\&P}$ (%)
10_3_$H1$_p10	800,614	797,594	0.38	800,614	790,735	1.22
10_3_$H1$_p20	800,890	797,870	0.38	800,890	791,011	1.20
10_3_$H1$_p30	800,924	798,136	0.35	800,924	791,045	1.23
10_3_$H2$_p10	740,947	738,540	0.32	740,947	733,276	1.03
10_3_$H2$_p20	741,487	739,190	0.31	741,487	735,646	0.77
10_3_$H2$_p30	741,523	739,417	0.28	741,523	735,682	0.78
10_3_$L1$_p10	519,319	518,334	0.19	519,319	515,902	0.63
10_3_$L1$_p20	519,354	518,750	0.12	519,354	518,049	0.19
10_3_$L1$_p30	519,389	518,785	0.12	519,389	518,084	0.25
10_3_$L2$_p10	568,152	566,976	0.21	568,152	564,831	0.58
10_3_$L2$_p20	568,188	567,012	0.21	568,188	564,867	0.57
10_3_$L2$_p30	568,224	567,146	0.19	568,224	564,903	0.58

The superiority of our approach is clearly shown in Table 2, which compares our branch and price algorithm to the general-purpose MIP solver of CPLEX on instances with 10 vessels; the root node initialization relies either on the artificial variables only ($B\&P$) or on the solution provided by the heuristic of Giallombardo et al. (2010) ($B\&P + INIT$). Column *opt_sol* reports the value of the optimal solution, column *init_sol* reports the value of the initial solution provided by the heuristic, while column $t(s)$ reports the computational time of branch and price expressed in seconds; the time required by the heuristic algorithm to provide the initial solution is denoted by $t(init)$, while the total computational time (initialization step + branch and price) is denoted by $t(tot)$. The best solution found by CPLEX in three hours of computation is denoted by *best_sol*; the gap provided by CPLEX is denoted by *gap_cplex* while the gap with respect to the optimal solution is denoted by *gap_opt*.

The branch and price algorithm clearly outperforms CPLEX: it always provides the optimal solution in a few seconds, whereas CPLEX often produces feasible solutions within a gap of 3%. In three cases, CPLEX cannot find a feasible solution in three hours. Furthermore, we remark that CPLEX finds the optimal solution but fails to prove optimality for instances 10_3_$L1$_p10 and 10_3_$L2$_p10 because of the poor linear relaxation bound, which cannot be improved during the search in the branch and bound tree.

Remarkably, for this class of instances, our branch and price shows computational times comparable (or even smaller) than the heuristic algorithm, while

Table 2 Branch and Price Results for 10 Ships and Three Berths Over One Week

10×3	CPLEX (3 h)			*B&P*		*B&P + INIT*			
instance	*best_sol*	*gap_cplex* (%)	*gap_opt* (%)	*opt_sol*	*t*(*s*)	*init_sol*	*t*(*s*)	*t*(*init*)	*t*(*tot*)
10_3_*H*1_*p*10	×	∞	∞	790,735	21	786,439	16	7	23
10_3_*H*1_*p*20	×	∞	∞	791,011	25	785,460	15	21	36
10_3_*H*1_*p*30	780,722	2.59	1.30	791,045	10	784,658	25	39	64
10_3_*H*2_*p*10	712,669	3.97	2.81	733,276	2	732,101	7	8	15
10_3_*H*2_*p*20	×	∞	∞	735,646	7	729,472	9	20	29
10_3_*H*2_*p*30	723,818	2.45	1.61	735,682	9	727,443	8	33	41
10_3_*L*1_*p*10	515,902	0.66	0.00	515,902	7	513,941	11	7	18
10_3_*L*1_*p*20	515,991	0.65	0.40	518,049	5	513,847	11	18	29
10_3_*L*1_*p*30	513,731	1.10	0.84	518,084	27	509,617	125	37	162
10_3_*L*2_*p*10	564,831	0.59	0.00	564,831	9	560,915	8	8	16
10_3_*L*2_*p*20	561,504	1.19	0.60	564,867	7	559,595	19	18	37
10_3_*L*2_*p*30	559,389	1.58	0.98	564,903	8	556,998	20	36	56

