AN EFFICIENT BLOCK-BASED HEURISTIC METHOD FOR STOWAGE PLANNING OF LARGE CONTAINERSHIPS WITH CRANE SPLIT CONSIDERATION

Xiantao Xiao(a), Malcolm Yoke Hean Low(b), Fan Liu(c), Shell Ying Huang(d), Wen Jing Hsu(e), Zhengping Li(f)

(a,b,c,d,e)School of Computer Engineering, Nanyang Technological University, Singapore 639798
(f)Singapore Institute of Manufacturing Technology, 71 Nanyang Drive, Singapore 638075

(a)xtxiao@ntu.edu.sg, (b)yhlow@ntu.edu.sg (c)liufan@ntu.edu.sg
(d)assyhuang@ntu.edu.sg (e)hsu@ntu.edu.sg (f)zpli@SIMTech.a-star.edu.sg

ABSTRACT
This paper presents an efficient block-based heuristic method for stowage planning of large containerships on a multi-port voyage, subject to a set of constraints. We describe in detail some issues in stowage planning problem, including ship profile, constraints, crane split and rehandle. A "block stowage" heuristic approach is developed to generate a set of stowage plans for a multi-port voyage automatically. Finally, we present the results with a practical test case, and analyze the tradeoff between crane split and rehandles in stowage planning.

Keywords: stowage planning, crane split, block stowage, rehandle

1. INTRODUCTION
Stowage planning is a crucial function for the container transportation business and it greatly affects a shipping line's operating cost. Existing stowage planning process in all major shipping lines worldwide is mainly carried out by human planners. The quality of the stowage plan generated depends very much on the experience of the stowage planners, who have gone through several years training onboard ships. With the capacity of containership rising from the relatively small 350 TEUs (Twenty Foot Equivalent Unit) to ten thousand TEUs, shipping lines are facing increasing challenge to generate workable and cost-saving stowage plans for their containerships as they move between ports.

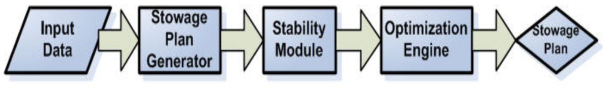

Figure 1: An automated system of stowage planning

The subject of our study is to build a fully automated system of stowage planning for large containerships on their multi-port voyages. As shown in Figure 1, the framework of the system is: given the input data, the stowage plan generator uses a list of heuristic strategies to generate a feasible stowage plan that fulfills a list of constraints. Then the stability module checks the stability of the feasible stowage plan and adjusts it to satisfy the stability requirements. Finally, the optimization engine takes the adjusted feasible stowage plan and optimizes it based on some specific objectives (such as minimizing the number of rehandles). In this paper, we present our current approach for the stowage plan generator. The work with reference to other parts of the system is still in progress and will be reported in the future.

The stowage plan generation process directly affects the number of rehandling and the berthing time of a containership at each port. These two components represent a significant part of the operating cost of the shipping operation. The ship berthing time at a port mainly depends on the crane split. In stowage planning, the requirement of perfect crane split may cause containers destined to a particular port to be stowed using many bays in a ship. This may result in a stowage plan that has unnecessary rehandles due to bays and blocks being used up partially to satisfy crane split requirement. The tradeoffs between rehandle and crane split will be analyzed in the later section of the paper.

The remaining part of the paper is organized as follows. Section 2 reviews the related literature. In section 3, we present some definitions and constraints of stowage planning. Our proposed block stowage algorithm is presented in Section 4. In Section 5, we give a simple test case and show some computational results. Section 6 concludes the paper.

2. LITERATURE REVIEW
Since the 1970s, the problem related to container stowage planning has been studied by shipping lines and researchers. The existing research works are mostly focused on the container loading problem, which can be formulated into a combinatorial optimization problem. The size of the container loading problem depends on the ship capacity and the shipping demand at each port. Even

for a medium size containership, the problem is nontrivial due to the large number of variables. Moreover, the problem has been proven to be NP-hard, which is to say that it is very unlikely to guarantee an optimal solution in a reasonable processing time. Meanwhile, a few researchers try to develop heuristic methodology to provide workable solutions to the stowage planning. A brief review of recent research follows.

