The two dimensional bin packing problem with side constraints

Markus Seizinger

Angaben zur Veröffentlichung / Publication details:

Seizinger, Markus. 2018. "The two dimensional bin packing problem with side constraints." In *Operations Research Proceedings 2017: Selected Papers of the Annual International Conference of the German Operations Research Society (GOR), Freie Universiät Berlin, Germany, September 6-8, 2017*, edited by Natalia Kliewer, Jan Fabian Ehmke, and Ralf Borndörfer, 45–50. Cham: Springer International. https://doi.org/10.1007/978-3-319-89920-6_7.

The two dimensional bin packing problem with side constraints

Markus Seizinger

University Center of Health Sciences at Klinikum Augsburg (UNIKA-T), Department of Health Care Operations/Health Information Management, Faculty of Business and Economics, University of Augsburg, Universitätsstraße 16, 86159 Augsburg, Germany

Abstract We propose a new variant of Bin Packing Problem, where rectangular items of different types need to be placed on a two-dimensional surface. This new problem type is denoted as two-dimensional Bin Packing with side constraints. Each bin may consist of different two-dimensional sides, and items of different types may not overlap on different sides of the same bin. By different parameter settings, our model may be reduced to either a two- or three-dimensional Bin Packing Problem. We propose practical applications of this problem in production and logistics. We further introduce lower bounds, and heuristics for upper bounds. We can demonstrate for a variety of instance classes proposed in literature that the GAP between those bounds is rather low. Additionally, we introduce a Column-Generation based algorithm that is able to further improve the lower bounds and comes up with good solutions. For a total of 400 instances, extended from previous literature, the final relative gap was just 6.8%.

1 Introduction

In the *two dimensional bin packing problem with side constraints* (2DBP-SC) we are given a set of rectangular items $i \in I$, each defined by its height h_i, width w_i and type t_i. A bin consists of S sides with dimensions H and W, respectively. The goal of 2DBP-SC is to assign every item a concrete position such that all items are packed without overlapping and using as few bins as possible. Furthermore, items are packed orthogonally and the newly introduced *side constraint* has to be satisfied, meaning that no two items of different type may be placed face-to-face on the same bin but on different sides. We assume items may not be rotated. This problem is an extension of the well-studied *two dimensional bin packing problem* (2DBP), where only one bin side exists ($S = 1$). 2DBP-SC is strongly NP-hard, since 2DBP is known to be such (Martello & Vigo 1998; Martello et al. 2000).

The example depicted in Fig. 1 gives an idea of the restrictions implied by the side constraint. This small instance contains bins with two sides and two item types. The side constraint reduces the available space on different sides of the bin. In contrast to the traditional 2DBP, where items simply occupy bin capacity for their respective shape, in 2DBP-SC items additionally block this capacity on all other sides of the bin. Only items of identical type may use this blocked area (indicated by hatched area). This means that the concrete capacity consumption of a single item depends on the actual packing of the bin. Notice that another item of type A (green) with dimensions $(w, h) = (4, 6)$ could be placed in the bottom right corner on the rear side of the bin, whereas an identically shaped item of type B cannot be placed in this position, because it would overlap with area blocked from the front side for type A.

