Journal of Scheduling **9:** 63–69, 2006.

APPROXIMATION ALGORITHMS FOR EXTENSIBLE BIN PACKING*

E.G. COFFMAN, JR.[1] AND GEORGE S. LUEKER[2]

[1] *Electrical Engineering Department, Columbia University, New York, NY 10027*
[2] *Computer Science Department, Donald Bren School of Information and Computer Sciences, University of California, Irvine, CA 92697-3435*

ABSTRACT

In a variation of bin packing called *extensible bin packing*, the number of bins is specified as part of the input, and bins may be extended to hold more than the usual unit capacity. The cost of a bin is 1 if it is not extended, and the size if it is extended. The goal is to pack a set of items of given sizes into the specified number of bins so as to minimize the total cost. Adapting ideas in Grötschel et al. (1981), Grötschel et al. (1988), Karmarkar and Karp (1982), Murgolo (1987), we give a fully polynomial time asymptotic approximation scheme (FPTAAS) for extensible bin packing. We close with comments on the complexity of obtaining stronger results.

1. INTRODUCTION

We study a variation of the bin-packing problem in which we are given a list of n positive numbers $x_1, x_2, \ldots, x_n$ specifying the sizes of n different items, and a positive integer m giving a number of bins, each of nominal capacity 1. We assume that each x_i is rational and expressed as the ratio of two integers. Let the bins be denoted by $b_1, b_2, \ldots, b_m$, and let the level of b_j, written $\ell(b_j)$, be the total size of the items packed into b_j. We use ℓ generically to refer to bin levels in whatever packing is currently under consideration. We allow bins to be filled beyond their nominal capacity, i.e., to be extensible, and define the *cost* of a bin b_j to be $\max(1, \ell(b_j))$. The problem is to assign each item to one of the bins so as to minimize the total cost of the m bins, i.e., so as to minimize the objective function

$$\sum_{j=1}^{m} \max(1, \ell(b_j)).$$

see Dell'Olmo et al. (1998) for suggestions of applications for extensible bin packing.

The greedy algorithm is a natural candidate for finding extensible bin packing approximations. According to this algorithm, each successive item is placed into a bin with the lowest current level. In Dell'Olmo et al. (1998) it is shown that, if the x_i are in nonincreasing order, the greedy packing gives a 13/12 approximation. The greedy algorithm preceded by a sorting of the list into nonincreasing order is the LPT (Largest Processing Time) algorithm of scheduling theory.

Note that to some extent this problem resembles the variable-size bin packing problem discussed in Murgolo (1987): We effectively allow bins of various sizes with various costs. An important distinction here is that the number of bins to be packed is fixed. (Indeed, without this restriction the optimum solution for our problem would simply pack all items into a single bin.)

We remark that if the number m of bins is fixed, then there is a simple dynamic programming solution in which the states correspond to m-tuples specifying the levels in each bin. Since this is *ex-benevolent* as defined in Woeginger (1999), it follows from the results of Woeginger (1999) that there is a fully polynomial time approximation scheme for the problem.

---

*A preliminary version of this paper appeared in the *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, Washington, D.C., January 2001, pp. 586-588.
Corresponding author: George S. Lueker. E-mail: lueker@ics.uci.edu

For the case in which m is varying, a polynomial time approximation scheme is presented in Alon et al. (1998), as a special case of their more general theorem.

Here we present an algorithm $A(I, \epsilon)$ which takes a problem instance and a parameter ϵ and produces an approximation guaranteeing that

$$A(I, \epsilon) \leq (1 + \epsilon)\,\mathrm{OPT}\,(I) + O(\epsilon^{-1} \log \epsilon^{-1}) \tag{1}$$

in time bounded by a polynomial in n and ϵ^{-1}. Thus, we establish the existence of a fully polynomial time asymptotic approximation scheme. The word *asymptotic* here indicates the presence of the additive term $O(\epsilon^{-1} \log \epsilon^{-1})$ in (1); note that because of this term the results are only of interest if m becomes large. Our methods are similar to those used in Grötschel et al. (1981,1988), Karmarkar and Karp (1982), Murgolo (1987).