Table 3 Linear Relaxation Results for 15 Ships and Three Berths Over One Week

15×3 instance	z_{LP}	z_{DW}	$\%z_{DW}$ (%)	*t*(*s*)
15_3_*H*1_*p*10	1,189,962	1,183,344	0.56	11
15_3_*H*2_*p*10	1,292,357	1,285,075	0.56	11
15_3_*L*1_*p*10	1,110,159	1,105,395	0.43	7
15_3_*L*2_*p*10	899,876	896,483	0.38	320

ensuring optimality of the solutions. Solutions provided by the heuristic are always improved. As a result, the heuristic initialization slows down our branch and price algorithm. Still, the superiority with respect to CPLEX is evident.

Computational results for instances with 15 vessels and three berths over a time horizon of one week are reported in Tables 3 and 4. The improvement of the linear relaxation bound is comparable to the one obtained for class 10×3 (about 0.5%), and the computational time, although not negligible, is still reasonable (except for instance 15_3_*L*2_*p*10). Nevertheless, the superiority of the Dantzig-Wolfe reformulation is clearly shown when embedded in branch and price. Indeed, our branch and price algorithm always finds the optimal solution, whereas CPLEX is only able to provide a feasible solution for instance 15_3_*H*2_*p*10 within three hours of computational time. Although the computational time has increased with respect to class 10×3, the heuristic initialization (columns *B&P + INIT*) proves to be very effective in speeding up the whole procedure: computational time is more than halved for instances 15_3_*H*1_*p*10, 15_3_*H*2_*p*10, and 15_3_*L*2_*p*10 and reduced by a third for instance 15_3_*L*1_*p*10 with respect to basic initialization with artificial variables (columns *B&P*). We remark that the exact algorithm always improves the solution provided by the heuristic of Giallombardo et al. (2010). Additionally, for instance 15_3_*L*1_*p*10, the heuristic with standard settings failed to produce a feasible result (denoted with an asterisk in the table). A modified version of the heuristic of Giallombardo et al. (2010) which requires a higher computational time has been used to feed the master problem.

Tables 5 and 6 report results for instances with 20 vessels and five berths over a time horizon of four days. The linear relaxation bound is improved on average by 2% and the root node is closed in about five minutes (on average). In Table 6 we report the best solutions found after three hours of computation: we notice that the increased size of the problem affects the efficiency of the algorithm, since no instance is solved at optimality. However, branch and price performs significantly better than CPLEX: CPLEX is not able to provide any feasible solution, whereas the branch and price algorithm always provides a solution, with a gap between 3% and 5%. We remark that the initial solutions provided by the heuristic are not improved by the branch and price. This was somehow expected as most of the computational time spent by the branch and price is devoted to the improvement of the dual bound rather than on finding feasible primal solutions.

Table 4 Branch and Price Results for 15 Ships and Three Berths Over One Week

15×3	CPLEX (3 h)			*B&P*		*B&P + INIT*			
instance	*best_sol*	*gap_cplex* (%)	*gap_opt* (%)	*opt_sol*	*t*(*s*)	*init_sol*	*t*(*s*)	*t*(*init*)	*t*(*tot*)
15_3_*H*1_*p*10	×	∞	∞	1,170,783	3,507	1,163,063	1,448	34	1482
15_3_*H*2_*p*10	1,250,124	3.27	1.61	1,272,236	3,787	1,265,782	1,673	32	1,704
15_3_*L*1_*p*10	×	∞	∞	1,098,411	1,203	1,095,021*	898	116	1,014
15_3_*L*2_*p*10	×	∞	∞	890,211	8,975	888,111	3,555	28	3,583

Table 5 **Linear Relaxation Results for 20 Ships and Five Berths Over Four Days**

20×5 instance	z_{LP}	z_{DW}	$\%z_{DW}$ (%)	$t(s)$
20_5_*H*1_*p*10_4*d*	1,383,614	1,356,460	1.96	315
20_5_*H*2_*p*10_4*d*	1,474,082	1,444,042	2.04	136
20_5_*L*1_*p*10_4*d*	1,298,356	1,274,821	1.81	483
20_5_*L*2_*p*10_4*d*	1,103,212	1,084,936	1.66	373

