Approximation Schemes for Ordered Vector Packing Problems

Alberto Caprara * Hans Kellerer ** Ulrich Pferschy **

Abstract

In this paper we deal with the d-dimensional vector packing problem, which is a generalization of the classical bin packing problem in which each item has d distinct weights and each bin has d corresponding capacities. We address the case in which the vectors of weights associated with the items are totally ordered, i.e., given any two weight vectors a_i, a_j, either a_i is componentwise not smaller than a_j or a_j is componentwise not smaller than a_i, and construct an asymptotic polynomial-time approximation scheme for this case. As a corollary, we also obtain such a scheme for the bin packing problem with cardinality constraint, whose existence was an open question to the best of our knowledge.

We also extend the result to instances with constant Dilworth number, i.e. instances where the set of items can be partitioned into a constant number of totally ordered subsets. We use ideas from classical and recent approximation schemes for related problems, as well as a nontrivial procedure to round an LP solution associated with the packing of the small items.

1 Introduction

In the classical one dimensional *bin packing problem* (BP) we are given n items, the i-th item having a weight $a_i \in (0, 1]$ for $i = 1, \ldots, n$, and an infinite number of unit capacity bins. The goal is to assign each item to a bin so that the sum of the weights in each bin does not exceed the capacity and the total number of nonempty bins is minimized. The *d-dimensional vector packing problem* (VP) is the generalization of BP in which every item i instead of just one weight has d weights $a_i^1, a_i^2, \ldots, a_i^d$ and every bin has d corresponding capacities all equal to 1 (possibly after scaling). We let $a_i := (a_i^1, a_i^2, \ldots, a_i^d)$ denote the *weight vector* associated with item i. This generalization was first introduced in [4] and then studied by various authors, being both of practical and theoretical interest.

A relevant special case of VP arises when $d = 2$ and the item weights on the second dimension are identical, say $a_i^2 = 1/k$ for all i. In this case, we have a BP with an upper bound k on the *number of items* which can be put together into one bin, called the *k-item*

*DEIS, University of Bologna, viale Risorgimento 2, I-40136 Bologna, Italy, acaprara@deis.unibo.it

**University of Graz, Department of Statistics and Operations Research, Universitätsstr. 15, A-8010 Graz, Austria, {hans.kellerer, pferschy}@uni-graz.at

bin packing problem (kBP). This problem was first treated in 1975 by Krause, Shen and Schwetman [7, 8]. In their paper kBP appears as a formulation of a task–scheduling problem on a multiprogramming computer system, which is still an important part of every operating system for multiple processor machines 25 years later. Further details are given in [6] and [7].

Also the general 2-dimensional vector packing problem has many practical applications, some of which are mentioned in [9], [10] and [1], where heuristics and exact algorithms are proposed.

An illustrative example of an important application of 2-dimensional vector packing arises in the logistics of cargo airplanes. In a real-world problem from a major cargo airline we are given a set of n transportation requests for a certain very busy flight route, e.g. Frankfurt – New York. Each such request basically consists of a single freight piece or package. The obvious objective of the freight disponent is to find an assignment of the packages to planes such that all packages can be transported on a minimal number of planes. Assuming that all available planes have the same capacity is quite common, since many airlines try to use homogenuos fleets for certain routes. Moreover, each plane usually makes the round-trip several times such that a demand of e.g. 8 planes may be realized by using only 2 planes each making the trip four times.

For each package i we are given two parameters, its *weight* a_i^1 and its *volume* a_i^2. Although an optimal packing of the shapes of the items to be transported would be a three-dimensional packing problem, in practice the actual loading of the items assigned to each plane is done by experienced workers without mathematical optimization of the loading pattern. For practical purposes it is sufficient to ensure that the volume of each (usually rectangular) package does not exceed a given upper bound, which is naturally smaller than the actual volume available on the plane. The upper bound on the total weight in each plane is given by its pay load. Clearly, the resulting optimization problem is a special case of VP with $d = 2$. In practice, packages from the same customer share the main characteristics of weight and volume, in the sense that a heavier package also requires more volume.

Note that applications of VP arise not only in packing contexts but e.g. also for scheduling problems. It is well known that the classical problem of assigning n jobs with given processing times p_i to m identical machines, so that the total makespan is as small as possible, is closely connected to a corresponding bin packing problem. In particular, the decision version of the scheduling problem, i.e. the question whether all jobs can be performed within a given time limit C, is equivalent to the problem of packing n items with weights p_i into m bins of capacity C.

A major point of interest in scheduling is the treatment of resource constraints (see e.g. [4]) This means that the execution of a job requires not only machine capacity but also one or several additional resources which are available in limited amounts. In a very general model of resource constraints we can assume that each workplace consists of a machine and k other supplies of limited availability. Assigning a job to a workplace requires machine time p_i but also a given amount of each supply. The corresponding decision problem can be modeled as an instance of VP with $k+1$ dimensions, one for the machine time and k for the supplies.

