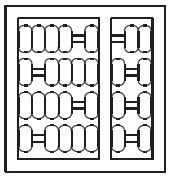

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

An APTAS for the Variable-Sized Bin Packing Problem With Color Constraints

E. C. Xavier *F. K. Miyazawa*

Technical Report - IC-05-15 - Relatório Técnico

July - 2005 - Julho

An APTAS for the Variable-Sized Bin Packing Problem With Color Constraints*

E. C. Xavier† F. K. Miyazawa†

July 5, 2005

Abstract

We consider the Variable-Sized Bin Packing Problem With Color Constraints (VBPCC). In this problem we have an unbounded number of bins, each one with size in $\{w_1, \ldots, w_k\}$, a list of items L, each item $e \in L$ with size s_e and color c_e. The objective is to pack the items of L into bins, such that no bin has items of more than p colors, the total size of items packed in a bin is no more than the bin size and the total size of used bins is minimized. We present an asymptotic polynomial time approximation scheme when the number of different item colors is bounded by a constant C.

1 Introduction

We consider the Variable-Sized Bin Packing Problem with Color Constraints (VBPCC). An instance of the problem VBPCC is a tuple $I = (L, s, c, w, p)$, where L is a list of items, s and c are size and color functions over L, p is the maximum number of different colors that a bin can have and w is a function of bins size. We assume that the number of different item colors in the list L is bounded by a constant C. Clearly we must have $p \leq C$, and if $p = C$ we have the classical bin packing problem. We assume w.l.o.g. that the maximum size of a bin is 1 and the image of w is the set $\{w_1, \ldots, w_k\}$, where $w_i < w_{i+1}$ and $w_k = 1$. Given a set $B \subseteq L$, we denote by $w(B) = \min\{w_i : \sum_{e \in B} s_e \leq w_i\}$ and $s(B) = \sum_{e \in B} s_e$. A packing of L is a partition $\mathcal{P} = \{B_1, \ldots, B_k\}$ of L, where for each $B \in \mathcal{P}$ the number of different colors in B is at most p and $s(B) \leq 1$. We denote by $w(\mathcal{P}) = \sum_{B \in \mathcal{P}} w(B)$ the cost of the packing $\mathcal{P}$ and by $|\mathcal{P}|$ the number of bins used in $\mathcal{P}$. Given an instance $I = (L, s, c, w, p)$ the problem VBPCC is to find a packing $\mathcal{P}$ of L such that $w(\mathcal{P})$ is minimized.

Throughout this paper, we assume that the number of different colors in the input instance is bounded by a constant C and each color belongs to the set $[C] = \{1, \ldots, C\}$.

Given an algorithm $\mathcal{A}$ for the VBPCC problem and an instance I, we denote by $\mathcal{A}(I)$ the cost of the packing generated by the algorithm $\mathcal{A}$ and by $\mathrm{OPT}(I)$ the cost of an optimal packing for

*This research was partially supported by CNPq (Proc. 470608/01–3, 478818/03–3, 306526/04-2 and 490333/04-4) and ProNEx–FAPESP/CNPq (Proc. 2003/09925-5).

†Instituto de Computação — Universidade Estadual de Campinas, Caixa Postal 6176 — 13084–971 — Campinas–SP — Brazil, {eduardo.xavier,fkm}@ic.unicamp.br.

instance I. An asymptotic polynomial time approximation scheme (APTAS) is a polynomial time algorithm $\mathcal{A}$ such that, for a given ε, it satisfies $\mathcal{A}(I) \leq (1+O(\varepsilon))\mathrm{OPT}(I)+K$, for some constant K, for any instance I.

The VBPCC problem was first considered by Dawande et. al [2, 1] where a tentative of an APTAS was considered. We observed that their algorithm does not lead to an APTAS as claimed. First of all, they do a linear rounding step of the list of items L and then obtain an optimal packing for the new list. Doing this they do not guarantee a packing for the original items because of the color constraints. To pack the small items they use a First Fit strategy, and claim that each bin, perhaps a constant number of bins, are filled by at least $(1 - O(\varepsilon))$, but this is also not true due to the color constraints.

In this report we present an APTAS for this problem. In the linear rounding step we separate items by colors and generate all possible packings for the rounded items. To pack the small items we use another strategy.

As was noticed by Dawande et. al [2, 1], we only use bins such that their size are at least ε, since this condition does not affect too much the cost of the solution, i.e, the algorithm remains an APTAS.

