

A note on the approximability of cutting stock problems [☆]

G.F. Cintra ^a, F.K. Miyazawa ^{b,*}, Y. Wakabayashi ^c, E.C. Xavier ^b

^a *Faculdade de Computação e Informática, Universidade Presbiteriana Mackenzie, Rua da Consolação 896, 01302-907 São Paulo-SP, Brazil*

^b *Instituto de Computação, Universidade Estadual de Campinas, Caixa Postal 6176, CEP 13084-971 Campinas-SP, Brazil*

^c *Instituto de Matemática e Estatística, Universidade de São Paulo, Rua do Matão, 1010, 05508-090 São Paulo-SP, Brazil*

Received 30 September 2004; accepted 27 September 2005

Available online 12 June 2006

Abstract

Cutting stock problems and *bin packing* problems are basically the same problems. They differ essentially on the variability of the input items. In the first, we have a set of items, each item with a given multiplicity; in the second, we have simply a list of items (each of which we may assume to have multiplicity 1). Many approximation algorithms have been designed for packing problems; a natural question is whether some of these algorithms can be extended to cutting stock problems. We define the notion of “well-behaved” algorithms and show that well-behaved approximation algorithms for one, two and higher dimensional bin packing problems can be translated to approximation algorithms for cutting stock problems with the same approximation ratios.

© 2006 Elsevier B.V. All rights reserved.

Keywords: Bin packing; Cutting stock; Approximation algorithm

1. Introduction

Cutting stock problems are of great interest, both from a theoretical and a practical point-of-view. Their applications go from packing of items into boxes or containers, to cutting of fabrics, hardboards, glasses, foams, etc. The exact computational complexity status of these problems is unknown. It seems that the decision versions of these problems

may not be included in NP and that we can only assume that they lie somewhere below EXPSpace.

In this paper we show that some approximation algorithms for bin packing problems give rise to approximation algorithms for cutting stock problems. More precisely, according to the typology proposed by Wäscher, Haussner and Schumann [18], the problems we consider here are the single bin-size bin packing (SBSBP) and the single stock-size cutting stock (SSSCS). In d -dimensional SBSBP problems, we are given a list L of n items, where each item $i \in L$ is a d -dimensional parallelepiped, and we are asked to pack the elements of L into a minimum number of unit-capacity d -dimensional parallelepipeds. The items have to be packed orthogonally and oriented in all dimensions. Furthermore, no

[☆] This research was partially supported by CNPq (Proc. 478818/03-3, 306526/04-2, 308138/04-0 and 490333/04-4), ProNEx-FAPESP/CNPq (Proc. 2003/09925-5) and CAPES.

* Corresponding author. Tel.: +55 19 3788 5882; fax: +55 19 3788 5847.

E-mail address: fkmi@ic.unicamp.br (F.K. Miyazawa).

two items can overlap in the packing. In d -dimensional SSSCS problems, we are given additionally a (positive integer) demand d_i (multiplicity) for each item $i \in L$. Therefore, SBSBP problems can be considered particular cases of SSSCS problems, where all demands are equal to 1. Note, however, that although an instance I for a SSSCS problem can be trivially translated to an instance I' for the corresponding SBSBP problem, the size of I' may be exponential in the size of I . This means that such a trivial translation is not a good approach to tackle SSSCS problems.

To simplify the notation we denote by 1CS, 2CS and 3CS the one, two and three-dimensional SSSCS problems, respectively; and by 1BP, 2BP and 3BP the corresponding SBSBP problems. For the latter, several approximation algorithms have appeared in the literature [12,7,3,1,13,6,4,5]. Curiously, despite the similarity of the problems, we did not find references to approximation algorithms for cutting stock problems. Pioneering works on these problems were carried out by Gilmore and Gomory [9–11] in the early 1960s, and since then many contributions have appeared [14–17]. We refer the reader to Cheng et al. [2] for a survey.

In this note we discuss how to extend some approximation algorithms for SBSBP problems to approximation algorithms for SSSCS problems through the notion of “well-behaved” algorithm. In Section 3 we consider the 1CS problem. We define the concept of well-behaved algorithm and show that any well-behaved algorithm for the 1BP problem can be translated to an algorithm for the 1CS problem. In Section 4 we mention how to obtain similar results for higher dimensional SSSCS problems. We assume that the reader is familiar with the algorithms for the 1BP problem we mention here: NF (Next Fit), FF (First Fit), BF (Best Fit), NFD (Next Fit Decreasing), FFD (First Fit Decreasing), BFD (Best Fit Decreasing), and H_M (Harmonic).