A different and more standard model of resource constrained scheduling assumes that all processing times are equal to 1 and the k resources are separated from the machines. In this case, the analogy to bin packing is reached by considering all jobs which are executed in the same time period t as the content of bin t. Hence, the number of bins is given by the time limit C whereas the bin capacity for the items of size 1 is set to m. Every job i requires an amount r_{ij} of resource j for its execution. All jobs running in parallel, i.e. being packed into the same bin, must share the given resources. Hence, the sum of r_{ij} over all jobs i in the same bin must be bounded for every j by the available amount of this resource. Obviously, the resulting decision problem is a VP with $k+1$ dimensions.

For both of these models one can try to exploit the fact that in most practical applications the attributes of the given jobs are related. In particular, the set of jobs can be frequently partitioned into a small number of classes of jobs sharing the same characteristics, and for every class the consumption of resources (resp. supplies) are correlated with each other (resp. with the processing time). More precisely, considering e.g. the second model, given two jobs h, i in the same class, we have that $r_{ij} \geq r_{hj}$ for each resource j, i.e. job i requires a larger amount of *all* resources with respect to job h. (This is analogous to the cargo airplane application mentioned above, in which weights and volumes for the packages associated with the same customer are correlated.) The relevant special case of VP arising in these situations is the subject of the present paper.

We will first consider the so-called *ordered* VP. To this end, we define a natural order relation on the vectors in $\mathbb{R}^d$ and write $u \succeq v$ for any two vectors u, v in the same space if u is at least as large as v in *every component*. The ordered VP is the special case of VP in which, for any two items i, j either $a_i \succeq a_j$ or $a_j \succeq a_i$ holds, i.e. the relation "$\succeq$" defines a total ordering on the set of items. Note that kBP is a special example of the ordered 2-dimensional VP. Furthermore, we can introduce a *cardinality constraint* for VP analogous to kBP by requiring that at most k items may be packed into every bin. This can be modeled by extending VP to a $(d+1)$-dimensional vector packing problem where $a_i^{d+1} = 1/k$ for all i. Clearly, an ordered VP is still ordered also after adding this additional dimension. Hence, all results derived in this paper for the ordered VP also hold for the corresponding cardinality constrained problem.

Clearly, the item set N of every general VP can be partitioned into subsets $N^1, \ldots, N^c$ such that, for every subset N^j and every two items $i, h \in N^j$, either $a_i \succeq a_h$ or $a_h \succeq a_i$ holds, i.e. "$\succeq$" defines a total ordering on each subset N^j. The *Dilworth number* of a VP instance is the minimum c for which such a partitioning exists. Of course, an ordered VP instance has Dilworth number 1. The Dilworth number is equal to the "classical" Dilworth number for the partially ordered set given by the item weight vectors and the order relation "$\succeq$". Hence, the Dilworth number of a VP instance can be computed efficiently by max-flow techniques (see e.g. [5]). **We will also show in this paper how to extend our results to VP with *constant* Dilworth number. The discussion above points out that VP instances arising from real-world applications often have a relatively small Dilworth number.**

All the problems mentioned above are generalizations of BP and hence NP-hard in the strong sense. Moreover, it is well-known that no polynomial time approximation algorithm with a worst-case performance ratio better than 3/2 can exist for BP, unless P=NP. However, Fernandez de la Vega and Lueker [3] gave an *asymptotic polynomial time approximation scheme* (APTAS) for BP. Their method can easily be extended to an asymptotic $(d+\varepsilon)$-approximation algorithm for VP. Recently, Chekuri and Khanna [2] improved this result proposing a polynomial-time algorithm with worst-case performance ratio $1+\varepsilon d+O(\ln \varepsilon^{-1})$, for any $\varepsilon > 0$, which leads to a polynomial-time algorithm with worst-case performance ratio $O(\ln d)$ when the dimension d is fixed. The existence of an APTAS, even for the case $d = 2$, was recently ruled out by Woeginger [11] (on the assumption P$\neq$NP). For kBP, to the best of our knowledge, the best known asymptotic approximation ratio achievable is 3/2, as given in [6].

Our main result is showing the existence of an APTAS for the ordered VP in Section 2. As a corollary, we get an APTAS for kBP. Our scheme is based on standard techniques such as small items elimination and item grouping (cf. [3]), but also on the enumeration of the solutions for the instance obtained from grouping and on the use of an LP to include the small items for each solution. These latter techniques are similar to those used in [2] for multiprocessor vector scheduling (a problem related to VP in which the number of bins is fixed, all bins must have the same capacity b on all dimensions, and one would like to minimize the value of b so as to pack all items). Nevertheless, while the rounding of the LP solution is trivial for the case of the *constant* dimension, see [2], when the dimension is a part of the input rounding has to be done carefully to achieve an APTAS, as we will show in Section 2. In Section 3 we will show how our approach can be extended to VP with *constant* Dilworth number. An immediate corollary is an APTAS for the special case of a 2-dimensional VP where the number of different item weights in one dimension is bounded by a constant.

