Stowage planning and pile problems

Peer Giemsch[1]

Universität Karlsruhe (TH), Fakultät für Wirtschaftswissenschaften,
Institut für Anwendungen des Operations Research,
Kaiserstr. 12, D-76128 Karlsruhe, Germany, peer@giemsch.de

Abstract. Stowage planning occurs frequently in logistics but is generally considered as a problem with one loading point and one unloading point. The stowage pattern can be seen as a generalization of a stack if access to the packed items is possible only from one direction. We will call this a pile of items.
A well-known application of a pile of items are the bay plans of containerships, which consist of many parallel stacks of containers. The generation of stowage plans of such containerships becomes complicated due to the multiple loading und unloading points (here: ports) and the objective to minimize moved items at each point.
In our presentation we want to introduce in the field of stowage planning in terms of an application of discrete mathematics in operations research and present heuristics for some cases in order to stimulate discussion for advanced solution methods.

1 Stowage Planning

Cargo stowage is the way of arranging and allocating items in a transport unit in view of the most economic conveyance ([5]). The stowage planning has to consider one big object B, which can be a transport unit or some storage space. The cargo to be stowed consists of many small objects $O = \{o_1, \dots o_n\}$. In a general model the objects have beside of volumetric and gravimetric characteristics some other properties:

- Given orientations ("this side up")
- Limited stackability
- Different builds
- Securing means and values

Usually there is only one loading point for the small objects, where the stowage takes place and one unloading point for clearing the big object.
Though we want to focus on a new setting: Each small object has a loading point, where it waits to get stowed and an unloading point, where it will be discharged. This can be expressed through $\alpha : O \to \mathbb{N}$ and $\omega : O \to \mathbb{N}$ with $\alpha(o) \leq \omega(o) \quad \forall o \in O$. The $\alpha(O)$ are the loading points or departures and the $\omega(O)$ are the unloading points or destinations. The big object B travels from the first loading point to the last unloading point and at every point loading and unloading of small objects can occur.
Dependant on the dimensions of the objects and the nature of the transport

unit there are limited possibilities of access to the stowed items. This can cause some items *block* the retrieval of others. This yields to our first

Definition 1. Object $o_1 \in O$ *overstows* object $o_2 \in O$ if $\omega(o_1) > \omega(o_2)$ and o_1 blocks o_2.

Clearly the blocking condition has to be specified. Withal broad notions we can now make the central

Definition 2. *Stowage planning* is the problem of packing all given small items at each loading point into a given big object and thereby minimizes the number of items overstowed.

2 Classification as a packing problem

We can classify stowage planning as a packing problem due to [6]:

- The dimensionality can be one, two or three.
- Kind of assortment is of type I, which means to use all objects and pack all items.
- Quantity measurement is discrete.
- Objects are of rectangular shape.

The objective can be set to:

- Packing all items in the object at each loading point and
- avoiding unnecessary temporarily (re-)loadings.

Latter means to minimize number of overstowed items. Obviously there is a trade-off between overstowage and volume utilization.
Overstowage cannot only occur due to a given tour of the big object but also to other causes:

- The availability of the small objects can be limited and therefore they build a sequence with partial order.
- There are operational characteristics of the packing problem like multi-stage environment.
- There are some dynamics of allocation, which means a dynamic stowing of items with reallocation like in a warehouse.

Additionally it is perhaps possible to change the sequence of loading points to obtain fewer overstows. For this aspect see [10].

3 Special cases of the problem

There are some special cases, we will discuss now.

3.1 Special cases in time

If $\alpha(o) = 1, \omega(o) = n \ \forall o \in O$, then we have a classical container loading problem in the respective dimension and there are no overstowed items.
If $\omega(o) = n \ \forall o \in O$ and no restriction on α emerges then the stowage planning can be seen as if the items have an availability sequence or the transport unit collects the items along the loading points. There are no overstowed items but possible rearrangement costs.
If $\alpha(o) = 1 \ \forall o \in O$ then all items are loaded at the same point and dropped along the tour in their destinations. There are no overstowed items if the stowage pattern at the loading point consists of guillotine cuts because then we can sort appropriately.
If the small objects are fragmented in disjuct sets like $O = O_1 \dot{\cup} O_2 \dot{\cup} \ldots \dot{\cup} O_k$ with no common loading and unloading points $\omega(O_i) < \alpha(O_j) \ \forall i < j$ and further $\alpha(o) = k_j, \ \omega(o) = l_j \ \forall o \in O_j$ then we have a sequence of classical container loading problems and there are no overstowed items.

3.2 Special cases in space

Since overstowage can only occur if objects block each other, we can look for special cases in space.
If the transport unit is of a shape, that no two items can block each other, then no overstowage occurs. This can be due the fact, that there is one access direction and the pile depth of the items is one or the pile depth is greater than one but there are more than one access directions (see for instance Fig.1).