Section 2 discusses some simplifying assumptions we can make without loss of generality. Section 3 then presents the approximation scheme. Section 4 makes a few concluding remarks about the complexity of producing stronger results.

2. SIMPLIFYING ASSUMPTIONS

We begin by noting that we can make certain assumptions without significantly affecting the difficulty of the problem. (Some similar observations are made in earlier work, e.g. (Alon et al., 1998; Dell'Olmo et al., 1998).) Say an item x_i is *small* if

$$x_i \leq \frac{\epsilon}{1 + \epsilon}. \tag{2}$$

We note that if our goal is to produce a polynomial-time approximation with the guarantee shown in (1), then we may assume that none of the items to be packed are small. To see this, suppose we have an approximation algorithm A' which guarantees (1) under the assumption that there are no small items. Then we could use this to pack an arbitrary list of items and still satisfy (1) as follows. First use A' to pack all items which are not small. Next pack the small items greedily. Consider two cases:

(a) Packing the small items does not increase the objective function. Then clearly we still satisfy (1).
(b) Packing the small items does increase the objective function. For convenience let ϵ' be the maximum size of a small item, i.e., $\epsilon' = \epsilon/(1+\epsilon)$. Then some small item was packed into a bin whose new level exceeded 1, so each bin was packed to a level of at least $1 - \epsilon'$. Thus, letting $\ell(\cdot)$ represent the final levels of the bins, we have

$$\mathrm{OPT}(I) \geq \sum_{j=1}^{m} \ell(b_j) > m(1 - \epsilon'),$$

and

$$A'(I) \leq \sum_{j=1}^{m} \max(1, \ell(b_j)) \leq \sum_{j=1}^{m} (\ell(b_j) + \epsilon') \leq m\epsilon' + \sum_{j=1}^{m} \ell(b_j).$$

Hence

$$\frac{A'(I) - \mathrm{OPT}(I)}{\mathrm{OPT}(I)} \leq \frac{m\epsilon'}{m(1 - \epsilon')} = \epsilon.$$

Thus we henceforth assume

$$x_i > \frac{\epsilon}{1 + \epsilon} \text{ for } i = 1, 2, \ldots, n. \tag{3}$$

Next say that an item is *big* if its size is at least 1, and note that we can assume without loss of generality that there are no big items. For suppose that there are b big items. Let X_b be any set of $\min(b, m - 1)$ of these items. We claim that there is an optimum packing that packs each item of X_b into a bin by itself; it follows

that a $(1+\epsilon)$-approximation algorithm can easily be constructed from a $(1+\epsilon)$-approximation algorithm for instances without big items. To prove the claim, suppose some optimum packing placed some item x in a bin together with an item in X_b. Then we could simply move x to one of the bins that contained no item of X_b without increasing the cost of the packing. Thus we henceforth assume

$$x_i < 1 \quad \text{for } i = 1, 2, \ldots, n. \tag{4}$$

Next we note that, assuming (4) holds, if the sum of the item sizes equals or exceeds $2m$, then greedily packing all items in arbitrary order is guaranteed to produce the optimum. To see this note that when (4) holds, the greedy method has the property that as long as any bin is packed to a level of less than 1, no bin will be packed to a level of 2 or more. When all items are packed, some bin must be packed to a level of at least 2, so all bins have a level of at least 1. Hence the cost of the packing is simply the sum of the item sizes, so the packing must be optimum. Thus we henceforth assume

$$\sum_{i=1}^{n} x_i < 2m. \tag{5}$$

Finally, we note that under assumptions (3), (4), and (5), there is always an optimum packing that does not pack any bin to a level of 3 or more. To see this, of the finitely many optimum packings, select one that minimizes the quantity

