Applying Tabu Search to Container Loading Problems

A. Bortfeldt and H. Gehring, FernUniversität Hagen

Abstract: This paper presents a Tabu Search Algorithm (TSA) for container loading problems with a container to be loaded. The TSA is based on a two-stage procedural approach. While alternative stowing plans are generated directly using an integrated greedy heuristic, the execution of the heuristic is controlled by means of a superior Tabu Search. The TSA considers several restrictions of practical relevance. The performance of the TSA, which is tailored to problems with weakly heterogeneous box sets, is demonstrated using numerous reference problems from the literature.

1 Introduction and formulation of the problem

The efficient use of standardized means of transport such as containers and pallets for general freight transport in production and distribution can be effectively supported by the use of computer-based solutions to container loading problems. The container loading problem considered here can be formulated as follows:

Let a container be given and a set of rectangular goods, referred to below as boxes. Determine a feasible arrangement of a subset of all boxes in the container in such a way that the stowed box volume is maximized and any additionally required constraints for loading the container are met.

A box arrangement is called feasible if all stowed boxes are placed in the container completely and without overlapping and lie parallel to the container border areas. From the great number of restrictions on container loading found in practice, the following four constraints are included in the problem:

(C1) Stability constraint
In the interests of high load stability it is required that each stowed box is completely supported by the container floor or the tops of other boxes.

(C2) Orientation constraint
In so far as the storage of goods in a box requires this, one or two side measurements of the box are excluded from use as the height measurement.

(C3) Stacking constraint
Certain boxes must not have any other boxes on top of them, for example, while the goods in the box must not be subjected to a vertical load.

(C4) Weight constraint
The total weight of the load must not exceed the given net weight limit for the container.

Some of the solution approaches developed in recent years for container loading problems with a container are characterized here with regard to their methods. LOH and NEE (1992), NGOI et al. (1994), BISCHOFF et al. (1995) and BISCHOFF and RATCLIFF (1995) all put forward problem-specific heuristics. MORABITO and ARENALES (1994) introduced a procedure for intelligent graph searches with AND/OR graphs, while GEHRING and BORTFELDT (1997) developed a genetic algorithm for container loading problems. Modern heuristic search techniques have up to now rarely been used for three-dimensional packing problems; this applies in particular to the search strategy Tabu Search, originated by GLOVER (1986).

In this paper, a Tabu Search Algorithm (TSA) for the formulated container loading problem will be introduced based on a two-stage procedural approach. Alternative stowing plans for a given problem will be generated directly by an integrated greedy heuristic which is designated as the basis heuristic. The repeated application of the basis heuristic is controlled by a superior Tabu Search. In the following, the basis heuristic will be described first and then the actual

TSA.
Finally, the performance of the procedure will be demonsrated using numerous reference problems from the literature on the subject.

2 The basis heuristic

The basis heuristic loads a given container cyclically, whereby in each cycle a single packing space, i.e. an empty rectangular space in the container, is filled with boxes. To do this, a favourable local arrangement is selected from all suitable box arrangements of a given simple structure, called local arrangements, and stored in a packing list together with the packing space. At the end of the cycle the space not filled by the local arrangement is broken down into additional packing spaces which are processed analogously later on. Fig. 1 shows the individual steps of a cycle in the basis heuristic.

(1) Determine the minimum volume packing space *pscurr* in the set of all packing spaces and remove pscurr from packing space set.

(2) If no non-loaded box can be placed in *pscurr*, go to step (1).

(3) Determine and evaluate all local arrangements for *pscurr* and provide the local arrangements in arrangement list *ArrList* in descending order sorted in accordance with the evaluation.

(4) Increase the packing space index *ip*: = *ip* + 1; carry the packing space *pscurr* and the best local arrangement *ArrList (1)* in position 1 to the packing list.

(5) Update the remaining box set; determine residual packing spaces for *pscurr* and add these to the packing space set.

(6) Go to step (1).

Fig. 1. A cycle of the basis heuristic in imperative notion.

Before the first cycle the packing space set is initialized with the container space as the sole packing space. Only the fillable packing spaces are numbered with the packing space index *ip*, which is set to zero at the beginning. The basis heuristic is terminated when the set of non-processed packing spaces is empty.