In the conclusions of [11] it is mentioned that “a slight modification of the method of Fernandez de la Vega and Lueker [3] yields asymptotic polynomial time approximation schemes for the subproblems of d-dimensional vector packing with constant Dilworth number”. This would cover the results presented in this paper (at least for a constant dimension – it is not clear whether the above sentence refers to a constant dimension or not). Nevertheless, to the best of our understanding, in order to achieve our results a *substantial* modification of the method in [3] is necessary, along the lines presented in this paper. In particular, even if a near-optimal solution for the *large* items (see the next section) in an ordered VP instance is easy to achieve following the approach in [3], it is not always possible to extend this solution to a near-optimal one for the overall instance, including the *small* items. (Certainly a greedy approach does not work, as it may end up with bins almost filled in different dimensions.) In fact, the packing of the large items can hardly be done independently of the small items, as it is done in [3] and other similar approaches, to achieve near-optimality.

Consider for example the kBP instance with $k = 5$ and $n = 2s + 8s$ items (where s is a positive integer), $2s$ with weight $1/2$ and $8s$ with weight $1/8$. Clearly, the optimal solution packs one item of weight $1/2$ and four items of weight $1/8$ per bin, requiring a total of $2s$ bins. If the small items are those with weight $1/8$, the optimal packing of the large items, requiring s bins, is obtained by packing two items of weight $1/2$ in each bin. Then, the remaining small items must be packed into separate bins, requiring $\left\lceil \frac{8s}{5} \right\rceil$ additional bins. If we replace 8 by 2^p for some integer $p \geq 3$ and let $k = 2^{p-1} + 1$, assuming the small items are those with weight $1/2^p$, the optimal solution still needs $2s$ bins, whereas the solution obtained by optimally packing the large items first uses $s + \left\lceil \frac{2^p \; s}{2^{p-1}+1} \right\rceil$ bins, which goes to $3s$ as p goes to infinity. Accordingly, the worst-case approximation guarantee cannot be better than $3/2$ (even for kBP) if the large items are packed first.

Although we have already used the notion of approximation algorithm throughout the Introduction, we will conclude this section with a precise definition of approximation algorithms used in the paper. These algorithms are generally classified by their *worst-case ratio*: For any VP algorithm A let $C^A(I)$ denote the number of bins used by algorithm A, and let $C^{OPT}(I)$ denote the minimum (optimum) number of bins required to pack the items of a given instance I. Then the *asymptotic* worst-case performance ratio is defined as

$$R_A = \lim_{C^{OPT}(I)\to\infty} \sup_I \frac{C^A(I)}{C^{OPT}(I)}\,.$$

Observe that $R_A \le K_1$ if there are two constants K_1 and K_2 such that for every problem instance I

$$C^A(I) \le K_1 C^{OPT}(I) + K_2\,.$$

Moreover, R_A is an *absolute* worst-case ratio if $K_2 = 0$. Consider a value $\varepsilon \in (0,1)$. We say that A is an *asymptotic* $(1+\varepsilon)$*-approximation algorithm* if $R_A \le (1+\varepsilon)$. An APTAS is an asymptotic $(1+\varepsilon)$-approximation algorithm for any $\varepsilon > 0$ with running time polynomial in the size of the encoded instance.

2 An APTAS for the Ordered VP

This main section contains the description of an APTAS for the ordered VP both for the fixed dimension and for the case where d is a part of the input. Let ε be the required accuracy. We will assume $\varepsilon < 1/2$. As it is usually the case for packing problems we will distinguish between *small* and *large items.* Let us define for the given value $\varepsilon \in (0, 1/2)$

$$\begin{aligned} L &:= \{i \mid a_i^k \ge \varepsilon \text{ for some } k \in \{1,\dots,d\}\}, & \ell &:= |L|, & &\textit{large items} \\ S &:= \{i \mid a_i^k < \varepsilon \text{ for all } k \in \{1,\dots,d\}\}, & s &:= |S|, & &\textit{small items.} \end{aligned}$$

For simplicity we assume that $L = \{1,\dots,\ell\}$ with $a_1 \succeq a_2 \succeq \ldots \succeq a_\ell$.