2 The Algorithm of Dawande, Kalagnanam and Sethuraman

In this section we give a brief description of the algorithm of Dawande et. al [2, 1] and present the points where their algorithm fails.

Let $I = (L, s, c, w, p)$ be an instance for the VBPCC problem and let L_b be the items in L with size at least ε^2 (big items) and let L_s be the remaining items in L (small items). Given two lists X and Y, we denote by $X\|Y$ the concatenation of these two lists.

Let $n = |L_b|$. The algorithm sorts the list L_b in non-increasing order of size and partition this list into groups (lists) $L_1, \ldots, L_M$, each one with $\lceil n\varepsilon^2 \rceil$ items, except perhaps in the last list that can have less than $\lceil n\varepsilon^2 \rceil$ items. Call the first item in each group as the group-leader. Let L'_i be the list having $|L'_i| = |L_i|$ items, where each item has size equal to the size of the group-leader of L_i. Let $L' = L'_1\| \ldots \|L'_M$.

For the list L' it is possible to generate all configurations of bins in constant time since the number of different items size is bounded by a constant M, the number of different item colors is also bounded by a constant C and the maximum number of items that can be packed in a bin is $1/\varepsilon^2$. Let $t = MC$. Given an item size and an item color, denote by d_i the number of items of this type $i \in [t]$.

Let N be the total number of bin configurations. Let x_j be a variable that represents the number of times that a configuration $j \in [N]$ is used in a solution, a_{ij} be the coefficient that represents the number of times an item type $i \in [t]$ is used in configuration j and w_j the size of the bin used in configuration j. The next step of the algorithm is to solve the following linear programming:

$$\min \sum_{j=1}^{N} w_j x_j$$

$$\begin{aligned} \sum_{j=1}^{N} a_{ij} x_j &\geq d_i && \forall\, i \in [t] \quad (1) \\ x_j &\geq 0 && \forall\, j \in [N], \quad (2) \end{aligned} \qquad \text{(LP)}$$

The algorithm solves this linear program and uses the solution obtained by rounding up the variables x. The solution is a packing for the list L' that is used to generate a packing for the list L_b.

The next step of the algorithm is to pack the small items into the solution provided by the linear programming. To do this, it uses a first-fit strategy: Pack an item in the first bin that has enough space to accomodate it and that satisfy the color constraints.

The list L_b is partitioned into lists $L_1 \| \ldots \| L_M$ and a list L' is obtained from L by a linear rounding step. Let L''_i be a list where $|L''_i| = |L_i|$, and each item has size equal to the group-leader of the list L_{i+1}, for $i = 1, \ldots, M-1$, and L''_M be an empty list. Let $L'' = L''_1 \| \ldots \| L''_M$. Clearly $\mathrm{OPT}(L'') \leq \mathrm{OPT}(L_b)$.

Dawande et. al [2, 1] claimed that the following relation is valid

$$\mathrm{OPT}(L') \leq \mathrm{OPT}(L'') + \lceil n\varepsilon^2 \rceil \leq \mathrm{OPT}(L_b) + \lceil n\varepsilon^2 \rceil,$$

given the argument that L' and L'' differ only in their first and last groups. Given a packing for the list L'' they mention that it is easy to construct a packing for the list $L'_2 \| \ldots \| L'_M$, since $|L'_i| = |L''_{i-1}|$, for $i = 2, \ldots, M$, and their items size are the same. But the this can be false. Notice that the color of the items of L'_i and L''_{i-1} may be different. Therefore, it is not clear how to construct a packing for $L'_2 \| \ldots \| L'_M$ given a packing for L''.

Let B be the number of bins used by their algorithm. After packing the small items using the First-Fit strategy, they claimed that at least $B - \lceil \frac{C}{p} \rceil$ bins have residual capacity at most ε. This is also not true. Suppose all small items have different colors from the big items. It is easy to construct examples where optimal packings for the big items given by the linear programming have all bins with p different colors and the residual space is larger than a given ε.

3 An APTAS for the VBPCC Problem

In this section we present an APTAS for the VBPCC problem. In the next subsection we show how to pack big items doing a linear rounding for each different color. The algorithm to pack the big items generates a polynomial number of packings for the big items, and also provide information of how to pack small items. In the following subsection, we present an algorithm to pack the small items that is based in the solution of a linear program. The algorithm generates a polynomial number of packings such that at least one is very close to the optimal.

3.1 Packing Big Items with Linear Rounding

Let L_b be the list of items in L with size at least ε^2 (big items) and let L_s be the remaining items in L (small items). In this section we show how to do the linear rounding for the big items and

generate a packing for them.