Table 7 **Linear Relaxation Results for 20 Ships and Five Berths Over One Week**

20×5 instance	z_{LP}	z_{DW}	$\%z_{DW}$ (%)	$t(s)$
20_5_*H*1_*p*10	1,383,614	1,369,818	1.00	721
20_5_*H*2_*p*10	1,474,082	1,459,341	1.00	504
20_5_*L*1_*p*10	1,298,356	1,287,080	0.87	520
20_5_*L*2_*p*10	1,103,212	1,094,480	0.79	640

Finally, Tables 7 and 8 report the results for instances with 20 vessels and five berths over a time horizon of one week. In Table 7 we can observe that the improvement in terms of the linear relaxation bound has slightly decreased (from 2% to about 1% on average) compared to the four-day case. Remarkably, the MILP linear relaxation bound is unchanged with respect to the four-day case, despite the different time horizon and modified time windows. This gives an insight to how "fractional" the MILP linear relaxation solution is. Furthermore, the computational effort required by column generation has increased, as it takes about 10 minutes on average to close the root node. The MILP linear relaxation is solved in fractions of a second. Despite the longer computational time required to solve its linear relaxation, the DW formulation proved to be stronger when embedded in a branch and bound scheme.

Table 8 compares the best solutions found by CPLEX and our branch and price algorithm after three hours of computation. Again, branch and price clearly outperforms CPLEX: it always provides a feasible solution with a gap between 2% and 3%, whereas CPLEX finds a feasible solution only for instance 20_5_*L*1_*p*10, with higher gap. The initial solutions provided by the heuristic are slightly improved by the branch and price algorithm even if just marginally.

Summing up, computational experiments show that our specialized branch and price algorithm outperforms the general-purpose solver in terms of quality of solutions and computational time. The results confirm that designing sophisticated algorithms and exploiting the problem structure is crucial when tackling large-scale optimization problems such as the TBAP. The dual bound provided at the end of the search tree of the branch and price algorithm is always substantially better than that of the MILP formulation embedded in a branch and cut algorithm (CPLEX).

Most of the implemented accelerating techniques concern the pricing problem. This is motivated by the fact that preliminary results produced with a "basic" implementation of the branch and price algorithm (that included only heuristic pricing as an accelerating technique) point out that about 99% of the computational time was spent in the pricing. The design of sophisticated and specialized techniques for the pricing problem is extremely successful: we reduced the computational time by 90% on average, as shown in Table 9. For the basic and the specialized implementation of the branch and price algorithm we report the computational time ($t(s)$) and the percentage of time spent in solving the pricing problem ($\%pricing$). The results confirm the large increase in speed of the solution process produced by the designed accelerating techniques.

5.3. Hierarchical vs. Integrated Planning

The exact branch and price algorithm enables us to perform a comparative analysis between hierarchical and integrated optimization approaches for the tactical berth allocation problem which was not possible before. Indeed, heuristic solutions without any quality guarantee cannot provide solid comparison terms.

In hierarchical planning, berth allocation is solved first, according to an estimated handling time for the vessels; in a second stage, the quay crane assignment is performed on the resulting berth allocation plan. On the contrary, in the integrated planning approach, the two problems are solved simultaneously.

Remarkably, our algorithm can be adapted to solve the berth allocation problem only. Whereas approaches based on column generation have been

Table 6 **Branch and Price Results for 20 Ships and Five Berths Over Four Days**

20×5	CPLEX (3 h)	*B&P* + *INIT* (3 h)			
instance	gap	*best_sol*	gap	*init_sol*	$t(init)$
20_5_*H*1_*p*10_4*d*	∞	1,305,216	3.78	1,305,216	65
20_5_*H*2_*p*10_4*d*	∞	1,381,241	4.35	1,381,241	311
20_5_*L*1_*p*10_4*d*	∞	1,231,384	3.41	1,231,384	65
20_5_*L*2_*p*10_4*d*	∞	1,050,171	3.20	1,050,171	70