$$|\{j|\ell(b_j) \geq 3\}| + \sum_{j=1}^{m} \max(\ell(b_j) - 3, 0). \tag{6}$$

Suppose this packing had a bin b of level 3 or more. Then by (5), there would also be a bin b' of level less than 2. Thus, by (4), moving an item from b to b' would decrease the quantity in (6) without increasing the packing cost. This contradicts our assumption that we chose an optimum packing that minimized (6). So now, without loss of generality, we restrict our attention to packings that satisfy

$$\ell(b_j) < 3, \quad \text{for } j = 1, 2, \ldots, m. \tag{7}$$

3. A FULLY POLYNOMIAL TIME ASYMPTOTIC APPROXIMATION SCHEME

Our approach is similar to that used for the fully polynomial asymptotic approximation scheme for ordinary bin packing in Karmarkar and Karp (1982) and for variable-size bin packing in Murgolo (1987). Specifically, we use the relationship between optimization problems and separation problems discussed in Grötschel et al. (1981), Grötschel et al. (1988). (One contribution of Karmarkar and Karp (1982) was the use of sophisticated rounding schemes such as geometric grouping. Here we use a relatively simple rounding scheme.)

For convenience, rather than proving a bound of the form in (1), we will show that

$$A(I, \epsilon) \leq (1+\epsilon)^2 \mathrm{OPT}(I) + O(\epsilon^{-1} \log \epsilon^{-1}). \tag{8}$$

Of course, a bound of the form in (1) could then be obtained by a different choice of ϵ.

Let N be the smallest value of j for which

$$\left\lfloor \frac{(1+\epsilon)^j}{\epsilon} \right\rfloor \epsilon^2 \geq 1.$$

Then it is not hard to calculate that, for small ϵ,

$$N \sim \epsilon^{-1} \ln \epsilon^{-1}. \tag{9}$$

We round the items to sizes $s_1, s_2, \ldots, s_N$, where

$$s_j = \begin{cases} \left\lfloor \frac{(1+\epsilon)^j}{\epsilon} \right\rfloor \epsilon^2 & \text{for } 1 \leq j < N \\ 1 & \text{for } j = N. \end{cases}$$

Round the size of each item of size x up to the smallest s_j for which $x \le s_j$. Since we assume all items are of size at least $\epsilon/(1+\epsilon)$, this will increase the sizes of the items bounded by s_1 by a ratio of at most

$$s_1/\frac{\epsilon}{1+\epsilon} \le \left|\frac{1+\epsilon|}{\epsilon}\right| \epsilon^2/\frac{\epsilon}{1+\epsilon} \le (1+\epsilon)^2.$$

The rounding will increase the size of items larger than s_1 by a ratio of at most

$$\begin{aligned}\frac{s_{j+1}}{s_j} &\le \frac{\lfloor (1+\epsilon)^{j+1}\epsilon^{-1}\rfloor}{\lfloor (1+\epsilon)^j\epsilon^{-1}\rfloor} \\ &\le \frac{(1+\epsilon)^{j+1}\epsilon^{-1}}{(1+\epsilon)^j\epsilon^{-1}-1} = \frac{(1+\epsilon)^{j+1}}{(1+\epsilon)^j-\epsilon} \le \frac{(1+\epsilon)^{j+1}}{(1+\epsilon)^{j-1}} \le (1+\epsilon)^2.\end{aligned}$$

Thus the rounding increases the cost of the optimum solution by a factor of at most $(1+\epsilon)^2$. Let n_j be the number of items of size s_j after the rounding.