2.1 Enumerating packings for the large items

Let us first assume that $\ell > 2/\varepsilon^2$. Later on we will also illustrate how to handle the (simpler) case $\ell \le 2/\varepsilon^2$. We introduce *item grouping* to attain a simplified structure of packings for the large items, so that only a polynomial number of different packings have to be considered. In particular, we group the items in L as follows. Define

$$h := \left\lfloor \ell\varepsilon^2 \right\rfloor$$

and let p and q be such that $\ell = ph + q$ and $1 \le q \le h$. Note that $h \ge 2$. We now show that

$$p \le \frac{2}{\varepsilon^2} \qquad (1)$$

and hence p is bounded by a constant. If (1) were not true we would have the contradiction

$$\ell = ph + q > \frac{2}{\varepsilon^2}\left\lfloor \ell\varepsilon^2 \right\rfloor \geq \frac{2}{\varepsilon^2}(\ell\varepsilon^2 - 1) = \ell + \left(\ell - \frac{2}{\varepsilon^2}\right) > \ell,$$

because $\ell > 2/\varepsilon^2$.

We partition L into the $p+1$ subsets $L_i := \{ih+1, \ldots, (i+1)h\}$, $i = 0, \ldots, p-1$, each containing h items and $L_p := \{ph+1, \ldots, \ell\}$. Note that each bin in a feasible solution contains at most $\left\lfloor \frac{1}{\varepsilon} \right\rfloor$ items from L, since a_ℓ has a weight of at least ε in at least one dimension and all other items are not smaller in this dimension. Accordingly, we let a *packing vector* t be a $(p+1)$-dimensional vector with $t_i \leq \left\lfloor \frac{1}{\varepsilon} \right\rfloor$ for $i = 0, \ldots, p$, where t_i denotes the number of items from L_i. We observe that, while all sets of large items that fit in a bin correspond to a packing vector, the converse does not hold.

The total number f of all possible packing vectors t with $t_i \leq \left\lfloor \frac{1}{\varepsilon} \right\rfloor$ for all i is clearly bounded by

$$f \leq \left(\left\lfloor \frac{1}{\varepsilon} \right\rfloor + 1\right)^{p+1}$$

since each of the $p+1$ components of t can be chosen between 0 and $\left\lfloor \frac{1}{\varepsilon} \right\rfloor$. Hence, f is constant for any fixed ε.

In our approximation scheme we compute all possible combinations of packing vectors to the bins. To this end, we need to know the number $m = C^{OPT}(I)$ of bins used by the optimal solution. This can be done either by trying all the possible values of m, between 1 and n, or by performing binary search. For the given m, we consider all possible *assignments* of packing vectors to the m bins. To get a very rough upper bound on the number of such assignments, consider that each of the f packing vectors can be selected up to m times. Hence, we can count for every packing vector the number of times it is assigned to a bin. Generating f such numbers between 0 and m yields a total of $(m+1)^f$ possible assignments which is polynomial in m (and n). Note that we do not have to identify the particular bin a vector is assigned to.

An assignment is called *feasible* if the sum over the ith components of all m packing vectors is equal to $|L_i|$ for $i = 0, 1, \ldots, p$. Note that in this way we will also consider the *optimal* assignment, i.e. the assignment that associates with each bin exactly the packing vector defined by this bin in the optimal solution.

We can define for every packing vector t an *induced packing* of items in L into a bin (not necessarily feasible) by packing *up to* t_i arbitrary items from set L_{i+1} for $i < p$. The entry t_p of the packing vector is ignored in the induced packing. If the packing vector t corresponds to a feasible packing then also the packing induced by t is feasible, since every item $k \in L_i$ in the feasible packing is replaced in the induced packing either by no item or by an arbitrary item $j \in L_{i+1}$ such that $a_j \preceq a_k$. Note that a feasible assignment may involve a packing vector corresponding to an infeasible bin. However, the crucial point that guarantees that our approach is correct is that a packing vector corresponding to a feasible bin gives rise to an induced packing which is always feasible.

For every feasible assignment considered, we generate the corresponding induced packing for all bins. Recall that this means that for every bin with an assigned packing vector t we pack up to t_i arbitrary items from L_{i+1} for $i = 0, \ldots, p-1$. Since the assignment is feasible, it is easy to generate the induced packing so that all items in $L_1, L_2, \ldots, L_p$ are packed, noting that $|L_i| = |L_{i+1}|$ for $i = 0, \ldots, p-2$ and $|L_{p-1}| \geq |L_p|$. If in the thereby generated packing the capacity of some bin is exceeded, the feasible assignment we started from cannot be the optimal one, and we simply discard it. Otherwise, i.e. if all the m d-dimensional capacity constraints are fulfilled, let $b_j = (b_j^1, \ldots, b_j^d)$ be the residual capacity vector of bin j for $j = 1, \ldots, m$. By construction of the packing induced by the packing vectors of the optimal assignment, the values b_j of this packing are componentwise at least as large as the residual capacities in the optimal solution for each bin j.