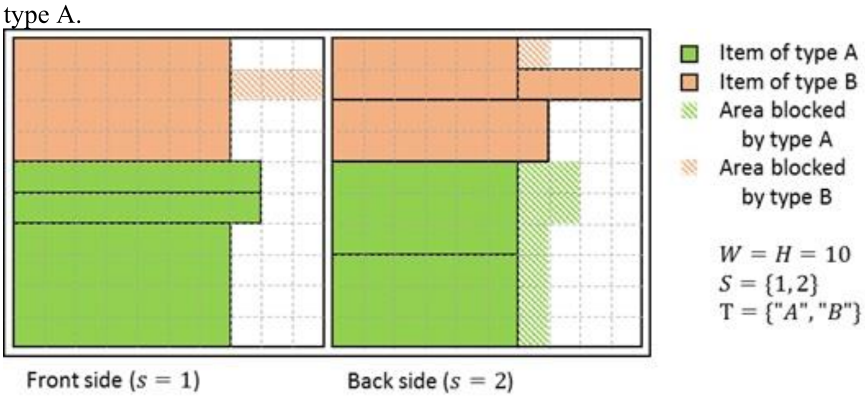

Fig. 1. Example Packing

The motivation for this problem originates from a real-world application. We analyzed a bottleneck resource in a paint shop. Items have to be placed on racks with two faces (front and rear). Additionally, two types of items exist. These types are not allowed to be placed face-to-face for quality reasons. Due to these restrictions the utilization of racks was very low and this production step became a limiting factor. 2DBP-LC corresponds directly to this problem. The same problem must be solved when packing a multi-temperature compartment truck. These trucks consist of flexible departments, dividing them into different temperature zones. The cargo has to be packed into shelves, while every item has to be in its respective temperature zone.

2 Lower Bounds

We introduce already known lower bounds for 2DBP and 3DBP and adapt them to our new class of 2DBP-SC. In general, two ideas exist: The geometric bound sums up volumes of all items and divides it by the area available in a single bin (Berkley & Wang 1987; Martello & Vigo 1998). In contrast, the bound of large items focusses on items fulfilling the conditions $w_i > \frac{W}{2}$ and $h_i > \frac{H}{2}$. We combine both ideas as in Martello et al. (2000) to bound L_2^{SC}. By adapting the bound to 2DBP-SC, we can account for item types. This allows us to further improve existing bounds.

3 Upper Bound

For the 2DBP several solution heuristics exist: Lodi et al. (2002b) describe well-known heuristics Next-Fit Decreasing Height, First-Fit Decreasing Height, and Best-Fit Decreasing Height. The latter dominates the remaining two regarding solution quality (Lodi et al. 2002a).
We adapt the *Best-Fit Decreasing Height* (BFDH) algorithm, first introduced by Berkey & Wang (1987). Our algorithm, called *Best-Fit Decreasing Height with side constraints* (BFDH-SC), creates feasible packings but does not guarantee to find an optimal one. We first order all items according to (a) type and (b) non-increasing height. The first item in the list – the active item – is packed on a level containing only items with identical type and sufficient remaining space. According to the best-fit rule we select that level, where the remaining horizontal space is minimized. If no such level exists, a new level is initialized on top of the latest one, if the bin has enough remaining height. If not, a new bin is initialized as well. Finally, the active item is removed from the list. The algorithm terminates, if the list of items is empty. The heuristic has a runtime complexity of $O(n^3)$.

4 Solution Algorithm

We decompose the problem straightforward according to Dantzig-Wolfe and solve the resulting set covering formulation with column generation. In particular, we form a *(Restricted) Master Problem* (RMP) and a *two dimensional packing problem* (2DPP), acting as the subproblem.
We will solve the RMP in every iteration of the column generation procedure to obtain the dual variables π_i associated with every item. We can then use this information to find a new feasible packing that can improve the current solution of the RMP by solving the subproblem. Thereafter, the next iteration starts over again by solving the slightly enlarged RMP. If all sufficient columns are added to the RMP, its solution is optimal, i.e. the LP-relaxation of RMP has been solved. From theory the bound improves the LP-relaxation of the original problem.

Initial columns are created by BFDH-SC, L_2^{SC} acts as initial lower bound. During the course of the algorithm, new lower bounds are available in every iteration (Lübbecke & Desrosiers 2005), using optimal solution values of RMP and subproblem $L^{CG} = \left\lceil \frac{z_{RMP}}{1-z_{sub}} \right\rceil$.

Subproblem decomposition

In our approach the subproblem has to find a feasible packing pattern p with negative reduced costs $c_p^{\pi} < 0$. If no negative reduced cost column exists, the column generation algorithm terminates with the optimal LP solution of RMP.
To solve this complex subproblem, we follow the approach of Pisinger & Sigurd (2007). There, the pricing problem is decomposed into a *one dimensional knapsack problem* (1DKP) and a relaxed 2DPP. 1DKP selects a subset of items $I' \subseteq I$ prior to the packing problem. We then solve the 2DPP with this reduced set, what leads to an increase in speed for the latter problem. Notice that this comes at the cost of decreasing solution quality, since we cut a quite large region of the solution space. If any item, which is part of the optimal packing in this iteration, is not an element of I', then 2DPP is not able to find this optimal solution, so it will be necessary to solve 2DPP for the original set of items I in some iterations.