The algorithm that packs the list L_b, which we denote by $\mathrm{A_{LR}}$, uses the linear rounding technique, presented by Fernandez de la Vega and Lueker [3], and considers only items with size at least ε^2. The algorithm $\mathrm{A_{LR}}$ returns a pair $(\mathcal{P}_B, \mathbb{P})$, where $\mathcal{P}_B$ is a packing for a list of very big items and $\mathbb{P}$ is a set of packings for the remaining items of L_b.

For the use of the linear rounding technique, we use the following notation: Given two lists of items X and Y, let $X_1, \ldots, X_C$ and $Y_1, \ldots, Y_C$ be the partition of X and Y respectively in colors, where X_c and Y_c have only items of color c for each $c \in [C]$. We write $X \preceq Y$ if there is an injection $f_c : X_c \to Y_c$ for each $c \in [C]$ such that $s(e) \leq s(f(e))$ for all $e \in X_c$. Given two lists L_1 and L_2 we denote by $L_1 \| L_2$ the concatenation of these lists.

For any instance X, denote by $\overline{X}$ the instance with precisely $|X|$ items with size equal to the size of the smallest item in X. Clearly, $\overline{X} \preceq X$.

The algorithm also uses a variant of the First-Fit (FF) algorithm to pack colored items. In this case, an item is packed in the first bin that have space to pack the item and satisfy the color constraints.

The algorithm $\mathrm{A_{LR}}$ is presented in Figure 1. It consists in the following: Let $L_1, \ldots, L_C$ be the partition of the input list L_b into colors $1, \ldots, C$ and let $n_c = |L_c|$ for each color c. The algorithm $\mathrm{A_{LR}}$ sorts each list L_c in non-increasing order of size and them partition the list L_c into at most $M = \lceil 1/\varepsilon^3 \rceil$ groups $L_c^1, L_c^2, \ldots, L_c^M$, where $L_c = L_c^1 \| \ldots \| L_c^M$. Each group with $\lfloor n_c \varepsilon^3 \rfloor$ items except perhaps the last list (with the smallest items) that can have less than $\lfloor n_c \varepsilon^3 \rfloor$ items.

Let $L_B = \cup_{c=1}^{C} L_c^1$. The algorithm generates a packing $\mathcal{P}_B$ of L_B with cost at most $O(\varepsilon)\mathrm{OPT}(I)$ and a set $\mathbb{P}$ with a polynomial number of packings for the items in $L_b \setminus L_B$. The packing $\mathcal{P}_B$ is generated by the algorithm FF with bins of size 1.

The algorithm generates a set of packings $\mathbb{Q}$, of polynomial size, for the list $(\overline{L_1^1} \| \ldots \| \overline{L_1^{M-1}} \| \ldots \| \overline{L_C^1} \| \ldots \| \overline{L_C^{M-1}})$. This can be done in polynomial time as the next lemma guarantees.

Lemma 3.1 *Given an instance $I = (L_b, s, c, w, p)$, where the number of distinct items sizes of each color is at most a constant M, the number of different colors is bounded by a constant C and each item $e \in L_b$ has size $s_e \geq \varepsilon^2$, then there exists a polynomial time algorithm that generates all possible packings of L_b. Moreover, each bin of each generated packing has an indication of the possible colors that may be used by further small items.*

Proof. The number of items in a bin is bounded by $y = 1/\varepsilon^2$. The number of distinct type of items is bounded by MC. The number of different configurations of bins is bounded by $r' = \binom{y+MC+1}{y}$. If we want to indicate the colors of small items that should be packed in each configuration, the number of different configurations will be $r = r'2^C$, which is a constant. Notice that we only generate configurations that satisfy the color constraints.

For each given configuration, we pack it with the smallest bin that has enough space to pack the configuration. The number of all feasible packings is bounded by $\binom{n+r}{n}$, which is bounded by $(n+r)^r$, which in turn is polynomial in n. □

Since $\overline{L_c^i} \succeq L_c^{i+1},\ i = 1, \ldots, M-1$ for each color c, it is easy to construct a packing for the list $L_1^2 \| \ldots \| L_1^M \| \ldots \| L_C^2 \| \ldots \| L_C^M$, given a packing for the list $(\overline{L_1^1} \| \ldots \| \overline{L_1^{M-1}} \| \ldots \| \overline{L_C^1} \| \ldots \| \overline{L_C^{M-1}})$.

The following is valid for the packing $\mathcal{P}_B$ of the list L_B.