Because not only the dimensions but also the coordinates of a reference corner are recorded for each packing space, the position of each stowed box can easily be reconstructed later on. The reference corner is always the bottom left-hand rear corner of a packing space, whereby it is assumed that the container is embedded in the first octant of a 3-D coordinate system, in which one bottom container corner is in the coordinate origin, while the length, width and height of the container point in the direction of the x, y and z axes.

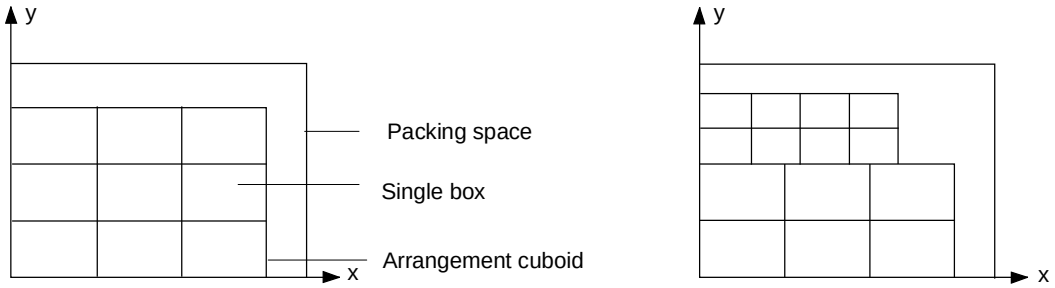
Some details of the basis heuristic are discussed below.

a) Structure of the local arrangements of a packing space

Local arrangements consist of one or two so-called arrangement cuboids and are therefore referred to as 1-arrangements or 2-arrangements (cf. Fig. 2). An arrangement cuboid is formed from boxes which all possess the same type, i.e. their side dimensions conform. In addition, all boxes in an arrangement cuboid are arranged with arbitrarily selected but conforming spatial orientation. An arrangement cuboid always contains *nx* boxes in the x-direction of the packing space, *ny* boxes in the y-direction, and one box in the z-direction.

The only arrangement cuboid of a 1-arrangement is placed in the reference corner of the packing space. The box numbers *nx* and *ny* are selected as high as the horizontal packing space dimensions permit. If the remaining set of boxes contains fewer than *nx*×*ny* boxes of the appropriate box type, both box numbers are reduced alternately.

Of the two arrangement cuboids in a 2-arrangement one is placed in the reference corner and the other is placed next to it in the x- or y-direction. If the other cuboid is placed in the y-direction, the box numbers nx_1 and ny_2 are determined analogously as in the case of the 1-arrangement. Box numbers nx_1 and ny_2 are stipulated so that the sum of the y-dimensions of both arrangement cuboids is maximized. As with the 1-arrangements, the box numbers for the arrangement are reduced in a suitable manner where this is necessary.



a) 1-Arrangement; nx=3, ny=3

b) 2-Arrangement; nx1=3, ny1=2, nx2=4, ny2=2

Fig. 2. 1-Arrangment and 2-Arrangement with neighbouring placement in y-direction (seen from above).

b) Generating residual packing spaces and evaluating local arrangements

With the generation of residual packing spaces for any already filled packing space the total unused space is always divided into separate rectangular spaces. If a 1-arrangement was selected for the packing space, one residual packing space is generated above the arrangement cuboid and two others in front of and beside the arrangement cuboid. In order to generate the latter packing spaces two division variants are used: allocation of an empty cuboid diagonally opposite the arrangement cuboid either to the new packing space in front of or to the new packing space next to the arrangement cuboid. Selection takes place in accordance with two division criteria:

The primary criterion involves the loss volume of the division variant. This results as the total volume of all residual packing spaces in the division in which none of the boxes in the current box set can be placed. The loss volume should be as low as possible.

The secondary, or inferior, criterion involves the largest residual volume in the division. This results as the volume of the residual packing space in the division variant with the greatest volume. The greatest residual packing space volume should be as large as possible.