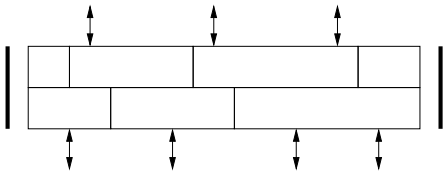

Fig. 1. Example of pile depth two

Another interesting case occurs when all items have a fixed orientation, their width is more than half of the width of the transportation unit and there is one access point (see Fig.2). This case is mentioned in [9] and solved in [2]. We will call this the one stack overstowage problem (OSOP).

In the multi-stack case all items are of the same size, with fixed orientation and can only be arranged in stacks like in the OSOP. There is one access direction for every stack (see Fig.3). The corresponding decision problem is $\mathcal{NP}$-complete due to [3] and its economic relevance is due to the fact that it is a simple version of the containership stowage problem ([7]).

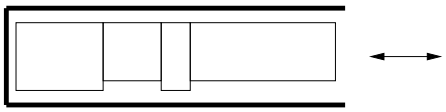

Fig. 2. One stack case

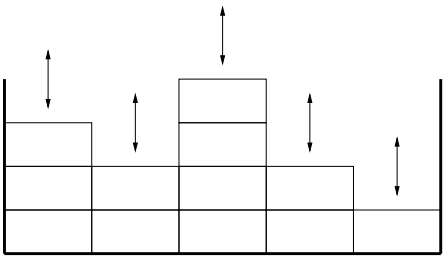

Fig. 3. multi-stack case

4 Piles

To handle the stowage planning problem, we have to find a characterisation of the stowage pattern that allows us to identify blocked items. We restrict to the cases of orthognal items of three dimensions and one specified access direction. Obviously two items are blocked if they touch each other and accordingly have a rectangle perpendicularly to the access direction in common (cf.[11]). But an item can block the retrieval of an other item also if they don‘t share any area. This leads to a characterisation of stowage pattern as graphs.

Definition 3 ([8]). A graph $G = (V, A)$ is called a *pile* if it is finite, directed, without loops and cycles.

We identify a stowage pattern with a pile. Thereby an item is represented through an node of the graph and there is an edge between two nodes if the corresponding two items touch each other vertically to the access direction. A item o is then called *blocked* if if there is an ingoing edge at node o. (See Fig. 4 for an example). In the case of the OSOP, the pile is a simple path.

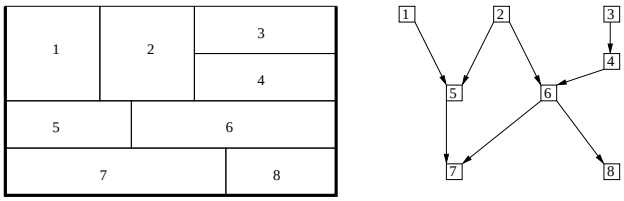

Fig. 4. Left: stowage pattern, right: pile

And in the multi-stack case the pile consists of many components, which are simple paths.

5 Solution Approach

5.1 General 3D case

A heuristic solution approach for packing (3D-)boxes in a single container with one loading point and some unloading points is mentioned in [4]:
Start from the final destination and the corresponding unloading point $\max\{\omega(o)\}$ and pack the shipment $O_{\max\omega}$ with respect of maximizing the space remaining for the rest. This is done by trying to pack walls while preserve container dimension in access-direction. Then iterate from last to nearest destination.
The crucial part is the wall-building method. If there is enough space in access direction, then this heuristics doesn't produce any overstowed items. However if some items of different destinations have to be intermixed in a wall this cannot be assured any more.
Another heuristics is to build towers of the items of each destination and try to place the towers descending on destination distances in the container.
In general, every algorithm for container loading can be applied but has to be shifted to stow items with further destination in the rear part of the container.

5.2 Multi-stack case

In the multi-stack case with all items of equal size but multiple loading and unloading points the solution method of the OSOP mentioned in [2] can be sucessfully applied. The optimal solution of OSOP gives us a policy for each loading point i: We have to decide if we need to rearrange the stack at point i to bring the stack in order (ascending destinations) for the next k destinations. This depends on the items to be stowed at this point and further points.
So if we assume now for the multi-stack case that each item assigned to a stack is loyal to this stack during its journey, then we can calculate after assigning all items to stacks independently for each stack via the OSOP method an optimal policy. If the items are not loyal we can calculate the best stack having some capacity left for some temporarily unloaded items with the OSOP method again.
The OSOP method is crucial, so we will display it here: Let the departures be numbered $0, 1, \ldots, m$ and the destinations numbered $1, 2 \ldots, m+1$. If $V_{(i,j)} = |\{o \in O | \alpha(o) = i, \omega(o) = j\}|$ then we can calculate the number $S(0, m+1)$ of overstowing items in an optimal stowage plan for the OSOP recursively (see [1]):