ALGORITHM $\mathrm{A_{LR}}(L_b)$

Input: List L_b with n items, each item $e \in L_b$ with size $s_e \geq \varepsilon^2$.

Output: A pair $(\mathcal{P}_B, \mathbb{P})$, where $\mathcal{P}_B$ is a packing and $\mathbb{P}$ is a set of packings, where $\mathcal{P}_B \cup \mathcal{P}'$ is a packing of L_b for each $\mathcal{P}' \in \mathbb{P}$.

1. Partition L_b into lists L_c for each color $c = 1, \ldots, C$ and let $n_c = |L_c|$.

2. Sort each list L_c in non-increasing order of items size.

3. Partition each list L_c into $M \leq \lceil 1/\varepsilon^3 \rceil$ groups $L_c^1, L_c^2, \ldots, L_c^M$, such that

$$\overline{L_c^i} \succeq L_c^{i+1},\ i = 1, \ldots, M-1$$
$$\text{where } |L_c^i| = q_c = \lfloor n_c \varepsilon^3 \rfloor \text{ for all } i = 1, \ldots, M-1,$$
$$\text{and } |L_c^M| \leq q_c.$$

4. Let $L_B = \cup_{c=1}^{C} L_c^1$.

5. Let $\mathcal{P}_B$ be a packing of L_B obtained by the algorithm FF with bins of size 1.

6. Let $\mathbb{Q}$ be the set of all possible packings over the list $(\overline{L_1^1} \| \ldots \| \overline{L_1^{M-1}} \| \ldots \| \overline{L_C^1} \| \ldots \| \overline{L_C^{M-1}})$, according to Lemma 3.1.

7. Let $\mathbb{P}$ be the set of packings for the items in $(L_1^2 \| \ldots \| L_1^M \| \ldots \| L_C^2 \| \ldots \| L_C^M)$, using the packings $\mathcal{Q} \in \mathbb{Q}$.

8. Return $(\mathcal{P}_B, \mathbb{P})$.

Figure 1: Algorithm to obtain packings for items with size at least ε^2.

Lemma 3.2 $w(\mathcal{P}_B) \leq C\varepsilon \mathrm{OPT}(I)$.

Proof. Notice that the algorithm FF packs at least one item per bin and since $|L_B| \leq Cn\varepsilon^3$ and each item has size at least ε^2, we have $|L_B| \leq C\varepsilon \mathrm{OPT}(I)$.

□

3.2 Packing the small items

Observe that algorithm $\mathrm{A_{LR}}$ generates a packing for very big items that costs at most $C\varepsilon \mathrm{OPT}(I)$, and a set $\mathbb{P}$ of packings for the remaining big items. For a given packing $\mathcal{P} \in \mathbb{P}$, the algorithm marked colors of small items that should be packed in each bin of $\mathcal{P}$. To pack the small items we use a solution given by a linear program.

Let $\mathcal{P} = \{B_1, \ldots, B_k\}$ be a packing of the list of items L_b and suppose we have to pack a list L_s of small items, with size at most ε^2, into $\mathcal{P}$. The packing of the small items is obtained from a solution of a linear program. Let $N_i \subseteq [C]$ be the set of possible colors that may be used to pack the small items in the bin B_i of the packing $\mathcal{P}$. For each color $c \in N_i$, define a non-negative variable x_c^i. The variable x_c^i indicates the total size of small items of color c to be packed in the bin B_i. Denote by $s(B_i)$ the total size of items already packed in the bin B_i and by $w(B_i)$ the capacity of bin B_i. Consider the following linear program denoted by LPS:

$$\max \sum_{i=1}^{k} \sum_{c \in N_i} x_c^i$$

$$\begin{aligned} s(B_i) + \sum_{c \in N_i} x_c^i &\leq w(B_i) && \forall\, i \in [k] \quad (1) \\ \sum_{i=1}^{k} x_c^i &\leq s(S_c) && \forall\, c \in [C], \quad (2) \end{aligned} \qquad \text{(LPS)}$$

where S_c is the set of small items of color c in S.

The constraint (1) guarantees that the items packed in each bin satisfy its capacities and constraint (2) guarantees that variables x_c^i is not greater than the total size of small items.

Given a packing $\mathcal{P}$, and a list L_s of small items, the algorithm first solves the linear program LPS, and then packs small items in the following way: For each variable x_c^i it packs, while possible, the small items of color c into the bin B_i, so that the total size of the packed small items is at most x_c^i. The possible remaining small items are packed using the algorithm FF into new bins of size 1. The algorithm to pack small items has polynomial time, since the linear program *LPS* can be solved in polynomial time.