In the case of packing spaces filled with a 2-arrangement, five residual packing spaces are generated in a similar manner, whereby the best variant of a maximum of 10 division variants is selected using the two criteria for division.

A minimum and a maximum variant are offered for the evaluation of experimentally generated local arrangements. The minimum variant for the evaluation uses as the single quality criterion the covered floor area of a packing space, which should be as large as possible. The maximum variant for the evaluation uses in addition the two division criteria explained above, with the help of which in each case the best packing space divisions of various local arrangements are compared. This comparison requires the provisional generation of residual packing spaces for each of the generated local arrangements of a packing space. All three quality criteria for local arrangements are used equally, whereby a simple evaluation points system is used.

3 The Tabu Search Algorithm

Fig. 3 shows a general Tabu Search Algorithm which will be adapted below to the given problem. *N(s)* designates the neighbourhood of a solution *s*.

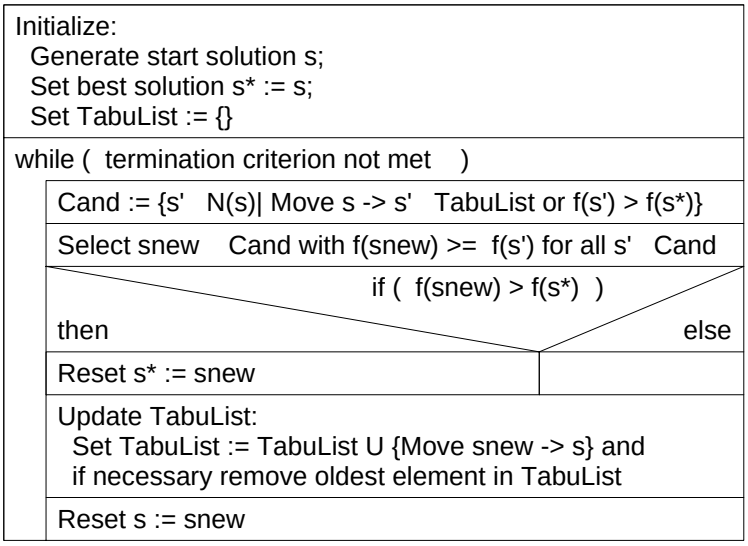


Fig. 3. A general Tabu Search Algorithm.

GLOVER and LAGUNA (1993) should be referred to with regard to the foundations of the search technique Tabu Search. Two generic characteristics of the approach shown in Fig. 3 are underlined here. The first characteristic concerns the structure of the tabu list. If on the transfer from a solution s to the follow-up solution s_{new} the move or value modification $s \rightarrow s_{new}$ is carried out, the respective inverse move $s_{new} \rightarrow s$ is recorded in the tabu list and prohibited for the following n iterations, where n indicates the length of the tabu list. A second characteristic of the approach used is that the following aspiration criterion is used: a move found in the tabu list can still be carried out if it leads to a new global best solution s^*.

In order to be able to introduce suitable neighbourhood structures and to integrate the basis heuristic in the Tabu Search process, the following code is agreed for feasible problem solutions.

A feasible problem solution is coded by a vector P, referred to as packing plan, with *maxps* elements and by a number of significant positions or fillable packing spaces *nps*. A vector element $P(ip)$, $ip = 1,..., nps$, covers two particulars:

- the index $P(ip).ia$ of the local arrangement planned to occupy the ip-th fillable packing space, and
- the number of all local arrangements $P(ip).narr$ of the ip-th packing space.

The transformation of a packing plan P into a feasible problem solution is carried out as follows:

- At the beginning of a transformation a packing plan P includes for each position ip only the index $P(ip).ia$ of the planned arrangement for the ip-th fillable packing space, while the number of local arrangements $P(ip).narr$ and the number of significant packing plan positions *nps* are still undefined.
- Transformation is carried out by means of the basis heuristic which is modified in the following way. For the ip-th fillable packing space the arrangement is used that is found in the previously generated arrangement list at position $P(ip).ia$ instead of the best local arrangement (cf. Fig. 1).
- At the same time the number of generated local arrangements is entered in the packing plan under $P(ip).narr$ and the number of fillable packing spaces or significant packing plan positions *nps* is increased. This results finally in both a feasible problem solution and a complete packing plan P.