$$S(i,j) = \min_{i+1 \leq k \leq j-1} \{S(i,k) + r(i,k,j) + S(k,j)\} \quad i = 0, 1, \ldots, m+1 \quad (1)$$

with

$$S(i,i+1) = S(i,i+2) = 0 \qquad i = 0,1,\ldots,m+1 \tag{2}$$

and if $V_{i,j} \neq 0 \; \forall (i,j)$, then

$$r(i,k,j) := \sum_{q=i+1}^{k-1} \sum_{l=k+1}^{m+1} V_{(q,l)} + \sum_{p=0}^{i} \sum_{n=k+1}^{j-1} V_{(p,n)} \tag{3}$$

If at some point only loading and no unloading occurs we have to adjust the calculation of $r(i,k,j)$ but this is omitted here. The optimal rearrangement policy is gained in the recursion.
We give a upper bound for the number of overstowing items, which is given by the following binary linear program. This is done to give an estimation for the performance of other possible heuristics for the multi-stack case Let $|O| = n$ and given stacks with numbers $k = 1,2,\ldots,K$ and capacity h_k.

$$\min \sum_{\substack{i,j \in O \\ \alpha(i)<\alpha(j)<\omega(i)<\omega(j)}} y_{ij} \tag{4}$$

subject to

$$\sum_{k=1}^{K} x_{ik} = 1 \qquad \forall i \in O \tag{5}$$

$$\sum_{i=1}^{n} \sum_{k=1}^{K} x_{ik} = n \tag{6}$$

$$\sum_{i \in \{o \in O | l \in [\alpha(i),\omega(i)]\}} x_{ik} \leq h_k \qquad \forall k, l = 1,\ldots,\max\{\omega(o)\} \tag{7}$$

$$x_{ik} + x_{jk} \leq y_{ij} + 1 \qquad \forall i,j,k \tag{8}$$

$$x_{ik} \in \{0,1\} \qquad y_{ij} \in \{0,1\} \tag{9}$$

(4) counts the number of items in the same stack which cause overstowage. (5) and (6) provide that all items are loaded. (7) provides that at each loading point no more item than the allowed stacking capacity is assingend and (8) connect the variables x and y. The variables meaning is:

$$x_{ik} = \begin{cases} 1 & \text{item } i \text{ assigned to stack } k \\ 0 & \text{otherwise} \end{cases}$$

and $y_{ij} = 1$ indicates, that item i and item j are assigned to the same stack, $y_{ij} = 0$ otherwise.

6 Further Research

The research of stowage planning in the context given above is at the very beginning and the most important questions are (1) how to classify the stowage planning problems more accurate, (2) how to modify known packing algorithmus to solve stowage planning, (3) can we get more information out of the possible piles of a given instance of small items and (4) how well do the proposed heuristics perform?

References

1. Aslidis, A.H. (1989) Combinatorial Algorithms for Stacking Problems. PhD thesis, Massachusetts Institute of Technology
2. Aslidis, A.H. (1990) Minimizing of overstowage in container ship operation. Operational Research 90, p.457-471
3. Avriel, M., Penn, M. and Shpirer, N. (2000) Container ship stowage problem complexity and connection to the coloring of circle graph. Discrete Applied Mathematics 103, p.271-279
4. Bischoff, E.E. and Ratcliff, M.S.W. (1995) Issues in the Development of Approaches to Container Loading. Omega 23, p.377-390
5. Branch, A.E. (1996) Elements of Shipping. Cheltenham: Nelson Thornes, 7.ed.
6. Dyckhoff, H. and Finke, U. (1992) Cutting and Packing in Produktion and Distribution. Heidelberg: Physica Verlag
7. Giemsch, P. and Jellinghaus, A. (2004) Optimization Models for the Containership Stowage Problem. appeared in: Operations Research Proceedings 2003 - Selected Papers of the International Conference on Operations Research (OR 2003), 2.-5.Sept. 2003, Heidelberg, Editors: D.Ahr; R. Fahrion, M. Oswald, G. Reinelt; Berlin: Springer
8. Kämmerer, L. (1997) Mathematische Modellierung und Behandlung von Stapelproblemen. Phdthesis, Bauhaus-Universität Weimar
9. Ladany, S.P. and Mehrez, A. (1984) Optimal routing of a single vehicle with loading and unloading constraints. Transportation Planning and Technology 8, p.301-306
10. Türkay, A. and Emel, E. (2003) Vehicle Routing Problem with Packing Constraints. Presentation on the 5th Euro/Informs Joint International Meeting, July 06-10, Istanbul
11. Wottawa, M. (1996) Struktur und algorithmische Behandlung von praxisorientierten dreidimensionlalen Packungsproblemen, PhD thesis, Mathematisch-Naturwissenschaftliche Fakultät der Universität Köln