The early study about the container loading problem can be traced back to the work by Aslidis (1989) and Aslidis (1990). The author examined the stack overstowage problem of small size under certain assumption. Aslidis's work leads to a set of heuristic algorithms which were used to solve the container loading problem without considering stability. Another early work was carried out by Imai and Miki (1989) who considered the minimization of loading-related rehandles.

Avriel and Penn (1993) formulated the stowage planning problem into a 0-1 binary linear programming. They found that the general algorithm is too slow even after some preprocessing of the data. Also, Averiel et al. (2000) showed that the stowage planning problem is NP-complete and showed a relation between the stowage problem and the coloring of circle graphs problem.

Wilson and Roach (1999, 2000) developed a methodology for generating computerised stowage plan. The methodology embodies a two stage process to computerised planning. First they use branch-and-bound algorithms for solving the problem of assigning generalized containers to a bay's block in a vessel. In the second step they use a tabu search algorithm to assign specific locations for specific containers. Wilson et al. (2001) also presented a computer system for generating solutions to the stowage pre-planning problem using a genetic algorithm approach. Dubrovsky et al. (2002) used a genetic algorithm technique for minimizing the number of container movements of the stowage planning. The authors developed a compact and efficient encoding of solutions to reduce the search space significantly.

In the papers of Ambrosino et al. (1998, 2004, 2006), the stowage planning problem is called Master Bay Plan Problem (MBPP). Ambrosino and Sciomachen (1998) reported the first attempt to derive some rules for determining good container stowage plans, where a constraint satisfaction approach is used for defining the space of feasible solutions. Ambrosino et al. (2004) described a 0-1 linear programming model for MBPP. They presented an approach consisting of heuristic preprocessing and prestowing procedures that allow the relaxation of some constraints of the exact model. Ambrosino et al. (2006) presented a three phase algorithm for MBPP, which is based on a partitioning procedure that splits the ship into different portions and assigns containers on the basis of their destination. However they assumed that the ship starts its journey at a port and visits a given number of other ports where only unloading operations are allowed, which means that the loading problem is only considered at the first port.

Averiel et al. (1998) dealt with stowage planning in order to minimize the number of rehandles, without considering stability and several other constraints. They presented a 0-1 binary linear programming formulation and found that the optimal solution is quite limited because of the large number of binary variables and constraints needed for the formulation. Consequently, they developed a heuristic procedure called the suspensory heuristic procedure. However, they assumed that the ship only has a large cargo bay without considering the hatch covers and stability.

Since all these studies reviewed were carried out under some simplistic assumptions, they can hardly be applied by shipping companies in real life, especially for large containerships. In this paper, we describe a heuristic stowage planning algorithm that considers existing containership features and constraints to rapidly generate a set of feasible plans for a multi-port voyage.

3. PROBLEM DEFINITION

3.1. The Containership Structure

From the side view of the containership (see Figure 2), a containership contains a number of bays with number increased from bow to stern. In particular, each 40 foot (40') bay is numbered with an even number, i.e. bay 02, 06, 10, etc., while a 40' bay is associated with two 20' bays with two contiguous odd numbers, i.e. bay 06 = bay 05 + bay 07. Usually a bay is divided by hatches into two sections, below deck and above deck.

From the cross section view of a bay (see Figure 3), every bay contains a set of slots. Each slot is identified by three indices:

- *bay*, that gives the bay it is located in;
- *row*, that gives its position relative to the vertical section of the corresponding bay (counted from the center to outside);
- *tier*, that gives its position relative to the horizontal section of the corresponding bay (counted from the bottom to the top).

Usually, the containers are divided by size into two types: 20' container and 40' container. A 20' slot for the stowage of a 20' container (referred to as a Twenty-Foot Equivalent Unit or TEU) is indexed with the number of the corresponding 20' bay; while a 40' slot (usually is yield by two 20' slots) for the stowage of a 40' container is indexed with the number of the corresponding 40' bay. As for the second index, the location has an even number if it is located on the seaside, i.e. row 02, 04, 06, and an odd number if it is located on the yard side, i.e. row 01, 03, 05, etc. Finally, for the third index, the tiers are numbered from the bottom of the containership to the hatch with even number, i.e. tier 02, 04, 06, etc., while in the above deck from hatch to the top of the container ship, the numbers are 82, 84, 86, etc. Thus, for instance, slot 180406 refers to the slot in bay 18, row 04 and tier 06.