The introduced representation of feasible problem solutions forms the basis for the following organization of the general Tabu Search Algorithm in Fig. 3.

Packing plan P_{start} is used as the coded start solution for the tabu search and arrangement index 1 is entered in the plan at all *maxps* positions. The transformed start solution corresponds obviously to the problem solution that is generated with the unmodified basis heuristic (cf. Fig. 1).

Two alternately usable neighbourhood structures are introduced in the space of the coded solutions. For any packing plan P with *nps* significant positions a large neighbourhood *Nl* and a small neighbourhood *Ns* are defined as follows:

- The large neighbourhood *Nl(P)* covers all packing plans P_N in which the following applies: while the entered arrangement indices of packing plan P and a neighbour P_N differ at a single arbitrarily selected alteration position ip^*, they correspond for all other packing plan positions, i.e. $P_N(ip^*).ia \neq P(ip^*).ia$ and $P_N(ip).ia = P(ip).ia$ for all ip with $1 \leq ip \leq maxps$, $ip \neq ip^*$. In addition it is required that the deviating arrangement index $P_N(ip^*).ia$ is in interval [1,*P(ip*).narr*] and the alteration position ip^* is in the significant area [1,*nps(P)*] of packing plan P.
- The small neighbourhood *Ns(P)* is defined analogously to the large neighbourhood. In deviation however, the alteration position ip^* is fixed within the interval $1 \leq ip^* \leq nps(P)$ for one iteration in each case. The given alteration position is set initially at the value $ip^* = 1$ and
  increased by one in each iteration. If an iteration solution is determined using alteration position ip^* which just covers $nps = ip^*$ significant positions, alteration position $ip^* = 1$ is given again. Alteration position ip^* is therefore varied cyclically.

Both neighbourhood structures correspond to the representation of moves used for the tabu list. A move is indicated by means of an alteration position ip^* and by the arrangement indices $P(ip^*).ia$ and $P_{new}(ip^*).ia$ of the old and new iteration solution respectively at the alteration position. As already mentioned, in each case the inverse move (ip^*, $P_{new}(ip^*).ia$, $P(ip^*).ia$) is recorded in the tabu list.

The configuration of the TSA requires several parameters to be stipulated, including above all the respective evaluation variant for local arrangements (minimum or maximum), the neighbourhood structure used (large or small), the length of the tabu list and the number of iterations. Two additional parameters serve the regulation of the neighbourhood size and this for controlling the trade-off between effort and the quality of the solution.

Finally, a diversification of the search can be carried out through a phase breakdown. Each phase is based on specific parameter values, while the search in each phase always begins with the start solution defined by packing plan P_{start} and with an empty tabu list. The best solution determined through all phases is defined as the problem solution.

A uniform parameter set was used for all the following test calculations. This set provides for four search phases which correspond to the four possible combinations from the two evaluation variants for local arrangements and the two neighbourhood structures. The complete search covers 120 iterations and ends after the final search phase, or alternatively after the use of more than 500 seconds CPU time.

4 Test results

The TSA developed here was implemented in C and tested using the following problem groups:

(1) 15 problems from LOH and NEE
The 15 problems introduced by LOH and NEE (1992) have a single orientation constraint (C2) only. With an average of 161.3 boxes to be stowed per problem, on average there are 22.2 boxes per box type.

(2) 700 problems from BISCHOFF and RATCLIFF

The 700 problems defined by BISCHOFF and RATCLIFF (1995) are subdivided into 7 test cases each with 100 problems an designated "BR1" to "BR7". The problems of a test case correspond with regard to the number of existing box types (cf. Table 2). The average number of boxes per box type was 50.2 for test case BR1, but fell continuously and was only 6.5 for test case BR7. The 700 problems from BISCHOFF and RATCLIFF also have a single orientation constraint (C2) only.