Table 8 **Branch and Price Results for 20 Ships and Five Berths Over One Week**

20×5	CPLEX (3 h)	*B&P* + *INIT* (3 h)			
instance	gap	*best_sol*	gap	*init_sol*	$t(init)$
20_5_*H*1_*p*10	∞	1,337,077	2.39	1,335,625	94
20_5_*H*2_*p*10	∞	1,429,249	2.06	1,428,889	84
20_5_*L*1_*p*10	5.12	1,258,150	2.25	1,258,097	100
20_5_*L*2_*p*10	∞	1,070,543	2.19	1,070,543	89

Table 9 Reduction of Computational Time Obtained with the Accelerating Techniques

10 × 3	Basic *B&P*		Improved *B&P*		
instance	*t*(*s*)	%*pricing* (%)	*t*(*s*)	%*pricing* (%)	Speed up(%)
10_3_*H*1_*p*10	114	97	21	10	82
10_3_*H*1_*p*20	995	97	25	12	97
10_3_*H*1_*p*30	557	99	10	18	98
10_3_*H*2_*p*10	12	82	2	10	83
10_3_*H*2_*p*20	29	90	7	11	76
10_3_*H*2_*p*30	25	92	9	13	65
10_3_*L*1_*p*10	4,054	99	7	42	100
10_3_*L*1_*p*20	761	99	5	62	99
10_3_*L*1_*p*30	470	99	27	93	94
10_3_*L*2_*p*10	4,697	99	9	54	100
10_3_*L*2_*p*20	1,573	99	7	62	100
10_3_*L*2_*p*30	2,680	99	8	63	100

recently proposed by Mauri, Oliveira, and Lorena (2008) and Buhrkal et al. (2011), it represents the first exact branch and price algorithm for the BAP. On the other hand, the quay crane assignment is easily solved by a general-purpose solver. For all the details concerning this experiment, including the BAP and QCAP formulations, we refer the reader to Vacca (2011).

Handling time estimation. In the hierarchical approach, the handling time of vessels is assumed to be known in advance. In practice, the expected handling time is provided by terminal planners, who base their estimations on quantitative data such as the vessel's workload, average QC productivity, availability of transfer equipment, the vessel's priority, as well as on their experience. In particular, some extra time can also be included in the estimation in order to guarantee more robustness and flexibility in the schedule.

In our experiment, among the available TBAP data, we are given the set of feasible quay crane assignment profiles defined for every vessel and known in advance; in particular, we know the duration in terms of working shifts of every feasible QC profile, expressed by the input parameter t_i^p.

To start the entire hierarchical optimization process, we produce two different estimations for the handling time, both motivated by the practice.

Scenario A. For every vessel, the handling time is given by the longest feasible quay crane assignment profile.

Scenario B. For mother vessels, the handling time is given by the shortest feasible quay crane assignment profile, whereas for feeders, the handling time is given by the longest feasible quay crane assignment profile.

Scenario A is very conservative and somehow represents the worst-case scenario, when all vessels are serviced at the lowest rate. However, this handling time estimation may be useful in producing robust schedules. Scenario B can be considered more realistic, since mother vessels typically have higher priority than feeders. In particular, we expect the terminal to operate mother vessels as fast as possible to minimize their stay at the port. Both scenarios are realistic and reasonable in practice.

Comparative analysis. Table 10 compares the optimal solutions for the hierarchical approach under scenarios A and B to the integrated TBAP approach. We consider instances with 10 vessels and three berths over a time horizon of one week. For all solutions we report the value of the objective function (opt_{sol}), the number of berths used (K), and the computational time in seconds ($t(s)$). Columns "%(A)," "%(B)" indicate the improvement of the integrated solution with respect to the hierarchical approach under scenarios A, B, respectively.

As expected, the integrated TBAP provides better solutions, although it seems that these specific instances do not allow for a significant gain in terms of objective function. The average improvement is 0.68% over scenario A and 0.36% over scenario B.

