Variable Sized Bin Packing With Color Constraints

Milind Dawande [a] , Jayant Kalagnanam [b] , Jay Sethuraman [c]

[a] *Univ. of Texas at Dallas, milind@utdallas.edu*

[b] *IBM, T. J. Watson Research Center, jayant@us.ibm.com*

[c] *Columbia Univ., jay@ieor.columbia.edu*

Abstract

In this paper, we introduce the *variable sized bin packing problem* with novel packing constraints, called the *color constraints*. In an instance of this problem, we have n items to be packed into bins of m distinct sizes. Items come with two attributes: *color* and *size*. In addition to the usual capacity constraints, we require that each bin contain items with at most p distinct colors, where p is a pre-specified positive integer. Our objective is to minimize the total capacity of the bins used in the packing. This problem arises as a model for the *slab design problem* in the production planning process of a steel plant. An APTAS and two 3-approximations are presented.

Key words: bin-packing problem; color constraints; approximation algorithms.

1 Our Results

Let $\{w_1, w_2, ..., w_m\}$ be the distinct bin sizes available, $S = \{I_1, I_2, ..., I_n\}$ be the set of items to be packed, and $C = \{c_1, c_2, ..., c_k\}$ be the set of distinct colors. For $i = 1, ..., k$, let $S^i = \{I^i_1, ..., I^i_{n_i}\}$ be the set of items of color c_i sorted by descending item weight. (Clearly, $S = \cup_{i=1}^{k} S^i$.) The weight of an item I_j is $s(I_j)$. Also, the *contents* of a bin B is the total weight of the items packed into B and is denoted by $c(B)$. We study two problems in this paper: (i) $(VBPC)$: Find an optimal packing of all the items into the bins (note that there are different bin sizes available) such that the number of distinct colors in each bin is no more than two and (ii) $(GVBPC)$: Find an optimal packing of all the items into the bins such that the number of distinct colors in each bin is at most a given integer p, where $2 \leq p \leq k$. By an *optimal packing* we mean a packing that minimizes the total capacity of the bins used in the packing.

Note: The case $p = 1$ is not very intersting in studying $(GVBPC)$; if $p = 1$, the problem decomposes and can be solved as k variable sized bin packing problems. For the same reason, we exclude the case $p = 1$ in studying $(GBPC)$.

1.1 An APTAS for GVBPC

Our first result is an asymptotic polynomial time approximation scheme for the GVBPC problem, based on a variant of the ideas of Karmarkar and Karp [2] and is a modest generalization of Murgolo [1]. There are two key ideas in the development of an APTAS: (i) *discretize* the sizes of the items so that there are at most M distinct sizes, and (ii) ignore bin-sizes that are too small. In discretizing the item sizes, we need to round *up* the sizes of the given items so as to reduce the number of different item sizes, while still being able to derive a packing for the given list of items. This rounding idea, coupled with the fact that there are k colors means that every item is on one of $k \cdot M$ types—all items belonging to a particular type have the same color and size.

Our APTAS proceeds along the following lines: we first establish that we can exclude arbitrarily small bin sizes with only a slight loss in the performance; this ensures that we do not use *too many* bins. We then round the sizes of the items to derive a new instance in which the number of *distinct* item sizes is *small*. In the terminology of Karmarkar and Karp [2], we use linea rounding. To start with, we assume that all items with size smaller than a β (L_s) are discarded (β will be chosen later); such items will be packed using a simple post-processing step, without significantly affecting the asymptotic performance guarantee. We then use linear grouping on all the items with size at least β, denoted by L_b, and round up the item sizes within each group. We then formulate a linear programming problem, whose solution will be used to determine an packing for this new instance, in which there are few distinct item sizes. To define the linear program, we need to enumerate all possible configurations—ways in which a bin can be packed in a feasible solution. (In our case, each item has a size and a color attribute.) Since every item we pack has is of size at least β, items could be one of k distinct colors, and each feasible bin can contain at most p different colors. Thus, the total number of configurations is at most N, which is independent of n, but depends exponentially on k, p and M.