The items in L_0 are left unpacked and are put each into a separate bin, which requires h extra bins. A trivial bound on the total weight of large items yields $m \geq \lceil \ell\varepsilon \rceil$ since, as mentioned before, in at least one dimension item a_ℓ and also all other large items have a weight at least ε. Hence, we get

$$h = \left\lfloor \ell\varepsilon^2 \right\rfloor \leq \lceil \ell\varepsilon \rceil \varepsilon \leq m\varepsilon.$$

Summarizing, since one of all the assignments derived from the considered assignments of packing vectors to bins corresponds to the optimal packing, we managed to pack all large items into $m + m\varepsilon$ bins while leaving in every bin a residual capacity for the small items at least as large as in the optimal solution.

We now consider the case $\ell \leq 2/\varepsilon^2$. In this case, the number of large items is constant and, assuming we know the number m of bins in an optimal solution, we enumerate all the $O(m^\ell)$ feasible packings of the large items into the bins, i.e. packings fulfilling the capacity constraints in every dimension, among which is the optimal assignment.

Summarizing, we have proved

Theorem 1 *For any fixed $\varepsilon \in (0, 1/2)$, there is a $(1+\varepsilon)$-approximation algorithm for the ordered VP if all items are large with respect to ε.* □

2.2 Packing the small items

For each feasible packing of large items generated above in either case, we pack the small items. First we solve the following LP.

For each small item $i \in S$ and bin $j = 1, \ldots, m$ let $x_{ij} \in [0, 1]$ denote the (fractional) part of item i which is packed into bin j.

$$\sum_{j=1}^{m} x_{ij} = 1, \quad i \in S, \tag{2}$$

$$\sum_{i \in S} a_i^k x_{ij} \leq b_j^k, \quad j = 1, \ldots, m, k = 1, \ldots, d, \tag{3}$$

$$x_{ij} \geq 0, \quad i \in S,\ j = 1, \ldots, m. \tag{4}$$

Note that we are only interested in a feasible solution to this set of linear inequalities. To formally define an LP any objective function may be added.

If the LP has no feasible solution, the feasible assignment cannot correspond to the optimal assignment by the above discussion, and this assignment is discarded. Otherwise, a *basic* feasible solution of the LP contains at most $s + dm$ nonzero variables, and a straightforward counting argument (also given in [2]) shows that the number of items that are not packed into a single bin is at most dm. These items can be put into separate bins. Recalling that $a_i^k < \varepsilon$ for each $i \in S$ and $k = 1, \ldots, d$, this requires no more than $\left\lceil \frac{dm}{\lfloor \frac{1}{\varepsilon} \rfloor} \right\rceil$ bins, which can be bounded from above with a simple calculation by $2dm\varepsilon + 1$ for $\varepsilon < 1/2$.

Accordingly, the solution found for the correct value of m and the optimal assignment has a value bounded by

$$m + m\varepsilon + \left\lceil \frac{dm}{\lfloor \frac{1}{\varepsilon} \rfloor} \right\rceil \leq m + (2d+1)m\varepsilon + 1,$$

which proves

Theorem 2 *There is an APTAS for the ordered VP if the dimension d is constant.* □

Corollary 1 *There is an APTAS for kBP.* □

2.3 Rounding the LP solution for the general dimension

The case in which d is a part of the input needs a more careful approach. Let x^* be an optimal solution of the above LP. Our final aim is to round this solution in a way such that only $2m$ items remain unpacked in bins $1, \ldots, m$. By the discussion above, packing these items in additional bins in a greedy way will require no more than $4m\varepsilon + 1$ additional bins, yielding an APTAS also for the case of the general dimension d.

To simplify the notation, we will assume that no small item is entirely packed into one bin, i.e. that no entry of x^* is equal to one. The generalization to the case in which $x_{ij}^* = 1$ for some i, j is obvious by fixing item i into bin j and considering only the packing of the fractional items.

Let $S = \{1, \ldots, s\}$ with $a_1 \preceq a_2 \preceq \ldots \preceq a_s$. We will pack the items in S in increasing order into the m given bins until we "get stuck". Lemma 3 will state that at this point there are at most $2m$ small items left unpacked. In principle, in order to pack some item i we will try to pick a bin (which possibly already contains a fractional part of i) and pack item i completely into that bin. To make room for item i we will "throw out" the fractional parts of larger items $i+1, i+2, \ldots$ as far as necessary.