Table 10 Optimal Solutions for 10 Vessels and Three Berths Over One Week

10 × 3	Scenario A			Scenario B			Integrated TBAP				
instance	*opt_sol*	*K*	*t*(*s*)	*opt_sol*	*K*	*t*(*s*)	*opt_sol*	*K*	*t*(*s*)	%(A) (%)	%(B) (%)
10_3_*H*1_10	786,841	2	14	789,478	2	19	790,735	2	21	0.49	0.16
10_3_*H*1_20	787,689	2	14	789,754	2	14	791,011	2	25	0.42	0.16
10_3_*H*1_30	787,723	2	9	789,788	2	8	791,045	2	10	0.42	0.16
10_3_*H*2_10	730,702	2	8	733,276	2	3	733,276	2	2	0.35	0.00
10_3_*H*2_20	730,418	2	6	732,659	2	12	735,646	2	7	0.72	0.41
10_3_*H*2_30	730,454	2	11	732,695	2	14	735,682	2	9	0.72	0.41
10_3_*L*1_10	513,661	2	5	515,017	2	5	515,902	2	7	0.44	0.17
10_3_*L*1_20	513,696	2	5	515,052	2	6	518,049	2	5	0.85	0.58
10_3_*L*1_30	513,731	2	7	515,087	2	4	518,084	2	27	0.85	0.58
10_3_*L*2_10	559,683	2	7	561,705	2	7	564,831	2	9	0.92	0.56
10_3_*L*2_20	559,719	2	4	561,741	2	10	564,867	2	7	0.92	0.56
10_3_*L*2_30	559,755	2	4	561,777	2	6	564,903	2	8	0.92	0.56

Table 11 **Optimal Solutions for 10 Vessels and Three Berths Over Four Days**

10×3	Scenario A			Scenario B			Integrated TBAP				
instance	*opt_sol*	*K*	*t*(*s*)	*opt_sol*	*K*	*t*(*s*)	*opt_sol*	*K*	*t*(*s*)	%(A) (%)	%(B) (%)
10_3_*H*1_10_4*d*	774,211*	3	45	774,681*	3	68	777,398	3	39	0.41	0.35
10_3_*H*1_20_4*d*	776,331	3	10	777,919*	3	31	779,674	3	47	0.43	0.23
10_3_*H*1_30_4*d*	776,365	3	13	778,233*	3	15	782,300	3	66	0.76	0.52
10_3_*H*2_10_4*d*	718,900	3	8	719,739*	3	23	722,431	3	28	0.49	0.37
10_3_*H*2_20_4*d*	719,987	3	11	720,223*	3	60	724,345	3	36	0.61	0.57
10_3_*H*2_30_4*d*	719,701	3	13	720,223*	3	61	725,585	3	28	0.82	0.74
10_3_*L*1_10_4*d*	507,422	3	7	508,457*	3	7	512,533	2	4	1.01	0.80
10_3_*L*1_20_4*d*	507,304	3	7	508,505	3	6	512,533	2	19	1.03	0.79
10_3_*L*1_30_4*d*	507,339	3	6	508,540	3	6	512,991	2	10	1.11	0.88
10_3_*L*2_10_4*d*	553,971	3	9	555,978*	3	21	558,750	2	16	0.86	0.50
10_3_*L*2_20_4*d*	554,380	3	5	556,272	3	4	558,786	2	25	0.79	0.45
10_3_*L*2_30_4*d*	554,380	3	4	556,280	3	4	558,822	2	6	0.80	0.46

The computational effort required by the integrated approach is comparable to that of the hierarchical approach.

These results seem to indicate that the tested instances are not very tight. Therefore, we perform additional tests by defining more congested instances. Such instances are obtained by reducing the time horizon from seven to four working days; in order to maintain feasibility, vessels' time windows are also relaxed and the quay crane capacity increases from $Q = 8$ to $Q = 10$.

Results for instances with a time horizon of four days are illustrated in Table 11. As soon as instances become more congested, the hierarchical approach clearly shows its drawbacks: first of all, it is not always able to provide a feasible solution. More specifically, the quay crane assignment may not be feasible for a given berth allocation plan, because of the QC capacity constraint (scenario B), or berths are not sufficient to accommodate all incoming vessels (scenario A).