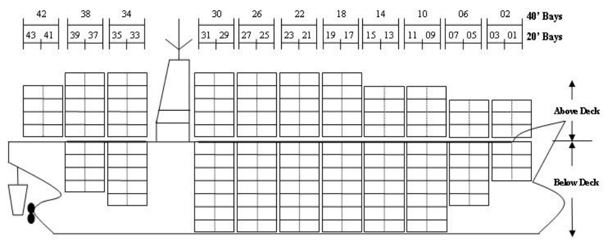
42 38 34 30 26 22 18 14 10 06 02 40' Bays
43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11 09 07 05 03 01 20' Bays
Above Deck
Below Deck

Figure 2: Side view of a containership

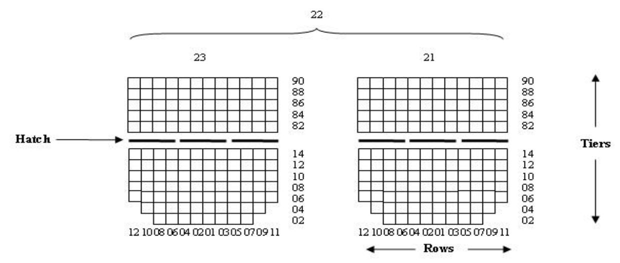

Figure 3: Cross section view of a bay

3.2. The Constraints of Stowage Planning

In stowage planning, a container contains many features, such as port of loading (POL), port of destination (POD), ISO, type, weight, etc. Together with the containership profile, the stowage plan is generated which is subject to many constraints related to the container characteristics and operational instructions. We list the main constraints of stowage planning in the following:

- *Standard container.* The dimension of a 40' container is equivalent to two 20' containers. When a 40' container is stowed into a 40' slot (for instance, slot 180406), the corresponding two 20' slots (slot 170406 and slot 190406) are also occupied and are not available for stowing 20' containers.
- *High cube container.* A 40' high cube container is almost identical to a standard 40' container, except that it is one foot taller. So if one or more 40' high cube containers are stowed into a row below deck, the topmost slot in this row must be left empty. To minimize killing of slots, for below deck, the high cube containers should be stowed in the deep rows (with more tiers).
- *Reefer.* A refrigerated container or reefer is a container used for the transportation of temperature sensitive cargo. Since a reefer relies on external power to maintain the required temperature, it must be stowed into a slot with electrical plug.
- *Hazardous container.* Hazardous container should be subjected to segregation constraints which is provided by the shipping company. Hazardous containers also should be stowed away from the accommodation area and heat source (e.g., engine and fuel tank).
- *Operational constraints.* No container can hang in the air, in other words, a slot below a container cannot be left empty. 20' container cannot be stowed on top of a 40' container. Above deck, no 40' container can be stowed on top of 20' containers.

3.3. Rehandle

Due to the structure of the containership, the containers are stowed in vertical stacks. When a container is unloaded, the containers above it in the same row must be unloaded first. Moreover, if the container is stowed below a hatch, to open the hatch, all containers above this hatch must also be unloaded. In stowage planning, a common situation is that, at port i, the container with POD j (after port i) must be unloaded and reloaded at port i in order to access the container below them with POD i. This is called "overstow" or "forced rehandle". Another situation is that, although a container with POD j does not block any container with POD i, to prevent costlier overstow in future ports or other reasons, the ship planner still decide to unload it and reload it at port i. This is called "voluntary rehandle". Usually, rehandling a container costs tens or hundreds of US dollars. A simple heuristic to reduce the number of forced rehandle is to load the containers in the order of their PODs, i.e. for stowing containers at port i, first load the containers with POD k (port k is the farthest port from port i), then load the containers with POD $k-1$ (the second farthest port), and so on. Finally load the containers with POD $i+1$ (port $i+1$ is the nearest port from port i).