The small items that are packed into new bins use at most

$$\lceil \frac{(s(L_s) - \sum_{i=1}^{k} \sum_{c \in N_i} x_c^i)}{(1-\varepsilon^2)} + \frac{|\mathcal{P}|\varepsilon^2 C}{(1-\varepsilon^2)} \rceil + \lceil C/p \rceil$$

new bins, since each bin is filled by at least $(1-\varepsilon^2)$ except perhaps by at most $\lceil C/p \rceil$ bins.

The algorithm packs the small items in each packing $\mathcal{P} \in \mathbb{P}$. In the end, the algorithm generates another set of packings $\mathbb{P}'$ for all items. At least one of the generated packings has cost at most $(1 + O(\varepsilon))\mathrm{OPT}(I) + K$, for a constant K. The algorithm returns the packing with smallest cost.

Now we prove that the presented algorithm is an APTAS for the VBPCC problem.

Theorem 3.3 *Let $I = (L, s, c, w, p)$, be an instance for the VBPCC problem. The packing $\mathcal{P}$ returned by the algorithm satisfy $w(\mathcal{P}) \leq (1 + O(\varepsilon))\mathrm{OPT}(I) + K$, where K is a constant.*

Proof. Let O be an optimal packing for instance I. Let O' be the packing without the small items and with the big items rounded according to the linear rounding of algorithm $\mathrm{A_{LR}}$. Assume that each bin of O' has an indication of the colors of small items used in the corresponding bin of O. Clearly the packing $O' \in \mathbb{Q}$, except that it may use smaller bins than the ones used in O.

When the algorithm generates a packing $\mathcal{P}$ for the list $L_1^2 \| \ldots \| L_1^M \| \ldots \| L_C^2 \| \ldots \| L_C^M$ using the packing O' with items $(\overline{L_1^1} \| \ldots \| \overline{L_1^{M-1}} \| \ldots \| \overline{L_C^1} \| \ldots \| \overline{L_C^{M-1}})$, it is true that $w(\mathcal{P}) \leq w(O)$ since in $\mathcal{P}$ we probably use bins of smaller size for each given configuration of big items.

Let $\mathcal{P} = \{B_1, \ldots, B_k\}$. Notice that we must have

$$w(O) \geq w(\mathcal{P}) + (s(L_s) - \sum_{i=1}^{k} \sum_{c \in N_i} x_c^i).$$

The total size of small items that are packed into new bins is at most

$$(s(L_s) - \sum_{i=1}^{k} \sum_{c \in N_i} x_c^i) + |\mathcal{P}|\varepsilon^2 C.$$

The algorithm packs small items in bins of size 1 obtaining a new packing $\mathcal{P}'$. The total cost of the packing $\mathcal{P}'$ is

$$\begin{aligned} w(\mathcal{P}') &\leq w(\mathcal{P}) + \lceil \frac{(s(L_s) - \sum_{i=1}^{k} \sum_{c \in N_i} x_c^i)}{(1-\varepsilon^2)} + \frac{|\mathcal{P}|\varepsilon^2 C}{(1-\varepsilon^2)} \rceil + \lceil C/p \rceil & (1) \\ &\leq \frac{w(O)}{(1-\varepsilon^2)} + \frac{|\mathcal{P}|\varepsilon^2 C}{(1-\varepsilon^2)} + \lceil C/p \rceil + 1 & (2) \\ &\leq \frac{w(O)}{(1-\varepsilon^2)} + \frac{\varepsilon C w(O)}{(1-\varepsilon^2)} + \lceil C/p \rceil + 1. & (3) \end{aligned}$$

The last inequality follow from the fact that $|\mathcal{P}| \leq |O|$ and the smallest size of a bin is ε. Using this result and Lemma 3.2 we conclude the proof. □

References

[1] M. Dawande, J. Kalagnanam, and J. Sethuranam. Variable sized bin packing with color constraints. Technical report, IBM, T.J. Watson Research Center, NY, 1998.

[2] M. Dawande, J. Kalagnanam, and J. Sethuranam. Variable sized bin packing with color constraints. *First Brazilian Symposium on Graph, Algorithms and Combinatorics. Eletronic Notes in Discrete Mathematics*, 7:5, 2001.

[3] W. Fernandez de la Vega and G. S. Lueker. Bin packing can be solved within $1 + \epsilon$ in linear time. *Combinatorica*, 1(4):349–355, 1981.