2. Notation

An instance $I = (L, s, d)$ of the 1CS problem consists of a list L of elements, in which each element $e \in L$ has size $s_e \in (0, 1]$ and demand $d_e \in \mathbb{Z}^+$; thus $s = (s_e)_{e \in L}$ and $d = (d_e)_{e \in L}$. The number of items in the instance I , which we denote by $\|I\|$, is the sum $\sum_{e \in L} d_e$. That is, the number of items is at least the number of elements of L . The demand d_e of an ele-

ment e indicates that there is a multiplicity of d_e items of the element of size s_e . We say that an item i corresponding to an element $e \in L$ is an item of type e . That is, in the instance I there are d_e items of type e .

For any structure T , we denote by $\langle T \rangle$ the size in bits of the representation of T . Given k lists Q_1, \dots, Q_k , where $Q_i = (a_1^i, \dots, a_{n_i}^i)$, we denote by $Q = Q_1 \parallel \dots \parallel Q_k$ the concatenation of these lists, defined as the list $Q = (a_1^1, \dots, a_{n_1}^1, \dots, a_1^k, \dots, a_{n_k}^k)$. The number of elements of a list or a set S is denoted by $|S|$.

If $L = (a_1, \dots, a_n)$ then $expand(L, s, d)$ denotes the list $L' = (s_1^1, \dots, s_{d_1}^1, \dots, s_1^n, \dots, s_{d_n}^n)$, where $s_j^i = s(a_i)$, for $1 \leq j \leq d_i$. Given an instance L' of the 1BP problem, we denote by $condense(L')$ the triple (L, s, d) , where $L' = expand(L, s, d)$ and $|L|$ is minimum.

For a given instance $I = (L, s, d)$, a one-dimensional bin B can be represented (or described) by a pair (L_B, d_B) , where $L_B \subseteq L$, $0 \leq d_B(e) \leq d(e)$ for each $e \in L_B$. We say that such a pair (L_B, d_B) is a bin type for I . Clearly, $\langle B \rangle$ is bounded by a polynomial in $\langle I \rangle$.

The *eq-partition* (equal partition) of a list Q is the list (Q_1, \dots, Q_k) , where k is minimum and (i) $Q = (Q_1 \parallel \dots \parallel Q_k)$; (ii) $e' = e''$ for $e', e'' \in Q_i$, $1 \leq i \leq k$. This definition also applies to lists whose items are bins.

3. One-dimensional single stock-size cutting stock problem

The one-dimensional single stock-size cutting stock (1CS) problem can be defined as follows:

Problem 1. (1CS) Given an instance $I = (L, s, d)$ as defined above, find a packing of the items in I into the minimum number of unit-capacity bins.

A natural approach to obtain approximation algorithms for the 1CS problem is to adapt known algorithms for the 1BP problem. As we mentioned before, the naive approach that transforms a given instance I for the 1CS problem into the list $expand(I)$ and applies an algorithm for the 1BP problem on this list is flawed as both $expand(I)$ and the size of the packing that is produced may be exponential in the size of I . Of course, expansions of I may be easily avoided, so the main concern is whether we can adapt the algorithms so as to produce solutions with *short descriptions* (that is,

descriptions that are polynomial in the size of I). Putting in a more general setting, we would like to address the following question: which properties should an algorithm for the 1BP problem satisfy in order to be transformable into an algorithm for the 1CS problem that produces a packing with a short description? In what follows, we define the notion of well-behaved algorithm, and give an answer to this question.

Definition 3.1. An algorithm \mathcal{A}' that receives an input list L' for the 1BP problem is well-behaved if it satisfies the following two properties:

- P1. *Stable order property.* The algorithm packs consecutively the equal-sized items that are consecutive in the input list L' . More precisely, if (L'_1, \dots, L'_p) is an eq-partition of L' then the algorithm packs the items of each L'_i consecutively. Formally, we may consider that the algorithm behaves as follows:
 - 1.1. Take $(L'', s, d) := \text{condense}(L')$.
 - 1.2. Take $L := \text{expand}(L'', s, d)$, where L'' is a permutation of L' .
 - 1.3. Pack the items following the order given by L .
- P2. *Grouping property.* To pack an item, the algorithm does the following.
 - 2.0. Suppose (L_1, \dots, L_p) is an eq-partition of L , where L is the list mentioned in the previous property. The algorithm \mathcal{A}' packs first the list L_1 .
 - 2.1. Before packing the first item of a list L_i ,
 - 2.1.1. let $\mathcal{B} = (B_1, B_2, \dots, B_k)$ be the list of existing non-empty bins, in the order they were generated.
 - 2.1.2. Let $(\mathcal{B}_1, \dots, \mathcal{B}_q)$ be the eq-partition of \mathcal{B} . Each list $\mathcal{B}_j = (B_j^1, \dots, B_j^{n_j})$ is said to be a *group*.
 - 2.1.3. Let \mathcal{B}_{q+1} be a group with sufficiently many empty bins.
//New bins are obtained from this group.
 - 2.2. To pack the first item $e \in L_i$,
 - 2.2.1. the algorithm packs e into a bin $B_j^t \in \mathcal{B}_j$, for some j , such that either $j \leq q$ or $(j = q + 1 \text{ and } t = 1)$.
 - 2.2.2. Now B_j^t becomes the *current bin* and \mathcal{B}_j the *current group*.
 - 2.3 While the list L_i is non-empty, to pack the next item $e \in L_i$,
 - 2.3.1. if possible, packs e into the current bin B_j^t .
 - 2.3.2. If \mathcal{A}' fails in the previous step and $B_j^{t+1} \in \mathcal{B}_j$ then \mathcal{A}' packs e into B_j^{t+1} . Now, B_j^{t+1} becomes the current bin.
 - 2.3.3. If \mathcal{A}' fails in the previous step, \mathcal{A}' packs e into a bin $B_{j'}^1$, for some group $\mathcal{B}_{j'}$. Now, $B_{j'}^1$ becomes the current bin and $\mathcal{B}_{j'}$ the current group.

It is not hard to check that NF, FF, BF, H_M , NFD, FFD, and BFD are well-behaved algorithms. Now using this fact, and the concept of a short description of a packing, defined below, we can derive our first result.

Definition 3.2. Let $I = (L, s, d)$ be an instance for the 1CS problem and \mathcal{P} a packing of I . A *description* of \mathcal{P} is a list \mathcal{D} of pairs (B, b_B) , where $B = (L_B, d_B)$ is a bin type for I and b_B is the multiplicity of the bin type B in the packing \mathcal{P} ; and if B_e is the number of items of type e in the bin B , then $\sum_{(B, b_B) \in \mathcal{D}} b_B B_e = d_e$ for any $e \in L$. We say that \mathcal{D} is a *short description* if the bin types B are all distinct and $\langle \mathcal{D} \rangle$ is polynomially bounded in $\langle I \rangle$.

Theorem 3.3. Let I be an instance for the 1CS problem and \mathcal{A}' an algorithm for the 1BP problem. If \mathcal{A}' is well-behaved, then there exists a polynomial time algorithm \mathcal{A} that produces a packing that is precisely the packing produced by \mathcal{A}' on the list $\text{expand}(I)$, differing possibly only on the description of the packing.

Proof. Let $I = (L, s, d)$, and L' be the permutation of $\text{expand}(I)$ that is obtained as a consequence of the stable order property P1, after applying \mathcal{A}' to $\text{expand}(I)$. Assume that (L_1, \dots, L_k) is the eq-partition of L' .

Let $(\mathcal{B}_1, \dots, \mathcal{B}_q)$ be an eq-partition of the bins generated by algorithm \mathcal{A}' for the items $L_1 \parallel \dots \parallel L_i$ and let \mathcal{B}_{q+1} be a list of sufficiently many empty bins. Clearly, the algorithm \mathcal{A} may use a short description of $(\mathcal{B}_1, \dots, \mathcal{B}_q)$. Now, consider the packing of the items of the list L_{i+1} . To pack the first item of L_{i+1} , the algorithm chooses a bin B_j^t of a group $\mathcal{B}_j = (B_j^1, \dots, B_j^{n_j})$, where $j \leq q + 1$, and tries to pack the items of L_{i+1} in the bins $(B_j^1, \dots, B_j^{n_j})$, consecutively. If it fails to pack all items of L_{i+1} in these bins, it continues in the same fashion moving to the first bin of another group.