To produce a feasible solution for the hierarchical approach, we implemented a semi-automatic method which consists of changing the selected profile prior to the berth allocation. We have two cases:

- Insufficient number of berths (scenario A): in this case the longest quay crane profile is substituted with the shortest available one. In case of ex-æquo, the profile is chosen randomly.
- Violation of QC capacity constraints: the profile demanding the highest number of quay cranes for a certain time period is substituted with the longest one for the same vessel. In case of ex-æquo, the profile is chosen randomly.

Once the selected profile is changed, the berth allocation algorithm is run again to search for a feasible solution.

Solutions in Table 11 marked with an asterisk ($*$) required the described semi-automatic intervention. The computational time reported in the table corresponds to the overall computational time required for the berth allocation. The average number of iterations required to resolve the infeasibilities is 2.5.

The integrated TBAP provides better solutions; gains against the hierarchical approach are slightly higher for these instances. The average improvement is 0.76% over scenario A and 0.56% over scenario B. The computational effort required by the integrated approach is comparable to that of the hierarchical approach. We remark that the computational time to solve instances marked with an asterisk is just an underestimation of the total computational time requested as it considers automatic procedures only.

The method to deal with infeasible solutions could be improved. It is out of the scope of this paper to provide a fully functional hierarchical planning framework for berth allocation and quay crane assignment. Indeed, we remark that the integrated approach always finds the optimal solution for all tested instances. It is interesting to notice that the integrated solution makes use, in some cases, of one berth less than the solution provided by the hierarchical approach.

Summing up, computational results confirm the added value of integration in terms of cost reduction and efficient use of resources. The main outcome of the analysis is that the strong assumptions made by the sequential approach may prevent us finding any feasible solution automatically unless an integrated planning framework is in place. On the contrary, the integrated approach always finds the optimal one. This occurs especially for congested instances. Furthermore, the additional effort required to solve the integrated problem is moderate.

6. Conclusions

In this paper we have presented a new model and a branch and price algorithm for the tactical berth allocation problem.

The model is based on an exponential number of variables and it is solved via column generation. To obtain integer solutions, a branch and price scheme has been implemented, introducing several accelerating techniques specifically designed for our problem.

Computational tests prove that our exact algorithm outperforms commercial solvers; especially on small instances, branch and price always provides optimal solutions relatively quickly. The problem size still represents an issue, and additional advanced techniques should be further investigated to overcome the complexity of the problem.

Furthermore, the proposed branch and price algorithm enables us to provide an experimental comparison between the traditional hierarchical approach (that sequentially solves the berth allocation and the quay crane assignment) and the integrated planning approach. Computational experiments confirm the added value of integration in terms of cost reduction and efficient use of resources.

Most of the presented accelerating techniques concern the pricing problem and prove to be very successful, reducing the computational time in the pricing by 90% on average. Future research should focus on improving the master problem. Alternative linearizations of the quadratic objective function should be investigated. Preliminary results obtained including cutting planes (specifically, lifted cover inequalities) did not provide any improvement over the master problem original formulation.

References

Addis B, Carello G, Ceselli A (2012) Exactly solving a two-level location problem with modular node capacities. *Network* 59(1):161–180.

Barnhart C, Belobaba P, Odoni AR (2003) Applications of operations research in the air transport industry. *Transportation Sci.* 37(4):368–391.

Beasley JE, Christofides N (1989) An algorithm for the resource constrained shortest path problem. *Networks* 19(3):379–394.

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

Buhrkal K, Zuglian S, Ropke S, Larsen J, Lusby R (2011) Models for the discrete berth allocation problem: A computational comparison. *Transportation Res. Part E* 47(4):461–473.

Choo S, Klabjan D, Simchi-Levi D (2010) Multiship crane sequencing with yard congestion constraints. *Transportation Sci.* 44(1):98–115.