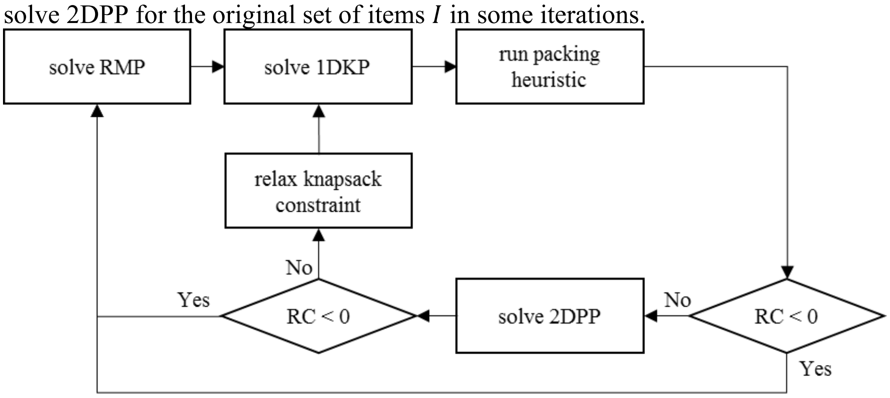

Fig. 2. Iteration scheme of column generation process

First, we extend a MIP formulation for a traditional 2DPP first published in Pisinger & Sigurd (2007) to take item types into account. Decision variables indicate whether a pair of items is part of the packing ($a_i, a_j = 1$) and if so, the relative position of two items (f_{ij}, l_{ij}, b_{ij}). Two non-overlapping items may ether be in front, behind, left, right, above, or below of each other. The side-constraint is implemented as (III). The problem's goal is to create a new pattern with minimal reduced costs.

$$z^{2DPP} = \min 1 - \sum_{i \in I} \pi_i a_i \quad \text{(I)}$$

$s.t.$:

$$f_{ij} + f_{ji} + l_{ij} + l_{ji} + b_{ij} + b_{ji} + (1 - a_i) + (1 - a_j) \geq 1 \quad \forall\, i, j \in I: i < j \quad \text{(II)}$$

$$l_{ij} + l_{ji} + b_{ij} + b_{ji} + (1 - a_i) + (1 - a_j) \geq 1 \quad \forall\, i, j \in I: i < j \wedge t_i \neq t_j \quad \text{(III)}$$

$$x_i - x_j + W l_{ij} \leq W - w_i \quad \forall\, i, j \in I \quad \text{(IV)}$$

$$y_i - y_j + H b_{ij} \leq H - h_i \quad \forall\, i, j \in I \quad \text{(V)}$$

$$s_i - s_j + Sf_{ij} \le S - 1 \quad \forall\, i,j \in I \qquad \text{(VI)}$$
$$x_i \le W - w_i \quad \forall\, i \in I \qquad \text{(VII)}$$
$$y_i \le H - h_i \quad \forall\, i \in I \qquad \text{(VIII)}$$
$$s_i \le D - 1 \quad \forall\, i \in I \qquad \text{(IX)}$$
$$x_i, y_i, s_i \ge 0 \quad \forall\, i \in I \qquad \text{(X)}$$
$$l_{ij}, b_{ij}, f_{ij} \quad binary \quad \forall\, i,j \in I \qquad \text{(XI)}$$
$$a_i \quad binary \quad \forall\, i \in I \qquad \text{(XII)}$$

Second, we additionally introduce a packing heuristic inspired by an algorithm by Martello et al. (2000) to speed up the process of creating a new column. This greedy algorithm starts with an empty bin and aligns items along edges of already placed items until no more items fit into the bin. It takes the side constraint explicitly into account but does not guarantee to find the minimum reduced-cost packing, though it is much faster than solving the subproblem to optimality with the above formulation.