More formally, we will derive step by step a rounded solution $\tilde{x}$ from the original LP solution x^*. Initially, we set $\tilde{x} := x^*$. We pack the smallest item 1 as follows. If there are bins j such that $\sum_{i \in S} \tilde{x}_{ij} = \sum_{i \in S} x_{ij}^* < 1$, we set $\tilde{x}_{ij} := 0$ for $i \in S$, i.e. we do not consider these bins to pack any item because it may happen that no item fits completely into such a bin. Then we find an (arbitrarily chosen) bin j such that $\tilde{x}_{1j} > 0$, letting $f_1 := 1$. If no such bin exists (which is possible due to the fact that some bins are not considered) we choose a bin j such that $\tilde{x}_{f_1 j} > 0$ and f_1 is as small as possible. (If all entries in $\tilde{x}$ are 0, item 1 as well as all other small items are unpacked, see below.) We pack item 1 in bin j, using the capacity of bin j allocated by the LP solution for items $f_1, \ldots, e_1$, where e_1 is the smallest index such that $y := \sum_{h=f_1}^{e_1} \tilde{x}_{hj} \geq 1$. In our rounded solution, this corresponds to setting $\tilde{x}_{1j} := 1$, $\tilde{x}_{1\ell} := 0$ for $\ell \neq j$, $\tilde{x}_{hj} := 0$ for $1 < h < e_1$, and $\tilde{x}_{e_1 j} := y - 1$. We say that the packing of item 1 *starts at level* f_1 and *ends at level* e_1. Note that, if $y > 1$, not all capacity allocated for item e_1 is used for item 1; in other words we still have a fraction $\tilde{x}_{e_1 j}$ that may be used to pack into bin j items $2, 3, \ldots, e_1$. To be more precise, we formally state our rounding procedure in Figure 1.

Procedure LP-round

For each item $i = 1, \ldots, s$ perform the following steps:

Bin closing:

For each bin j such that $\sum_{h \geq i} \tilde{x}_{hj} < 1$, let $\tilde{x}_{hj} := 0$ for $h \geq i$
(close bin j)
If all bins are closed then terminate

Bin selection:

If $\tilde{x}_{ij} > 0$ for some bin j then
Let $\tilde{x}_{i\ell} := 0$ for $\ell \neq j$ and $f_i := i$
Else
Let $f_i := \min\{r : r > i \text{ and } \tilde{x}_{rk} > 0 \text{ for some bin } k\}$
Let j be a bin such that $\tilde{x}_{f_i j} > 0$

Solution updating:

Let $\tilde{x}_{ij} := 1$ (pack item i in bin j)
Let $e_i := \min\{r : \sum_{h=i}^{r} \tilde{x}_{hj} \geq 1\}$ and $y := \sum_{h=i}^{e_i} \tilde{x}_{hj}$
Let $\tilde{x}_{hj} := 0$ for $h = 1, \ldots, e_i - 1$ and $\tilde{x}_{e_i j} := y - 1$

Figure 1: The LP rounding procedure.

If after the bin closing step no bin is left for item i, the remaining items $i, \ldots, s$ will be packed into separate bins as above.

The procedure is illustrated in Figure 2 on an example, showing the rounded solution $\tilde{x}$ at the beginning as well as after each solution updating step. Among 6 items, items $1, 2, 3, 4$ are packed, respectively, into bins $1, 2, 3, 3$, with $f_1 = 1, e_1 = 4; f_2 = 2, e_2 = 6; f_3 = 3, e_3 = 4; f_4 = 5, e_4 = 6$.

Lemma 3 *The number of small items not packed into the first m bins at the end of the rounding procedure is at most $2m$.*

Proof. The number of small items packed by the rounded solution into bins $1, \ldots, m$, is equal to $\sigma := \sum_{i \in S} \sum_{j=1}^{m} \tilde{x}_{ij}$. We will compare this value to $s = \sum_{i \in S} \sum_{j=1}^{m} x^*_{ij}$. Note that σ is initially equal to s and may be decreased only in the bin closing step and in the bin selection step, whereas it is unchanged in the solution updating step. It is immediate to see that the overall decrease in all bin closing steps is at most m. In the following we will bound the overall decrease in all bin selection steps.

Noting that there may be some decrease in the bin selection step for item i only if $f_i = i$, let k be the last such that $f_k = k$, i.e. $f_i > i$ for $i > k$. If $k \leq m$, then the overall decrease in all bin selection steps is not larger than m, as the decrease in each step is not larger than 1.

$$\tilde{x} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/8 & 3/8 & 1/2 \\ 1/8 & 1/8 & 3/4 \\ 1/2 & 1/4 & 1/4 \\ 1/4 & 0 & 3/4 \\ 0 & 1/2 & 1/2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 3/8 & 1/2 \\ 0 & 1/8 & 3/4 \\ 1/4 & 1/4 & 1/4 \\ 1/4 & 0 & 3/4 \\ 0 & 1/2 & 1/2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3/4 \\ 0 & 0 & 1/4 \\ 0 & 0 & 3/4 \\ 0 & 1/4 & 1/2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 3/4 \\ 0 & 0 & 1/2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Figure 2: Illustration of the rounding procedure.