Christiansen M, Fagerholt K, Ronen D (2004) Ship routing and scheduling: Status and perspectives. *Transportation Sci.* 38(1):1–18.

Cordeau JF, Laporte G, Legato P, Moccia L (2005) Models and tabu search heuristics for the berth-allocation problem. *Transportation Sci.* 39(4):526–538.

Dell'Amico M, Righini G, Salani M (2006) A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Sci.* 40(2):235–247.

du Merle O, Villeneuve D, Desrosiers J, Hansen P (1999) Stabilized column generation. *Discrete Math.* 194:229–237.

Gélinas E, Desrochers G, Desrosiers J, Solomon M (1995) A new branching strategy for time constrained routing problems with application to backhauling. *Ann. Oper. Res.* 61(1):91–109.

Giallombardo G, Moccia L, Salani M, Vacca I (2010) Modeling and solving the tactical berth allocation problem. *Transportation Res. Part B* 44(2):232–245.

Grønhaug R, Christiansen M, Desaulniers G, Desrosiers J (2010) A branch-and-price method for a liquefied natural gas inventory routing problem. *Transportation Sci.* 44(3):400–415.

Imai A, Nishimura E, Papadimitriou S (2001) The dynamic berth allocation problem for a container port. *Transportation Res. Part B* 35(4):401–417.

Imai A, Chen HC, Nishimura E, Papadimitriou S (2008) The simultaneous berth and quay crane allocation problem. *Transportation Res. Part E* 44(5):900–920.

ISL (2009) Shipping statistics and market review (SSMR)—Short comment, Vol. 53. Institute of Shipping Economics and Logistics, http://www.isl.org.

Liberatore F, Righini G, Salani M (2011) A column generation algorithm for the vehicle routing problem with soft time windows. 4*OR*: *A Quart. J. Oper. Res.* 9(1):49–82.

Lübbecke ME, Desrosiers J (2005) Selected topics in column generation. *Oper. Res.* 53(6):1007–1023.

Mauri G, Oliveira A, Lorena L (2008) A hybrid column generation approach for the berth allocation problem. van Hemert J, Cotta C, eds. *Evolutionary Computation in Combinatorial Optimization, Lecture Notes in Computer Science*, Vol. 4972. (Springer, Berlin/Heidelberg), 110–122.

Meisel F, Bierwirth C (2009) Heuristics for the integration of crane productivity in the berth allocation problem. *Transportation Res. Part E* 45(1):196–209.

Park YM, Kim KH (2003) A scheduling method for berth and quay cranes. *OR Spectrum* 25(1):1–23.

Righini G, Salani M (2006) Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optim.* 3(3):255–273.

Righini G, Salani M (2008) New dynamic programming algorithms for the resource constrained shortest path problem. *Networks* 51(3):155–170.

Rousseau L-M, Gendreau M, Feillet D (2007) Interior point stabilization for column generation. *Oper. Res. Lett.* 35(5):660–668.

Stahlbock R, Voss S (2008) Operations research at container terminals: A literature update. *OR Spectrum* 30(1):1–52.

Steenken D, Voss S, Stahlbock R (2004) Container terminal operation and operations research—A classification and literature review. *OR Spectrum* 26(1):3–49.

UNCTAD (2009) Review of maritime transport. United Nations conference on trade and development, http://www.unctad.org.

Vacca I (2011) Container terminal management: Integrated models and large-scale optimization algorithms. Ph.D. thesis, Ecole Polytechnique Fédérale de Lausanne.

Vacca I, Bierlaire M, Salani M (2007) Optimization at container terminals: Status, trends and perspectives. *Proc. 7th Swiss Transport Res. Conf. (STRC)*, http://www.strc.ch/conferences/2007/2007_vacca.pdf.

Vacca I, Salani M, Bierlaire M (2010) Optimization of operations in container terminals: Hierarchical vs integrated approaches. *Proc. 10th Swiss Transport Res. Conf. (STRC)*, http://www.strc.ch/conferences/2010/Vacca.pdf.

Vis IFA, de Koster R (2003) Transshipment of containers at a container terminal: An overview. *Eur. J. Oper. Res.* 147(1):1–16.