(3) 700 problems from BISCHOFF and RATCLIFF with additional constraints

Further 700 problems were obtained by extending the original BISCHOFF and RATCLIFF problems by two additional constraints. A top placement constraint of type (C3) was introduced by stipulating that the first 50 % of the box types in the original problems must not have boxes placed on top of them. A type (C4) weight constraint is agreed by allocation to each box a weight proportional to its volume and providing for a net weight limits for the container which amounts to 66.67 % of the total weight of all boxes to be loaded. The 700 extended problems are subdivided in the same way as the originals into the 7 test cases "BR1R" to "BR7R".

It should be noted that independently of the problem to be solved the TSA always considers a stability constraint (C1).

As is known, a box set with different box types is designated as heterogeneous. Depending on whether a heterogeneous set of boxes contains on average relatively many or relatively few items per box type the box supply is deemed to be strongly heterogeneous or weakly heterogeneous (cf. GEHRING and BORTFELDT 1997). On this basis the box sets in the problems introduced by LOH and NEE may be regarded as weakly heterogeneous, whereas the character of the box sets in the problems supplied by BISCHOFF and RATCLIFF range over the 7 test cases from weakly heterogeneous to strongly heterogeneous.

All test calculations with the TSA and with the genetic algorithm (GA) from GEHRING and BORTFELDT (1997) were carried out using a Pentium PC with a cycle frequency of 200 MHz. A time limit of 500 s CPU time was imposed on the GA as well.

(1) Results for the problems from LOH and NEE

The results for the 15 problems introduced by LOH and NEE may be seen in the following table (Table 1). For each procedure and problem the number of non-loaded boxes is shown in brackets if this is greater than zero, along with the achieved volume utilization. The packing density given for LOH and NEE'S procedure (1992) overestimates the achieved volume utilization (cf. BISCHOFF and RATCLIFF 1995).

The results shown in Table 1 state that the Tabu Search Algorithm achieves the best results among all compared procedures not only with regard to the average achieved volume utilization but also with regard to the number of best solutions achieved.

(2) Results for the problems from BISCHOFF and RATCLIFF

The test results for the 700 problems from BISCHOFF and RATCLIFF are shown in Table 2. Along with the test case designation the number of box types per problem is indicated. The results in the column "Bischoff and Ratcliff - Combination" arise from an evaluation for each problem of the better of the two heuristic methods shown further to the left.

For all test cases, and determined over all 700 problems, the TSA achieves clearly better volume utilizations than the heuristic methods shown in Cols. 2 - 4. In a comparison with the GA the Tabu Search Algorithm achieves better average volume utilizations for test cases BR1 to BR3, whose problems contain markedly weakly heterogeneous box sets. In the other test cases with more stronger heterogeneous box sets the GA dominated, and achieved a slightly better volume utilization on average as well, and had a much lower average calculation time.

Table 1. Results of tests of LOH und NEE's 15 problems.

Problem	Loh and Nee	Ngoi et al.	Bischoff et al.	Bischoff and Ratcliff	Gehring and Bortfeldt (GA)	Bortfeldt and Gehring (TSA)
	Packing density (in %)	Volume utilization (in %)	Volume utilization (in %)	Volume utilization (in %)	Volume utilization (in %)	Volume utilization (in %)
LN01	78.1	62.5	62.5	62.5	62.5	62.5
LN02	76.8 (32)	80.7 (54)	89.7 (23)	90.0 (35)	89.5 (39)	96.6 (28)
LN03	69.5	53.4	53.4	53.4	53.4	53.4
LN04	59.2	55.0	55.0	55.0	55.0	55.0
LN05	85.8	77.2	77.2	77.2	77.2	77.2
LN06	88.6 (45)	88.7 (48)	89.5 (24)	83.1 (77)	91.1 (32)	91.2 (49)
LN07	78.2 (21)	81.8 (10)	83.9 (1)	78.7 (18)	83.3 (7)	84.7
LN08	67.6	59.4	59.4	59.4	59.4	59.4
LN09	84.2	61.9	61.9	61.9	61.9	61.9
LN10	70.1	67.3	67.3	67.3	67.3	67.3
LN11	63.8	62.2	62.2	62.2	62.2	62.2
LN12	79.3	78.5	76.5 (3)	78.5	78.5	78.5
LN13	77.0 (15)	84.1 (2)	82.3 (5)	78.1 (20)	85.6	84.3 (4)
LN14	69.1	62.8	62.8	62.8	62.8	62.8
LN15	65.6	59.6	59.5	59.5	59.5	59.5
Avg 1 - 15	**74.2**	**69.0**	**69.5**	**68.6**	**69.9**	**70.4**

Table 2. Results of tests of BISCHOFF und RATCLIFF 's 700 problems.