Suppose that $\mathcal{B}_{i_1}, \dots, \mathcal{B}_{i_m}$ is the sequence of groups in the list $\mathcal{B} = (\mathcal{B}_1, \dots, \mathcal{B}_{q+1})$ (of bins) in which the algorithm \mathcal{A}' has packed the items of L_{i+1} , in the order the packing has occurred. Since \mathcal{A}' is a well-behaved algorithm, it packs the items in consecutive bins of each group. First suppose that $m > 1$. In this case, after packing the items of L_{i+1} in the group \mathcal{B}_{i_1} , the number of different bins increases by at most 1 (note that the packing of the items may start in any of the bins of the group). After packing items of L_{i+1} in the groups $\mathcal{B}_{i_2}, \dots, \mathcal{B}_{i_{m-1}}$, the number of different bins does not increase. After packing the remaining items of L_{i+1} in the group \mathcal{B}_{i_m} , the number of bins increases by at most 2. Therefore, the number of different bins after packing the whole list L_{i+1} increases by at most 3. When $m = 1$, the number of different bins increases by at most 2.

Notice that with a simple calculation, the algorithm \mathcal{A} can figure out how many items of L_{i+1} can be packed in a bin of a group \mathcal{B}_j and how many bins of this group it uses to pack these items. After packing all the lists L_1, \dots, L_k , we can conclude that the number of different bins is at most $3k$. This shows that (mimicking the behavior of algorithm \mathcal{A}') we may design a polynomial time algorithm \mathcal{A} that produces a packing that has a short description. \square

Denote by NF_{cs} , FF_{cs} , BF_{cs} , H_{Mcs} , NFD_{cs} , FFD_{cs} and BFD_{cs} the algorithms NF, FF, BF, H_M , NFD, FFD and BFD, respectively, adapted for the 1CS problem that generate packings with short descriptions.

Corollary 3.4. *The algorithm NF_{cs} (respectively, NF_{cs} , FF_{cs} , BF_{cs} , H_{Mcs} , NFD_{cs} , FFD_{cs} and BFD_{cs}) has asymptotic performance bound 2 (respectively, 1.7, 1.7, 1.691, ..., 1.691, ..., 11/9 and 11/9). The bound for H_{Mcs} holds when $M \rightarrow \infty$.*

Considering the same ideas of short descriptions presented for the well-behaved algorithms, we may also convert the AFPTAS of Fernandez de la Vega and Lueker [7] into an AFPTAS for the 1CS problem. That is, the following result holds.

Theorem 3.5. *There exists an AFPTAS for the 1CS problem.*

4. Two and higher dimensional single stock-size cutting stock problems

An instance $I = (L, w, h, d)$ for the 2CS problem consists of a list of elements L , each element

$e \in L$ with width $w_e \in (0, 1]$, height $h_e \in (0, 1]$ and demand $d_e \in \mathbb{Z}^+$. Most of the notation we used in the context of the 1CS problem can be extended easily to the context of 2CS, as for example, $\text{expand}(L, w, h, d)$, $\text{condense}(L)$, etc. For this problem we can also define the concept of well-behaved algorithm. Although better definitions may be given, we present a simple definition of *well-behaved algorithm* for the 2CS problem, as this can be extended easily to higher dimensions.

Definition 4.1. An algorithm \mathcal{A} that receives an input list L' for the problem 2BP is well-behaved if it satisfies the following properties:

- Q1. *Stable order property.* The behavior of the algorithm can be described as follows:
 - 1.1. Take $(L'', w, h, d) := \text{condense}(L')$.
 - 1.2. Take $L := \text{expand}(L'', s, d)$, where L'' is a permutation of L' .
 - 1.3. Pack the items following the order given by L .
- Q2. *Level oriented property.* The strategy used by the algorithm to produce a packing is the following:
 - 2.1 The algorithm generates a list \mathcal{L} of levels using a well-behaved algorithm for the 1BP problem.
 - 2.2 The algorithm uses a well-behaved algorithm for the 1BP problem to pack the levels of \mathcal{L} into unit-capacity two-dimensional bins.

Theorem 4.2. *Let I be an instance for the 2CS problem and \mathcal{A}' an algorithm for the 2BP problem. If \mathcal{A}' is a well-behaved algorithm, then there exists a polynomial time algorithm \mathcal{A} that produces a packing that is precisely the packing produced by the algorithm \mathcal{A}' on the list $\text{expand}(I)$, differing possibly only on the description of the packing.*

One of the most famous algorithm for the 2BP problem is the algorithm HFF (Hybrid First Fit), presented by Chung, Garey and Johnson [3]. These authors proved that HFF has an asymptotic performance bound of 2.125, and later Caprara [1] proved that this algorithm has an asymptotic performance bound of 2.077... Frenk and Galambos [8] proved that the next fit variant of the algorithm HFF, which we denote by HNF, has an asymptotic performance bound of 3.382... The algorithm with the best known asymptotic performance bound for

2BP, which we denote by HC, is due to Caprara [1] and has bound $1.691\dots$. These three algorithms are hybrid and use algorithms for the 1BP problem to pack items into levels and levels into two-dimensional bins. Moreover, all algorithms for the 1BP problem used as subroutines have a corresponding version for the 1CS problem, given by Corollary 3.4 or by Theorem 3.5.