Let a_{ij} be the number of items of type i used in configuration j, for $1 \leq i \leq t, 1 \leq j \leq N$. Let x_j be the number of times configuration j is used in a feasible solution to $(GVBPC)$ and let S_j be the size of the bin used in configuration j. Consider the following linear program:

$$\text{LP-GVBPC:} \quad \min \left\{ \sum_{j=1}^{N} S_j x_j : \sum_{j=1}^{N} a_{ij} x_j \geq d_i,\ i = 1, \ldots t; x_j \geq 0,\ j = 1, \ldots, N \right\}$$

The linear programming problem (LP-GVBPC) has N variables and t constraints, and so has an optimal basic feasible solution (which can be obtained using the ellipsoid method [3]) with at most t non-zero entries; rounding up the fractional solution thus obtained will result in the addition of at most t bins. Calling this solution (RLP), we see that $RLP(\hat{L_b}) \leq OPT(\hat{L_b}) + t < (1 + \beta + \frac{1}{\beta n})OPT(L_b) + t$. Our analysis upto this point has ignored the items in the list L_s, those that have size less than β. We assume that the items in L_b have been packed using the algorithms described earlier. The post-processing procedure to pack the small items is a simple strategy, similar to first-fit: Assume the current packing obtained from rounding the optimal linear programming solution uses r bins, labeled $B_1, B_2, \ldots, B_r$. Sort the small items in decreasing order of weight. Pack each item in the first bin to which this item can be assigned. If an item cannot be packed into any nonempty bin (either because of its size or because of its color), we open a new unit-sized bin.

1.2 Two 3-approximations and an online algorithm for VBPC

Procedure $PSB(S^i)$

(1) (Initialization): $i = 1$. $S^i = \{I_1^i, ..., I_{n_i}^i\}$ is the set of items of color c_i sorted by descending item weight.
(2) Try to pack item I_j^i into bins already open. If all items in S^i have been packed, STOP.
(3) If none of the currently open bins can accomodate I_j^i, open a new bin of *smallest* size that can accomodate I_j^i and pack I_j^i in this new bin. If all items in S^i have been packed, STOP.
(4) $i = i + 1$. Return to Step 2.

Algorithm PSB

(1) $i = 1$.
(2) while $i \leq k$
- Pack items of color c_i using procedure $PSB(S^i)$.
- $i = i + 1$; return to Step 2.

Theorem 1.1 *Algorithm PSB is 3-approximation for VBPC and this bound is tight.*

Algorithm $FFDC$, given below, is an adaptation (for color constraints and variable bin-sizes) of the classical first-fit decreasing (FFD) algorithm for the bin-packing problem. Since we have variable sized bins, algorithm $FFDC$ opens the *smallest* bin which can accomodate an item. Also, while packing an item into a bin, the residual capacity as well as the color constraint is checked.

Algorithm $FFDC$

(1) Sort the items in decreasing order of size. Thus, $s(I_1) \geq s(I_2) ... \geq s(I_n)$.
(2) For packing item i, traverse the currently open bins in the order in which they were opened. Pack item i in the first bin which can accomodate it. Note that we have to consider the *residual capacity* of the currently open bins as well as the *number of distinct colors* of the items currently packed into the bin. If no bin can accomodate item i, open a new bin of smallest size which can accomodate item i and assign item i to that bin.

Theorem 1.2 *Algorithm $FFDC$ is 3-approximation for VBPC.*

Algorithm $FFCL$: An online algorithm

(1) For packing item i, traverse the currently open bins in the order in which they were opened. Pack item i in the first bin which can accomodate it. If no bin can accomodate item i, open a new bin of size 1 and assign item i to that bin.
(2) Repeat Step 1 until all the items have been packed.
(3) At the end of Step 2, suppose there are l bins with residual capacity at least $\frac{1}{2}$. If there exists a pair of such bins such that both the bins in the pair have only one color in them, we repack the contents of one bin in the pair into the other bin.

Note that, for the l bins which have residual capacity at least $\frac{1}{2}$, each color can appear in at most one of these bins.

Theorem 1.3 *The total space used by Algorithm $FFCL$ is $\leq 2 * OPT + \lceil \frac{k}{2} \rceil$.*

References

[1] F. D. Murgolo, Efficient approximation scheme for Variable-sized bin packing, *SIAM J. Computing* (1987) 149-161.

[2] N. Karmarkar and R. M. Karp, An efficient approximation scheme for the one-dimensional bin packing problem, *IEEE FOCS* (1982) 312-320.

[3] M. Grotschel, L. Lovasz and A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica* **1** (1981) 169-198.