3.4. Crane Split

At a port, the ship will be served by a given number of (usually 3-5) quay cranes to unload and load containers. The bays of the ship will be partitioned into several areas. Each area will be served by one quay crane. This is called crane split. For operating safety, there should be a separation between two adjacent working cranes. The distance of the separation is defined as follows: if a crane is working at bay i, the neighboring crane has to be working at bay $i+8$ or further (see Figure 4). Therefore, if the working areas of two adjacent cranes are too close, one crane has to wait until the other crane finishes its work and moves to the other bay further enough. The waiting time of a crane is called "idle time". A perfect crane split is that all cranes finish their work at the same time with no idle time.

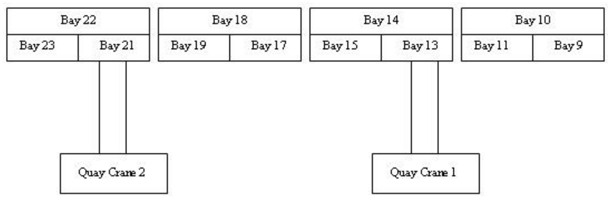

Figure 4: Minimum crane seperation distance

The quality of a crane split is measured by crane intensity (CI). CI is calculated by the following formula: CI equals to the total working time of all cranes divided by the longest crane working time. The duration a ship berthed in a port depends on the completion time of the longest crane and is mainly decided by the crane split.

3.5. The Objective of Stowage Planning

The containership profile containing the locations and the hatches, together with the lists containing all the characteristics of the containers to be loaded at every port in the voyage, are the input data required for generating the stowage plan. The evaluation of a stowage plan can be judged by many considerations, such as stability, handling cost, safety. In this paper, the objective is to generate a set of computerized feasible stowage plans that minimize the number of rehandles and maximize the crane intensity (CI).

4. BLOCK STOWAGE

This section presents a "Block Stowage" approach for the stowage planning problem. The main idea of "Block Stowage" is: first we partition all the locations of a containership into several blocks; then at every port, we divide all the loading containers into different groups by size, type and POD. Instead of stowing the containers one by one, we load the containers group by group into the blocks according to a set of rules. This approach is much more efficient especially for large containerships.

4.1. The Definition of Block

Based on the locations of the hatches in 40' bay, we define two types of block as follows (see Figure 5):

- *Blockbelow* contains the slots below a middle hatch or two symmetric side hatches in a 40' bay.
- *Blockabove* contains the slots above a middle hatch or two symmetric side hatches in a 40' bay.

In order to maintain the balance of the ship in stowage planning, we consider the slots affected by the two side hatches as one block.

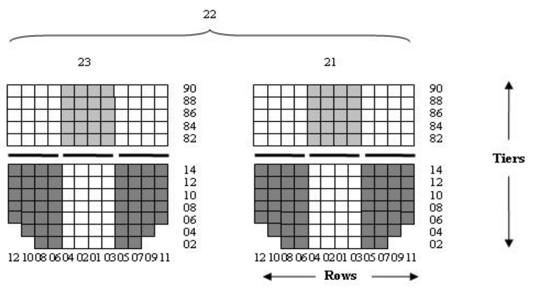

Figure 5: Blockbelow and Blockabove

4.2. The Block Assignment Algorithm

Before producing a stowage plan for a port, we first divide the loading containers into different groups according to their destination port, size and type. If a ship is on a loop voyage of n ports including the current port, there will be up to $n-1$ groups of containers to be loaded and transported to the $n-1$ ports. Group 1 includes the containers going to the farthest port in the loop. Group 2 includes the containers for the second farthest port and group $n-1$ is for the next port from the current port. For example, consider the containers at port C that are ready to be loaded to a ship with a loop voyage A-B-C-D-E-F-G-A. We divide the loading containers by destination port as:

- **Group 1** include the containers with POD B (the farthest port from current port in the voyage),
- **Group 2** include the containers with POD A,
- **Group 3** include the containers with POD G,
- **Group 4** include the containers with POD F,
- **Group 5** include the containers with POD E,
- **Group 6** include the containers with POD D (the farthest port from current port in the voyage).