We can restrict our attention to bins filled to a level of at most 3 by again using reductions as in Section 2. Now list all possible ways of packing a bin to a level of at most 3 with sizes chosen from the s_j; we call each of these ways of packing a bin a *configuration*. Let M be the number of configurations and let C_{ij} be the number of items of size s_j in the ith configuration. Note that the cost of packing a bin according to the ith configuration can be written as

$$\max\left(1, \sum_{j=1}^{N} C_{ij}s_j\right).$$

Much as in Alon et al. (1998), we can now express the problem as an integer program in which the variable z_i tells the number of times we use the ith configuration, as follows:

$$\text{minimize} \quad \sum_{i=1}^{M} z_i \max\left(1, \sum_{j=1}^{N} C_{ij}s_j\right) \tag{10}$$

$$\text{subject to} \quad \sum_{i=1}^{M} z_i\, C_{ij} \ge n_j, \quad \text{for } j = 1, 2, \ldots, N \tag{11}$$

$$\text{and} \sum_{i=1}^{M} z_i = m, \tag{12}$$

$$\text{with } z_i \in \{0, 1, 2, \ldots\}. \tag{13}$$

Line (10) above expresses the goal of minimizing the cost. Line (11) states that we must pack enough items of each size; as usual, we can make this an inequality since a packing that uses too many of certain sizes can always be converted to a feasible packing of no greater cost by simply discarding items. Line (12) states that we use the correct number of bins. Line (13), of course, states that we must use an integral number of each configuration.

Note that we can weaken (12) to

$$\sum_{i=1}^{M} z_i \ge m, \tag{14}$$

without affecting the optimum solution, since a packing that used too many bins could be converted to a feasible packing of no greater cost by simply merging bins.

To compare this to the algorithm in Karmarkar and Karp (1982), note that if we were solving the ordinary bin-packing problem, the set of configurations would only include packings that fit in a single bin, the cost

would be simply the sum of the z_i instead of the quantity given in (10), and constraint (12) or (14) would not be present.

Suppose now that we relax this integer program to a linear program, by allowing the z_i to assume any nonnegative value. This of course cannot increase the cost of the optimum solution. Since the number of inequalities is $N + 1$, we know that there will be a basic feasible solution with at most $N + 1$ non-integer values, so rounding the relaxed solution up to an integer solution would give an integer solution with value at most $3(N + 1)$ above the fractional solution, since each coefficient in the objective function is at most 3. Hence the linear and integer solutions differ by at most $3(N + 1) = O(\epsilon^{-1} \log \epsilon^{-1})$, using (9).

The dual of this linear program (assuming that (12) has been replaced by (14)) can be written as

$$\text{maximize} tm + \sum_{j=1}^{N} u_j n_j$$

$$\text{subject to} t + \sum_{j=1}^{N} u_j C_{ij} \leq \max\left(1, \sum_{j=1}^{N} C_{ij} s_j\right) \text{ for } i = 1, 2, \ldots, M$$

$$\text{with } t, u_j \geq 0.$$

In the terminology of economics, this dual program has an interpretation similar to that given in Karmarkar and Karp (1982): the dual variables u_j are interpreted as prices assigned to the items of size s_j, and the new dual variable t is interpreted as a price assigned to usage of a bin. If we can guarantee that for each feasible configuration the total price of the items plus the bin usage price is at most the cost of the bin, then by summing over all of the bins in a feasible packing we see that the total cost of the bins is at least the total price of the items plus m times the bin usage price. Although this dual may have a huge number of constraints, it can be solved in polynomial time using the methods of Grötschel et al. (1981,1988), Karmarkar and Karp (1982), Murgolo (1987).