The algorithm works as follows. Items are initially sorted by non-increasing relative value, so $\frac{\pi_i}{w_i h_i}$. The first item is then placed on the bottom left corner of the front side. Now, items are placed iteratively by selecting the best valid position. These positions are generated based on the current packing and the shape of the current item. Valid positions in the sense of this heuristic are all corner points of already placed items. These points are copied to every side of the bin, no matter on which side the actual item is positioned. The points are reduced to those, which allow placing the current item without violating any constraint. We consider the point maximizing the distance between item and borders of the bin the best point. Any other selection-rule can be used as well, but trying to place items of same type behind of each other as soon as possible leads to good results.

We use this heuristic in every subiteration and only if it fails to find a promising packing, will we relax the knapsack constraint and apply an exact algorithm for 2DPP-SC. This combination of the two approaches proved to be very efficient. But the exact method is still computationally expensive, due to the geometrical structure of the problem.

5 Experimental Results

For a numerical study, we use test instances described in Martello & Vigo (1998). Originally, items were member of one of four groups (‘wide’, ‘tall’, ‘large’, small’), defining a range for width and height of the individual item. Groups were extended with subclasses (‘A’, ‘B’) determining types of items. The first subclass assigns types independently of an item’s dimensions. In instances of subclass ‘B’, item types are predetermined by dimensions.

We defined bins to be of size $W = 10, H = 10$ with sides $S = 2$ and different item types $|\mathrm{T}| = 2$. For this setup, the algorithm was able to solve 179 of 400 instances to proven IP optimality, although its goal is to solve the LP-relaxation only. Initial lower bounds were improved in 89 cases, and initial solutions in 77 cases, so a significant number of instances – especially but not only small ones – could be solved to optimality by BFDH-SC. The absolute gap could be reduced to 0.78 (initially 1.2) bins, which corresponds to a relative gap of 6.8% over all instances. Results prove, the used algorithm is able to generate many columns in short time. Its biggest weakness is to meet the termination criterion, so to prove that no new negative reduced-cost column exists.

6 Conclusion

This thesis is the first work to introduce a new class of bin packing problems side constraint. One of its main contributions is to formally describe this new problem class. We proved such problems to be NP-hard and that the usage of a compact MIP-formulation is practically unsolvable. We applied several lower bounds of related bin packing problems and showed in a computational study that these bounds are quite tight. Additionally, we developed a new best-fit algorithm for our new problem to obtain fast and good solutions heuristically.
The development of a column generation procedure and extensive computational experiments on a set of extended, standard benchmark instances is another main contribution of this thesis. First, we decomposed the problem according to Danzig-Wolfe. Due to the enormous complexity, we further decomposed the subproblem process and introduced an additional knapsack problem, as well as a heuristic packing algorithm.

References

Berkey, J. O.; Wang, P. Y. (1987): Two-Dimensional Finite Bin-Packing Algorithms. In *The Journal of the Operational Research Society* 38 (5), p. 423.

Crainic, Teodor Gabriel; Perboli, Guido; Tadei, Roberto (2008): Extreme Point-Based Heuristics for Three-Dimensional Bin Packing. In *INFORMS Journal on Computing* 20 (3), pp. 368–384.

Lodi, Andrea; Martello, Silvano; Monaci, Michele (2002a): Two-dimensional packing problems. A survey. In *European Journal of Operational Research* 141 (2), pp. 241–252.

Lodi, Andrea; Martello, Silvano; Vigo, Daniele (2002b): Recent advances on two-dimensional bin packing problems. In *Discrete Applied Mathematics* 123 (1-3), pp. 379–396.

Lübbecke, Marco E.; Desrosiers, Jacques (2005): Selected Topics in Column Generation. In *Operations Research* 53 (6), pp. 1007–1023.

Martello, Silvano; Pisinger, David; Vigo, Daniele (2000): The Three-Dimensional Bin Packing Problem. In *Operations Research* 48 (2), pp. 256–267.

Martello, Silvano; Vigo, Daniele (1998): Exact solution of the two-dimensional finite bin packing problem. In *Management Science* 44 (3), pp. 388–399.

Pisinger, David; Sigurd, Mikkel (2007): Using Decomposition Techniques and Constraint Programming for Solving the Two-Dimensional Bin-Packing Problem. In *INFORMS Journal on Computing* 19 (1), pp. 36–51.