A local search extended placement heuristic for stowing under deck bays of container vessels

Pacino, Dario; Jensen, Rune Møller

Publication date:
2009

Document Version
Peer reviewed version

A Local Search Extended Heuristic for Stowing Under Deck Locations of Container Vessels

Dario Pacino and **Rune Møller Jensen**

IT University of Copenhagen, Rued Langgaards Vej 7, 2300 Copenhagen S, Denmark

fax.: +45 7218 5001, e.mail: dpacino@itu.dk

Abstract

This paper introduces a new algorithm for stowing containers in bays under deck that exploits the under-constrained nature of this sub-problem in hierarchical stowage algorithms.

1 Introduction

Scaleable vessel stowage planning algorithms (e.g., [4], [2], and [1]) embed a hierarchical decomposition of stowage planning that resembles the work process of human stowage coordinators. Essentially these algorithms first generate a master layout that distributes containers over bays and then solves a placement problem for each of these bays that assigns a position to each container to store in the bay. An interesting characteristic of these hierarchical methods is that, while the overall computational complexity of stowage planning is *NP*-complete, the low-level bay placement problems tend to be under-constrained. The reason for this is that a good master layout distributes the containers such that they are easy to place in the bays (e.g., by limiting the number of different discharge ports represented in the bay to reduce the risk of overstowage and by considering the bay's capacity of each container type as well as its weight and volume limits). The hypothesis of our work is that these under-constrained bay placement problems can be solved efficiently using a placement heuristic combined with local search. Our work is based on real stowage data from an automated stowage system used by a larger liner shipping company. Using the constraints and objectives of the company to store containers in an under deck storage location, we have implemented a three phase local search algorithm to assign containers to slots. The algorithm uses a novel placement heuristic to identify a good initial configuration, which is then brought to optimality in two local search phases: a feasibility phase and an optimality phase. The computational results support our hypothesis. For real stowage jobs of the industrial system, we find optimal or near optimal solutions in a few seconds. The only previous work using local

search to place containers in bays is Wilson and Roach (e.g., [4]). Their algorithm, however, does not use a placement heuristic. It is also unclear whether they include all relevant objectives of the problem and finally, they do not compare with an exact method.

2 Model, Approach, and Results

The model is based on the variable x_{st}^{40} denoting the 40-foot container (if any) placed in stack s and tier t and $x_{st}^{20\alpha}$ denoting the 20-foot container (if any) placed in stack s, tier t in the aft ($\alpha = A$) or fore ($\alpha = F$) slot. The model is subject to the following constraints: 40-foot containers are not assigned to cells occupied by 20-foot containers, the capacity of cells must be fulfilled, reefer containers must be stored in reefer slots, containers must have physical support from below, no 20-foot containers are allowed to be stored on top of 40-foot containers, all containers must be loaded, all pre-placed containers must have their given position, and the weight and height of the containers in a stack must not exceed its maximum limit. There are four objectives: 100 units cost for each overstowing container, 5 units cost for each non-reefer container in a reefer slot, 20 units cost for each discharge port present in a stack, and 10 units cost for each stack used. We refer the reader to [3] for a detailed description of the model.

The three phases of our approach are shown in Algorithm 1. The initial configuration is given by the placement heuristic (l.1), where a lexicographical order is enforced over the containers to load (reefers $\prec$ discharge port $\prec$ 20-foot container). Containers are then stowed in stacks from bottom-up prioritizing stacks with the same or greater discharge port first, and empty stacks second. Containers that cannot be heuristically placed are sequentially stowed. The idea behind the

Algorithm 1: SolveLocation()

```
1  π = placementHeuristic();                                   /* Heuristic Placement */
2  while ¬satisfy(constraints) do
3      π' ← π;                                                  /* Feasibility phase */
4      select s1 ←most violated slot do
5          select s2 ←most improving slot to swap do
6              π ← swap(s1, s2);
7          if π'=π then perform side move
8  while there is objective improvement do
9      π' ← π;                                                  /* Optimality phase */
10     select s1 ←most objective violating slot do
11         select s2 ←most objective improving slot to swap filter only feasible swaps do
12             π ← swap(s1, s2);
13         if π' = π then π =perform tie-breaking swap on π
14 return π;
```

two local search phases is to reduce the search space for optimality by restricting the neighborhood to only feasible configurations. The local search algorithms share the same neighborhood definition. A local move is defined by swapping the containers in two cells (notice that a cell can hold two 20-foot containers or one 40-foot container). Successively the feasibility phase (l.2-7) starts searching

for an assignment where all constraints are satisfied. Here swaps are selected in two steps, choosing first a cell violating most constraints (l.4), followed by a cell which resulting swap would reduce the constraint violations (l.5). If it is impossible to select an improving swap, a side move mechanism allows non-improving swaps (l.7). The search terminates when all constraints are satisfied (l.2). The optimality phase (l.8-13) searches for an optimal solution within the search space of feasible solutions. The difference between the two phases is the selection of the second cell (l.11), which here is chosen only between those that do not break feasibility and that at the same time improve the objective. However, some objectives often need several side moves to improve. Within those side moves (l.13), a tie-breaking rule gives a way to evaluate two solutions with identical objective value, indicating which one of them is the closest to an improvement. We refer the reader to [3] for a detailed description of the algorithm.

Our industrial instances consider under deck storage areas of 16 to 176 TEUs. Optimal results were obtained with an exact constraint programming method developed by Alberto Delgado Ortegon. The algorithm was implemented in COMET under Linux 2.6.18 running on a Dual-Core AMD Opteron, 2.6 GHz, 8GB RAM machine. The results, presented in table 1, use five restarts of the algorithm as local minima escape strategy and show that the algorithm reaches optimality in most of the cases within a few seconds.

Inst.	TEU	Opt.	LS	Time	Gap
1	16	60	60	1,33	0%
2	40	60	60	1,80	0%
3	70	120	120	1,01	0%
4	72	120	120	1,04	0%
5	72	230	230	0,97	0%
6	78	120	120	0,91	0%
7	86	150	150	2,53	0%
8	88	390	390	2,07	0%
9	90	90	90	0,91	0%
10	90	165	180	1,98	9%
11	90	215	232	2,58	8%
12	90	240	240	1,41	0%
13	90	330	330	4,84	0%
14	108	120	120	2,37	0%
15	108	240	268	3,55	12%
16	116	310	315	2,26	2%
17	148	225	243	4,45	8%
18	156	595	595	13,19	0%
19	176	360	382	6,20	4%

Table 1: Instance number (Inst.), twenty-foot equivalent unit (TEU), optimal value (Opt.), local search objective value, run time in seconds (Time), gap between optimal and local search objective value (Gap).

References

[1] D. Ambrosino, A. Sciomachen, and E. Tanfani. A decomposition heuristics for the container ship stowage problem. *Journal of Heuristics*, 12(3):211–233, 2006.

[2] J.G. Kang and Y.D. Kim. Stowage planning in maritime container transportation. *Journal of the Operational Research Society*, 53(4):415–426, 2002.

[3] Dario Pacino. A comet-based meta heuristic for assigning containers to cells in under deck storage locations. Master's thesis, IT-University of Copenhagen, Denmark, October 2008.

[4] I.D. Wilson and P. Roach. Principles of combinatorial optimisation applied to container-ship stowage planning. *Journal of Heuristics*, 5:403–418, 1999.