At this point it is convenient to switch to vector notation. Let $\mathcal{C}$ be the set of vectors given by the rows of matrix C, so that each element of $\mathcal{C}$ specifies a configuration. Then a nonnegative integer-valued vector $\vec{k} = (k_1, k_2, \ldots, k_N)$ is in $\mathcal{C}$ if and only if

$$\sum_{j=1}^{N} k_j s_j \leq 3.$$

Denoting $\vec{s} = (s_1, s_2, \ldots, s_N)$, $\vec{u} = (u_1, u_2, \ldots, u_N)$, and $\vec{n} = (n_1, n_2, \ldots, n_N)$, we can rewrite the dual program as

$$\begin{gathered} \text{maximize } tm + \vec{n} \cdot \vec{u} \\ \text{subject to } t + \vec{k} \cdot \vec{u} \leq \max(1, \vec{k} \cdot \vec{s}) \text{ for } \vec{k} \in \mathcal{C}, \\ \text{with } t, u_j \geq 0. \end{gathered} \tag{15}$$

Let $\mathcal{K}$ denote the entire feasible space of $(t, u_1, u_2, \ldots, u_N)$. By Grötschel et al. (1988), (6.5.14) Theorem, if we can solve the following strong separation problem in polynomial time, then in polynomial time we can solve (15) and find a basic optimum solution to its dual, i.e., a basic optimum solution to the original linear program.

Strong separation problem: *Given* $\vec{v} = (t, u_1, u_2, \ldots, u_N)$, *either determine that* $\vec{v} \in \mathcal{K}$, *or find a vector* $\vec{c}$, *called a* separator, *such that*

$$\vec{v} \cdot \vec{c} > \max\{\vec{v'} \cdot \vec{c} | \vec{v'} \in \mathcal{K}\}.$$

Again comparing this to the algorithm of Karmarkar and Karp (1982), note that there the dual problem is to maximize $\vec{n} \cdot \vec{u}$ subject to $\vec{k} \cdot \vec{u} \leq 1$ whenever $\vec{k} \cdot \vec{s} \leq 1$, so the separation problem is a standard knapsack problem.

In our case, it is enough to decide for a given t and $\vec{u}$ whether

$$\exists \vec{k} \in \mathcal{C} \text{ such that } t + \vec{k} \cdot \vec{u} > \max(1, \vec{k} \cdot \vec{s}), \tag{16}$$

and find such a $\vec{k}$ when the answer is yes. To do so, recall that each of the components of $\vec{s}$ is an integer multiple of ϵ^2, so we can use dynamic programming to develop a table T giving

$$T(l) = \max_{\vec{k} \in \mathcal{C}} \vec{k} \cdot \vec{u} \text{ subject to } \vec{k} \cdot \vec{s} = l\epsilon^2,$$

for each $l \in \{0, 1, \ldots, \lfloor 3/\epsilon^2 \rfloor\}$. Then (16) holds iff

$$\exists l \in \{0, 1, \ldots, \lfloor 3/\epsilon^2 \rfloor\} \text{ s.t. } t + T(l) > \max(1, l\epsilon^2),$$

and when this is true a $\vec{k}$ satisfying (16) is easily found by standard techniques.

The resulting algorithm runs in time polynomial in the problem size and ϵ^{-1}, and achieves the error bound given in (8).

4. CONCLUDING REMARKS

Thus far we have assumed that the nominal bin capacity is fixed at 1, and that numbers in the input are specified as rationals; to avoid ambiguity, we henceforth call this the *scaled* version of the problem. We use the term *unscaled* extensible bin packing to refer to the form of the problem in which all input are integers, the nominal bin capacity B is given as part of the input, and the cost of a bin filled to level ℓ is $\max(B, \ell)$. Of course these two versions of the problem are polynomially equivalent.

As noted in Alon et al. (1998), since extensible bin packing is strongly NP-complete, it cannot admit a fully polynomial time approximation scheme (FPTAS) (note the absence of the word "asymptotic," i.e., we are now requiring $A(I, \epsilon) \leq (1 + \epsilon)\mathrm{OPT}(I)$ instead of requiring (1)) unless $P = NP$, by an application of Garey and Johnson (1979), Theorem 6.8 and its Corollary. The same sort of argument was used in Hochbaum and Shmoys (1987) to show that makespan scheduling could not admit an FPTAS with performance guarantee $(1 + \epsilon)\mathrm{OPT}(I) + f(\epsilon^{-1})$ for any polynomial f. (To apply Garey and Johnson (1979), Theorem 6.8 and its Corollary to extensible packing, use the unscaled version so that we have integer-valued solutions; then the result carries over immediately to the scaled version.)