For convenience of description, we define the "destination port of a block" to be the nearest destination port of the containers stowed in the block. Then we divide the blocks in the containership into different groups according to the destination port of the block in a similar way as we divide the loading containers. An additional group is added for empty blocks (blocks with no containers). When some new containers are stowed into a block, the destination port of the block will be updated. Using the same example of the loop voyage of A-B-C-D-E-F-G-A and port C as the current port, we will have,

Blockbelow [0]: include the Blockbelows which are empty,

Blockbelow [1]: include the Blockbelows whose destination ports are B,

Blockbelow [2]: include the Blockbelows whose destination ports are A,

$\vdots$

Blockbelow [6]: include the Blockbelows whose destination ports are D,

Blockabove [0]: include the Blockaboves which are empty,

Blockabove [1]: include the Blockaboves whose destination ports are B,

Blockabove [2]: include the Blockaboves whose destination ports are A,

$\vdots$

Blockabove [6]: include the Blockaboves whose destination ports are D.

We load the containers in order of group number, i.e. first load Group 1, then Group 2 and so on with Group 6 loaded last. When loading a group, we divide the group into two sets by size: 20' and 40'. According to the stowage constraints, 20' containers cannot be on top of 40' containers. So we load the 20' set prior to the 40' set. When stowing a set of containers, we subdivide the set into different subsets by type, such as normal, empty, reefer, hazardous and out of gauge (OOG), etc.

The sequence of loading different subsets in a 20' set or 40' set is: reefer subset, normal subset, hazardous subset, empty subset and OOG subset. The strategy of stowing a set of containers in the subsets (excluding the reefer subset and OOG subset, because the reefer containers have specific locations in the ship and OOG container cannot be stowed below any container) is expressed as follows (for example, loading a set of containers in Group k):

Step 0. Let $t = k$, go to Step 1.

Step 1. Stow containers into the blocks which are in Blockbelow[t] and above which the blocks are empty. If there are some containers left, go to Step 2.

Step 2. Stow containers into the blocks which are in Blockbelow[0] (empty) and above which the blocks are empty. If there are some containers left, go to Step 3.

Step 3. Stow containers into the blocks which are in Blockabove[t] and below which the blocks are in Blockbelow[t]. If there are some containers left, go to Step 4.

Step 4. Stow containers to the blocks which are in Blockabove[0] and below which the blocks are in Blockbelow[t]. If there are some containers left, go to Step 5.

Step 5. Let $t = t - 1$ (if $t - 1 \geq 1$), go back to Step 1. If there are some containers left and $t = 1$, go to step 6.

Step 6. Stow the remaining containers to those blocks which will cause the least number of rehandles. **End.**

4.3. Considerations for Crane Split

Based on the block stowage assignment algorithm, to generate an ideal crane split, we follow certain rules when stowing containers as follows:

1. *The total move count of two adjacent 40' bays should not exceed the average number of moves per crane.* This constraint is used to balance the loading workload of the crane in the current port taking into consideration that no two cranes can work concurrently in adjacent 40' bays. For example, if the total loading/unloading moves at a port is n with c cranes, the average moves per crane is n/c. Thus the total move count of two adjacent 40' bays should not exceed n/c.

2. *The containers with the same POD in two adjacent 40' bays should not exceed the average discharging moves per crane at POD.* This constraint is used to balance the unloading workload of containers among the cranes for future ports. For example, if the ship has k containers for unloading at port p with c cranes, then the average discharge moves per crane at port p is k/c. Thus the number of containers with $POD = p$ in any two adjacent 40' bays should not exceed k/c at all ports before p.

In stowage planning, the above rules may lead to a situation that many blocks are partially used up to stow the containers with same POD evenly. This will very likely cause unnecessary rehandles. To adjust the balance between cran split and rehandles, we introduce an imbalance tolerance factor k to relax the above rules. For example, in rule 1, "the total move count of two adjacent 40' bays should not exceed n/c" will be relaxed to "the total move count of two adjacent 40' bays should not exceed $(1 + k)n/c$". In Section 5, we will show the effect of the tolerance factor on stowage planning.