Test case	Bischoff et al.	Bischoff and Ratcliff	Bischoff and Ratcliff - combination	Gehring and Bortfeldt (GA)	Bortfeldt and Gehring (TSA)
	Volume utilization (in %)	Volume utilization (in %)	Volume utilization (in %)	Volume utilization (in %)	Volume utilization (in %)
BR1 (3)	81.76	83.79	85.40	85.80	89.04
BR2 (5)	81.70	84.44	86.25	87.26	88.73
BR3 (8)	82.98	83.94	85.86	88.10	88.22
BR4 (10)	82.60	83.71	85.08	88.04	87.38
BR5 (12)	82.76	83.80	85.21	87.86	86.68
BR6 (15)	81.50	82.44	83.84	87.85	85.50
BR7 (20)	80.51	82.01	82.95	87.68	83.92
Avg BR1-BR7	**81.97**	**83.45**	**85.07**	**87.51**	**87.07**

(3) Results for the problems from BISCHOFF and RATCLIFF with additional constraints

The results from the tests of the 700 problems from BISCHOFF and RATCLIFF with additional constraints may be seen in Table 3. With the more constrained problems in test cases BR1R to BR7R the TSA achieved a better average volume utilization in 6 of 7 test cases and also averaged over all 700 problems. The GA dominated only in test case BR7R with the lowest box number per type.

Table 3. Results of tests of BISCHOFF und RATCLIFF 's 700 extended problems.

Test case	Gehring and Bortfeldt (GA)	Bortfeldt and Gehring (TSA)
	Volume utilization (in %)	Volume utilization (in %)
BR1R (3)	64.17	65.57
BR2R (5)	64.26	65.69
BR3R (8)	62.81	65.42
BR4R (10)	63.31	65.09
BR5R (12)	63.27	64.82
BR6R (15)	64.97	65.34
BR7R (20)	64.68	64.44
Avg BR1R-BR7R	**63.92**	**65.20**

The test results can be summarized by stating that the developed Tabu Search Algorithm achieves higher volume utilizations in particular for problems with weakly heterogeneous box sets and in this regard proves to be superior to several procedures taken from the literature on the subject.

References

BISCHOFF, E. E., JANETZ, F. & RATCLIFF, M. S. W. (1995). Loading pallets with non-identical items. European Journal of Operational Research, Vol. 84, pp. 681-692.

BISCHOFF, E.E. & RATCLIFF, M.S.W.(1995). Issues in the development of approaches to container loading. Omega, Vol. 23, pp. 377-390.

GEHRING, H. & BORTFELDT, A. (1997). A Genetic Algorithm for Solving the Container Loading Problem. Accepted for International Transactions in Operational Research 1997.

GLOVER, F. (1986). Future paths for integer programming and links to artificial intelligence. Computers and Operations Research, 5(1986), pp. 533 - 549.

GLOVER, F. & LAGUNA, M. (1993). Tabu Search. in: REEVES (1993), pp. 70 - 150.

LOH, T. H. & NEE, A. Y. C. (1992). A packing algorithm for hexahedral boxes. Proceedings of the Conference of Industrial Automation, Singapore 1992, pp. 115 - 126.

MORABITO, R. & ARENALES, M. (1994). An AND/OR-graph approach to the container loading problem. International Transactions in Operational Research. Vol. 1, Nr. 1, pp. 59 - 73.

NGOI, B. K. A., TAY, M. L. & CHUA, E. S. (1994). Applying spatial representation techniques to the container packing problem. International Journal of Production Research, Vol. 32, pp. 111 - 123.

REEVES, C. R. (Ed) (1993). Modern heuristic techniques for combinatorial problems. Oxford: Blackwell Scientific Publications.