The scaling does however affect the question of whether a bound like (1) can be obtained in polynomial time.[1] In particular, for the unscaled version the bound in (1) is not achievable unless $P = NP$. In fact, by an argument like that in Hochbaum and Shmoys (1987) (see also Garey and Johnson (1979), Theorem 6.6), we can show that no bound of the form

$$A(I, \epsilon) \leq (1 + \epsilon)\,\mathrm{OPT}\,(I) + f(\epsilon^{-1}) \tag{17}$$

would be possible for any f for which $\log f(\epsilon^{-1})$ is a polynomial in ϵ^{-1}. To see this, suppose such an approximation algorithm existed. Let

$$s = \epsilon^{-1} f(\epsilon^{-1}). \tag{18}$$

Let sI denote the result of scaling all of the numbers in I by a factor of s; then the encoding length of sI would be polynomially bounded in ϵ^{-1} and the encoding length of I. Applying the supposed approximation algorithm to sI would produce a result satisfying

$$A(sI, \epsilon) \leq (1 + \epsilon)\,\mathrm{OPT}\,(sI) + f(\epsilon^{-1}),$$

so

$$s^{-1} A(sI, \epsilon) \leq (1 + \epsilon)\,\mathrm{OPT}\,(I) + s^{-1} f(\epsilon^{-1}). \tag{19}$$

Since $\mathrm{OPT}(I) \geq 1$, from (18) we have

$$s^{-1} f(\epsilon^{-1}) = \epsilon \leq \epsilon\,\mathrm{OPT}\,(I),$$

so (19) gives an error ratio of $1 + 2\epsilon$ in time bounded by a polynomial in the problem size and ϵ^{-1}, and hence of course bounded by a polynomial in the problem size and $(2\epsilon)^{-1}$. Note that the condition we impose on the approximation here is fairly strict (in particular, the additive term in (17) is required to be independent of I), so this negative result does not imply that there is no fully polynomial time asymptotic approximation scheme for the unscaled version with a less strict performance bound.

ACKNOWLEDGEMENTS

We thank David Eppstein for helpful comments.

NOTE

1. This section extends and corrects comments in the last paragraph of Coffman and Lueker (2001).

REFERENCES

Alon, N., Y. Azar, G. J. Woeginger, and T. Yadid, "Approximation schemes for scheduling on parallel machines," *Journal of Scheduling*, **1**, 55–66 (1998).

Coffman, Jr., E. G. and G. S. Lueker, "Approximation algorithms for extensible bin packing," in *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2001, pp. 586–588.

Dell'Olmo, P. , H. Kellerer, M. G. Speranza, and Z. Tuza, "A 13/12 approximation algorithm for bin packing with extendable bins," *Information Processing Letters*, **65**, 229–233 (1998).

Garey, M. R. and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, 1979.

Grötschel M., L. Lovász, and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization," *Combinatorica*, **1**(2), 169–197 (1981).

Grötschel, M., L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Algorithms and Combinatorics 2. Springer-Verlag, 1988.

Hochbaum, D. S. and D. B. Shmoys, "Using dual approximation algorithms for scheduling problems: Theoretical and practical results," *journal of the ACM*, **34**(1), 144–162 (1987).

Karmarkar, N. and R. M. Karp, "An efficient approximation scheme for the one-dimensional bin-packing problem," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, 1982, pp. 312–320.

Murgolo, F. D., "An efficient approximation scheme for variable-sized bin packing," *SIAM Journal on Computing*, **16**(1),149–161, (1987).

Woeginger, G. J., "When does a dynamic programming formulation guarantee the existence of an FPTAS," in *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 1999, pp. 820–829.