We now consider the case $k > m$. The definition of f ensures $f_1 \leq f_2 \leq \ldots \leq f_k$. Hence, since $f_k = k$, the packing of all items in $1, \ldots, k$ starts at a level not larger than k, i.e. $f_i \leq k$ for $i = 1, \ldots, k$. Clearly, the number of items i with $f_i \leq k$ whose packing ends at a **level larger than** k**,** i.e. $e_i > k$, cannot be larger than m, the number of bins available. This means that $e_i \leq k$ for at least $k - m$ items among $1, \ldots, k$. Now, since the decrease of σ during the bin selection steps in the first k iterations is only due to changing entries $\tilde{x}_{ij}$ with $i \leq k$, this decrease is at most m since $\sum_{i=1}^{k} \sum_{j=1}^{m} x_{ij}^* = k$, and $\sum_{i=1}^{k} \sum_{j=1}^{m} \tilde{x}_{ij} \geq k - m$ (at least $k - m$ items among $1, \ldots, k$ are packed by the rounding procedure). After packing k no further decrease takes place in the bin selection step, so the total decrease in this step is at most m.

In both cases, we have an overall decrease not larger than $2m$, i.e. the number of small items unpacked at the end of the rounding step does not exceed $2m$. □

For further illustration of this rounding procedure another example is given. Consider the case in which all items are small, $s = p \cdot m$ and all items in S have weight $(1/p, \ldots, 1/p)$. In a fractional solution x^* we may have $x_{ij}^* = 1/m$ for $i = 1, \ldots, s$ and $j = 1, \ldots, m$. Our rounding procedure packs item 1 in bin 1 ($f_1 = 1$ and $e_1 = m$), with a decrease in the bin selection step of $\sum_{j \neq 1} \tilde{x}_{1j} = 1 - 1/m$, item 2 in bin 2 ($f_2 = 2$ and $e_2 = m + 1$), with a decrease of $\sum_{j \neq 2} \tilde{x}_{2j} = 1 - 2/m$, (note that $\tilde{x}_{21}$ is set to 0 in the solution updating step for item 1), ..., item $m - 1$ in bin $m - 1$ ($f_{m-1} = m - 1$ and $e_{m-1} = 2m - 2$), with a decrease of $1/m$. Accordingly, the overall decrease in the first $m - 1$ bin selection steps is $\frac{m-1}{m} + \frac{m-2}{m} + \ldots + \frac{1}{m} = \frac{m-1}{2}$. Afterwards, the procedure continues, packing item i in bin $i \bmod m$, for $m \leq i \leq s - m + 1 = (p - 1)m + 1$, wihtout any futher decrease in the bin selection step. Then, after having packed $(p - 1)m + 1$ items, all bins are closed and the decrease in the bin closing step is $\frac{1}{m} + \frac{2}{m} + \ldots + \frac{m-1}{m} = \frac{m-1}{2}$.

The discussion above shows

Theorem 4 *There is an APTAS for the ordered VP.* □

As already mentioned, adding a cardinality constraint to an ordered VP instance yields another ordered VP instance. This shows

Corollary 2 *There is an APTAS for the k-item ordered VP.* □

We outline our APTAS in Figure 3. It should be noted that efficiency was not an objective in the design of the algorithm. Its aim is rather to illustrate the existence of an APTAS for the ordered VP.

In particular, the practical inefficiency is due to the complete enumeration of the feasible assignments of large items to the bins. Although the inefficiency in handling large items is common to all approximation schemes for packing problems, in our scheme the number of feasible assignments is (roughly) $n^{\frac{1}{\varepsilon}^{\frac{1}{\varepsilon}^{2}}}$, for each of them we solve an LP, etc., whereas most of the schemes for easier packing problems have linear running time for fixed ε.

Although this makes the practical application of our approximation scheme very unlikely, some of the ideas may be useful within the design of practical heurstics. For instance, the separation between "large" and "small" items, packing the large ones first (by some reasonable heuristic instead of almost complete enumeration) and then the small ones (possibly by LP and rounding) may turn out to be effective in practice.

Algorithm APTAS

Partition the item set into S and L
For all $m = 1, \ldots, n$ (possible number of bins in the optimal solution)
  If $\ell > 2/\varepsilon^2$ then
    Partition L into $L_0, \ldots, L_{p-1}$ (each containing $\lfloor |L|\varepsilon^2 \rfloor$ items) and L_p
      (containing $|L| - p\lfloor |L|\varepsilon^2 \rfloor$ items)
    Pack the items in L_0 into $|L_0|$ bins
    For each feasible assignment of packing vectors to the m bins
      Pack the items in $L_0, \ldots, L_{p-1}$ into the m bins
        as induced by the packing vectors
      If the packing generated is feasible then
        Pack the items in L_p into $|L_p|$ extra bins
        Solve the LP corresponding to the packing of the items
          in S into the m bins
        If the LP has a feasible solution then
          Pack the items in S according to the LP solution,
            using extra bins for the unpacked small items
          Possibly update the best solution found so far
  Else ($\ell \leq 2/\varepsilon^2$)
    For each feasible packing of the large items to the m bins
      Solve the LP corresponding to the packing of the items in S
        into the m bins
      If the LP has a feasible solution then
        Pack the items in S according to the LP solution,
          using extra bins for the unpacked small items
        Possibly update the best solution found so far

Figure 3: Outline of the APTAS for the ordered VP.