Corollary 4.3. *There exists an algorithm HNF_{CS} (resp. HFF_{CS} , HC_{CS}) with asymptotic performance bound $3.382\dots$ (resp. $2.077\dots$, $1.691\dots$) for the 2CS problem.*

Most of the ideas presented here can also be extended to higher dimensions. In particular, the 4.84-approximation algorithms of Li and Cheng [13] and of Csirik and van Vliet [6] can be translated to algorithms for the 3CS problem, as they generate packings that consist of levels. We can prove that these algorithms are well-behaved and that the following holds.

Corollary 4.4. *There exist algorithms for the problem 3CS with asymptotic performance bound 4.84.*

References

- [1] A. Caprara, Packing 2-dimensional bins in harmony, in: Proceedings of 43rd Symposium on Foundations of Computer Science, IEEE Computer Society Press, 2002, pp. 490–499.
- [2] C.H. Cheng, B.R. Feiring, T.C.E. Cheng, The cutting stock problem – a survey, International Journal of Production Economics 36 (3) (1994) 291–305.
- [3] F.R.K. Chung, M.R. Garey, D.S. Johnson, On packing two-dimensional bins, SIAM Journal on Algebraic and Discrete Methods 3 (1982) 66–76.
- [4] E.G. Coffman Jr., M.R. Garey, D.S. Johnson, Approximation algorithms for bin packing – an updated survey, in: G. Ausiello, M. Lucertini, P. Serafini (Eds.), Algorithms Design for Computer System Design, Springer-Verlag, New York, 1984, pp. 49–106.
- [5] E.G. Coffman Jr., M.R. Garey, D.S. Johnson, Approximation algorithms for bin packing: a survey, in: D. Hochbaum (Ed.), Approximation Algorithms for NP-hard Problems, PWS Publishing Company, Boston, 1997, pp. 46–93.
- [6] J. Csirik, A. van Vliet, An on-line algorithm for multidimensional bin packing, Operations Research Letters 13 (3) (1993) 149–158.
- [7] W. Fernandez de la Vega, G.S. Lueker, Bin packing can be solved within $1 + \epsilon$ in linear time, Combinatorica 1 (4) (1981) 349–355.
- [8] J.B. Frenk, G. Galambos, Hybrid next fit algorithm for the two-dimensional rectangle bin packing problem, Computing 39 (1987) 201–217.
- [9] P. Gilmore, R. Gomory, A linear programming approach to the cutting stock problem, Operations Research 9 (1961) 849–859.
- [10] P. Gilmore, R. Gomory, A linear programming approach to the cutting stock problem – part II, Operations Research 11 (1963) 863–888.
- [11] P. Gilmore, R. Gomory, Multistage cutting stock problems of two and more dimensions, Operations Research 13 (1965) 94–120.
- [12] N. Karmarkar, R.M. Karp, An efficient approximation scheme for the one dimensional bin packing problem, in: Proceedings of the 23rd Symposium on Foundations of Computer Science, IEEE Computer Society Press, 1982, pp. 312–320.
- [13] K. Li, K-H. Cheng, A generalized harmonic algorithm for on-line multidimensional bin packing, Technical Report UH-CS-90-2, University of Houston, 1990.
- [14] G. Scheithauer, J. Terno, The modified integer round-up property of the one-dimensional cutting stock problem, European Journal of Operational Research 84 (3) (1995) 562–571.
- [15] P.H. Vance, Branch-and-price algorithms for one-dimensional cutting stock problem, Computational Optimization and Applications 9 (3) (1998) 211–228.
- [16] F. Vanderbeck, Computational study of a column generation algorithm for bin packing and cutting stock problems, Mathematical Programming 86 (3) (1999) 565–594.
- [17] G. Wäscher, An LP-based approach to cutting stock problems with multiple objectives, European Journal of Operational Research 44 (2) (1990) 175–184.
- [18] G. Wäscher, H. Haussner, H. Schumann, An improved typology of cutting and packing problems, European Journal of Operational Research, 2006 (this issue).