5. CASE STUDY

In our testing, we consider a containership with capacity of 5000 TEUs. The voyage of the containership is given as H-A-B-C-D-E-F-G-H. At every port, a number of containers are unloaded and loaded (see Table 1).

Table 1: Workload at each port

Port	A	B	C	D	E	F	G	H
Unload	904	236	57	770	1605	459	1266	254
Load	1146	1212	339	422	1194	651	6	399

To quantify the tradeoff between crane split and rehandle cost, we generated stowage plan using different tolerance level in the imbalance in crane workload. Tables $2 - 5$ show the statistics of the stowage plans generated for the set of ports for different tolerance level. The stowage plans generated are feasible with respect to the stowage planning constraints. The plans generated by our system are labelled as Plan A, whereas the plans produced by human planner are labelled as Plan B. In Tables $2 - 5$, t_{max} denotes the longest crane working time in each stowage plan. Each set of 8 stowage plans at different tolerance levels are generated within one minutes on an Intel Core2 PC with 2.66 Ghz CPU and 2 GB of RAM. As can be seen from Tables $2 - 6$, the crane split becomes worse and the total ship berthing time in the entire voyage (t_{total} in Table 6) becomes longer with the increase of tolerance factor, while the number of rehandles is reduced (actually no rehandle is needed when the tolerance factor k is greater or equal to 0.10 in this test case).

Given the cost of each rehandle and port stay per hour at each port, we can calculate the exact handling cost of a stowage plan. Consequently, we can generate a set of stowage plans using different tolerance level and select the minimum cost stowage plan. This will be indeed valuable for shipping company in practice.

6. CONCLUSION

In this paper, we present a stowage plan generator based on a "block stowage" heuristic approach to generate stowage plans for large containerships automatically.

Table 2: Stowage plans with $k = 0$

Port	A	B	C	D	E	F	G	H
Crane Numbers	5	4	3	4	4	4	4	4
t_{max}(Plan A)	875	804	354	671	1474	714	701	406
t_{max}(Plan B)	895	948	358	894	1480	703	751	732
CI(Plan A)	4.69	3.60	2.24	3.55	3.81	3.55	3.63	3.21
CI(Plan B)	4.58	3.06	2.25	2.70	3.78	3.24	3.39	1.78
Rehandles(Plan A)	0	0	0	0	29	43	0	0
Rehandles(Plan B)	0	1	3	8	0	14	0	0

Table 3: Stowage plans with $k = 0.05$

Port	A	B	C	D	E	F	G	H
Crane Numbers	5	4	3	4	4	4	4	4
t_{max}(Plan A)	875	804	374	671	1479	705	706	406
t_{max}(Plan B)	895	948	358	894	1480	703	751	732
CI(Plan A)	4.69	3.60	2.12	3.55	3.79	3.63	3.60	3.21
CI(Plan B)	4.58	3.06	2.25	2.70	3.78	3.24	3.39	1.78
Rehandles(Plan A)	0	0	0	0	15	74	0	0
Rehandles(Plan B)	0	1	3	8	0	14	0	0

Table 4: Stowage plans with $k = 0.10$

Port	A	B	C	D	E	F	G	H
Crane Numbers	5	4	3	4	4	4	4	4
t_{max}(Plan A)	875	794	389	701	1504	688	711	421
t_{max}(Plan B)	895	948	358	894	1480	703	751	732
CI(Plan A)	4.69	3.65	2.04	3.40	3.72	3.23	3.58	3.10
CI(Plan B)	4.58	3.06	2.25	2.70	3.78	3.24	3.39	1.78
Rehandles(Plan A)	0	0	0	0	0	0	0	0
Rehandles(Plan B)	0	1	3	8	0	14	0	0

Table 5: Stowage plans with $k = 0.15$

Port	A	B	C	D	E	F	G	H
Crane Numbers	5	4	3	4	4	4	4	4
t_{max}(Plan A)	870	794	429	676	1636	686	706	391
t_{max}(Plan B)	895	948	358	894	1480	703	751	732
CI(Plan A)	4.71	3.65	1.85	3.53	3.42	3.24	3.60	3.34
CI(Plan B)	4.58	3.06	2.25	2.70	3.78	3.24	3.39	1.78
Rehandles(Plan A)	0	0	0	0	0	0	0	0
Rehandles(Plan B)	0	1	3	8	0	14	0	0