3 Extension to Bounded Dilworth Number

Let $N^1, \ldots, N^c$ be the partitioning of the item set N into the minimum number of ordered sets as described in the Introduction. Throughout the section, we will consider the case in which c is a constant.

The APTAS for this problem is derived along the same lines as the approximation scheme in Section 2. We treat the items in each set N^j in the same way as we treated the whole item set N in the previous section. Namely, we consider the set L^j of the large items in N^j, i.e. items with $a_i^k \geq \varepsilon$ for some $k \in \{1, \ldots, d\}$. Let $\ell^j := |L^j|$. If $\ell^j \leq 2/\varepsilon^2$, there is only a constant number of large weight items in L^j. Otherwise, we perform item grouping again and partition L^j into a constant number $p^j + 1$ of subsets containing (again except the last one) $\lfloor \ell^j \varepsilon^2 \rfloor$ consecutive items. The small items from all sets N^j will be considered later.

Now we consider all packing vectors corresponding to the packing of large items into bins. Since the number of sets L^j is at most c and each of them is either partitioned into a constant number $p^j + 1$ of subsets with consecutive items, or contains only a constant number of items, a packing vector has constant length. More precisely, now a packing vector has $p^j + 1$ entries for each set N^j which is partitioned into $p^j + 1$ subsets, each telling the number of large items packed from each subset. Moreover, a packing vector has ℓ^j entries for each set N^j with a constant number of large items, and has value 0 or 1 indicating whether the corresponding item is packed or not. Obviously, every entry of a packing vector is again at most $\left\lfloor \frac{1}{\varepsilon} \right\rfloor$. Therefore, the overall number of packing vectors is again bounded by a constant and the number of possible assignments of packing vectors to the m bins is bounded by a polynomial in m.

The induced packing can be constructed from each generated packing vector in the same way as described in Section 2.1 (of course only for the components corresponding to sets N^j with $\ell^j > 2/\varepsilon^2$). As before we consider only feasible assignments of packing vectors to bins which leaves again at most $\lfloor \ell^j \varepsilon^2 \rfloor$ items unpacked for every set N^j. Packing all these items into separate bins requires at most $cm\varepsilon$ extra bins by employing a trivial weight bound separately on each set N^j.

There remain the small items to be packed for every generated feasible packing. This is done again by the same LP as in the previous section. The fractional solution values are rounded separately for each set N^j. By the argument of Section 2.3 we now have at most $2cm$ items which are not packed into the first m bins, and we can pack any $\left\lfloor \frac{1}{\varepsilon} \right\rfloor$ of them together in one bin thus requiring as before at most $\left\lceil \frac{2cm}{\lfloor \frac{1}{\varepsilon} \rfloor} \right\rceil$ bins. Overall, the number of bins used is at most $m + cm\varepsilon + 4cm\varepsilon + 1$, which shows the following

Theorem 5 *There is an APTAS for VP with constant Dilworth number.* □

Acknowledgments:
The work of the first author was partially supported by CNR and MURST, Italy.

References

[1] A. Caprara, P. Toth, "Lower Bounds and Algorithms for the 2-Dimensional Vector Packing Problem", to appear in *Discrete Applied Mathematics* (2001).

[2] C. Chekuri, S. Khanna, "On Multi-dimensional Packing Problems", in *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'99)*, ACM Press (1999).

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

[4] M.R. Garey, R.L. Graham, D.S. Johnson, A.C. Yao, "Resource constrained scheduling as generalized bin packing", *J. Combinatorial Theory Ser. A* **21** (1976), 257–298.

[5] M. Grötschel, L. Lovász, A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, 1988.

[6] H. Kellerer, U. Pferschy, "Cardinality Constrained Bin-Packing Problems", *Annals of Operations Research* **92** (1999), 335–348.

[7] K.L. Krause, V.Y. Shen, H.D. Schwetman, "Analysis of several task-scheduling algorithms for a model of multiprogramming computer systems", *Journal of the ACM* **22** (1975), 522–550.

[8] K.L. Krause, V.Y. Shen, H.D. Schwetman, "Analysis of several task-scheduling algorithms for a model of multiprogramming computer systems, Errata", *Journal of the ACM* **24** (1977), 527.

[9] K. Maruyama, S.K. Chang, D.T. Tang, "A General Packing Algorithm for Multidimensional Resource Requirements", *International Journal Comput. Inform. Sci.* **6** (1977), 131–149.

[10] F.C.R. Spieksma, "A Branch-and-Bound Algorithm for the Two-Dimensional Vector Packing Problem", *Computers and Operations Research* **21** (1994), 19–25.

[11] G.J. Woeginger, "There is no Asymptotic PTAS for Two-Dimensional Vector Packing", *Information Processing Letters* **64** (1997), 293–297.