Table 6: Total berthing time and rehandles

	Plan A				Plan B
k	0	0.05	0.10	0.15	
t_{total}	5999	6020	6083	6188	6761
r_{total}	72	89	0	0	26

Aiming to emulate the plans made by human planners, the stowage plan generator exhibits very good performance in terms of handling cost and computational efficiency. Moreover, we show the tradeoff between crane split and rehandle, which is useful to reduce the cost of the stowage plan.

The stowage plan generator is only the first part of the stowage planning system. Our next step is to develop the stability adjustment module and optimization engine to generate optimized stowage plans.

ACKNOWLEDGEMENTS

The study is supported by grants from the Maritime and Port Authority of Singapore and APL.

REFERENCES

Ambrosino, D. and Sciomachen, A., 1998. A constraints satisfaction approach for master bay plans, In: Sciutto, G., Brebbia, C.A. (Eds.), *Maritime Engineering and Ports*. WIT Press, Boston, pp. 155-164.

Ambrosino, D., Sciomachen, A. and Tanfani, E., 2004. Stowing a containership: the master bay plan problem, *Transportation Research*, 38, 81-99.

Ambrosino, D., Sciomachen, A. and Tanfani, E., 2006. A decomposition heuristics for the container ship stowage problem, *Journal of Heuristics*, 12, 211-233.

Aslidis, T., 1989. Combinatorial algorithms for stacking problems, Ph.D. Thesis, MIT.

Aslidis, T., 1990. Minimizing of overstowage in container ship operations, *Operational Research*, 90, 457-471.

Avriel, M., Penn, M., 1993. Exact and approximate solutions of the container ship stowage problem, *Computers and Industrial Engineering* 25, 271-274.

Avriel, M., Penn, M., Shpirer, N. and Witteboon, S., 1998. Stowage planning for container ships to reduce the number of shifts, *Annals of Operation Research*, 76, 55-71.

Avriel, M., Penn, M., Shpirer, N., 2000. Container ship stowage problem: complexity and connection capabilities, *Discrete Applied Mathematics*, 103, 271-279.

Dubrovsky, O., Levitin, G. and Penn, M., 2002. A genetic algorithm with compact solution encoding for the container ship stowage problem, *Journal of Heuristics*, 8, 585-599.

Imai, A., Miki, T., 1989. A heuristic algorithm with expected utility for an optimal sequence of loading containers into a containerized ship. *Journal of Japan Institute of Navigation*, 80, 117-124.

Wilson, I.D. and Roach, P.A., 1999. Principles of combinatorial optimization applied to container-ship stowage planning, *Journal of Heuristics*,5, 403-418.

Wilson, I.D. and Roach, P.A., 2000. Container stowage planning: a methodology for generating computerised solutions, *Journal of Operational Research Society*, 51, 1248-1255.

Wilson, I.D., Roach, P.A. and Ware, J.A., 2001. Container stowage pre-planning: using search to generate solutions, a case study, *Knowledge-Based Systems*, 14, 3-4, 137-145.

AUTHORS BIOGRAPHY

XIANTAO XIAO is a project officer at the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include stowage planning optimization and mathematical modelling.

MALCOLM LOW is an Assistant Professor at the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include parallel computing, modeling and simulation, planning and scheduling optimization.

FAN LIU is a project officer at the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include automated stowage planning and combinatorial optimization.
SHELL YING HUANG is an Associate Professor at the School of Computer Engineering, Nanyang Technological University, Singapore. Her research interests are in intelligent decision support systems, intelligent agents and agent-based simulations.
WEN JING HSU is an Associate Professor at the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include parallel and distributed processing and provable efficient algorithms.
ZHENGPING LI is an Associate Research Scientist at the Singapore Institute of Manufacturing Technology (SIMTech). His research interests include supplychain management and logistics optimization.