On-line Variable Sized Covering

Leah Epstein[*]
The Interdisciplinary Center, Herzliya, Israel

Abstract

We consider one-dimensional and multi-dimensional vector covering with variable sized bins.

In the one-dimensional case, we consider variable sized bin covering with bounded item sizes. For every finite set of bins B, and upper bound $1/m$ on the size of items for some integer m, we define a ratio $r(B,m)$. We prove this is the best possible competitive ratio for the set of bins B and the parameter m, by giving both an algorithm with competitive ratio $r(B,m)$, and an upper bound of $r(B,m)$ on the competitive ratio of any on-line deterministic or randomized algorithm. The ratio satisfies $r(B,m) \geq m/(m+1)$, and equals this number if all bins are of size 1.

For multi-dimensional vector covering we consider the case where each bin is a binary d-dimensional vector. It was shown in [1] that if B contains a single bin which is all 1, then the best competitive ratio is $\Theta(1/d)$. We show an upper bound of $1/2^{d(1-o(1))}$ for the general problem, and consider four special case variants. We show an algorithm with optimal competitive ratio $1/2$ for the model where each bin in B is a standard basis vector. We consider the model where B consists of all unit prefix vectors. A unit prefix vector has i leftmost components of 1, and all other components are 0. We show that this model is harder than the case of standard basis vector bins, by giving an upper bound of $O(1/\log d)$ on the competitive ratio of any deterministic or randomized algorithm. Next, we discuss the model where B contains all binary vectors. We show this model is easier than the model of one bin type which is all 1, by giving an algorithm with competitive ratio $\Omega(1/\log d)$.

The most interesting multi-dimensional case is $d = 2$. The results of [1] give a 0.25-competitive algorithm for $B = \{(1,1)\}$, and an upper bound of 0.4 on the competitive ratio of any algorithm. In this paper we consider *all* other models for $d = 2$. For standard basis vectors, we give an algorithm with optimal competitive ratio $1/2$. For unit prefix vectors we give an upper bound of $4/9$ on the competitive ratio of any deterministic or randomized algorithm. For the model where B consists of all binary vectors, we design an algorithm with ratio larger than 0.4. These results show that all above relations between models hold for $d = 2$ as well.

1 Introduction

We consider the on-line problem of covering variable sized bins. The one-dimensional version of this problem is defined as follows: We are given a finite set B of allowed bin sizes, each

[*]School of Computer and Media Sciences, The Interdisciplinary Center, P.O.B. 167, 46150 Herzliya, Israel. E-Mail: epstein.leah@idc.ac.il

$b \in B$ satisfies $0 < b \leq 1$ and the largest element of B is 1. Items with size in $(0, 1]$ arrive on-line, each item is to be assigned to a bin upon arrival. The algorithm may assign a new item to a previously used bin, or open a new bin of any size in B. A bin of size b is *covered* if the total size of items assigned to it is at least b. The goal of an algorithm is to maximize the sum of the sizes of all covered bins. This sum is the value of an algorithm. In this paper we consider bin covering with bounded item sizes. The size of each arriving item is at most $1/m$ where m is a fixed integer.

The multi-dimensional version is defined as follows: For a dimension d we are given a set of d-dimensional bins B, which is a subset of $\{0, 1\}^d$, i.e., all components of the allowed bins are binary. The "vector weight" of a d-dimensional vector b is the sum of its components, this applies both to multi-dimensional bins ("bin weights") and to multi-dimensional items ("item weights"). In the one-dimensional case the "bin weight" of a bin is simply its size and the "item weight" of an item is the size of the item. Vector items with size in $[0, 1]^d$ arrive on-line, and as in the one-dimensional case, each new item is to be assigned to a new or previously used bin upon arrival. A bin of size b is covered if the vector x, which is the sum of all items assigned to the bin, satisfies $x \geq b$. The goal of an algorithm is to maximize the sum of the bin weights of all covered bins. This number is the value of the algorithm. We study the general problem of arbitrary sets B, and also consider three interesting cases of multi-dimensional variable sized vector covering. The first case is where B consists of all possible binary vectors. The second case is when all bins are standard basis vectors (have only one non-zero coordinate). In the last case we consider a unit prefix sequence of bins, each bin has i leftmost 1 coordinates, and all others are 0.

Applications: Variable sized bin and vector covering relate to a case where a large number of jobs of a finite type are to be done (these are the bins). Each job is done by one worker or by a combined work of more than one worker (the workers are the items). An item represents the amounts of work a worker can do in a day. Each worker is assigned to one job (bin). The benefit of a job is the minimum amount of work to do it, which is proportional to its size. This scenario relates to one dimensional bin covering.

The scenario of vector covering is equivalent to the case where a job has d different qualities. We can normalize and assume that each quality which exists in the job has size 1, each type of job has to have a subset of the qualities, the different types of jobs are again represented by bins, which are d-dimensional binary vectors. The work of each worker is also represented by a d-dimensional vector which corresponds to the parts of each quality that the worker can do in one day.

Definition of competitive ratio: We measure the performance of an on-line algorithm by the competitive ratio. Denote the value of the on-line algorithm on a given sequence by V_{on}, and the value of an optimal off-line algorithm, that knows the sequence in advance, by V_{opt}. The competitive ratio is the supremum r ($r \leq 1$) for which $V_{on} \geq r \cdot V_{opt} - \eta$ is satisfied for every sequence (η is a constant which does not depend on the input). For randomized algorithms we replace V_{on} by $E(V_{on})$ and in this case the competitive ratio is the supremum r for which $E(V_{on}) \geq rV_{opt} - \eta$ is satisfied for every sequence. An on-line algorithm is called optimal if it has a competitive ratio which is the best possible competitive ratio. Note that in variable sized covering, the competitive ratio can be constant, or a function of B. Since this is a definition of competitive ratio for maximization algorithms, and $r \leq 1$, the goal is

to design algorithms with large competitive ratio, as closer to 1 as possible.

Known results: The *one-dimensional* version of the vector covering problem (for the case that all bins are of size 1) was first investigated in the thesis [2] of Assmann and in the journal article by Assmann, Johnson, Kleitman and Leung [3]. There it is proved that the greedy algorithm (that simply keeps putting items into the same bin until it is covered and then moves on to the next bin) has a worst case guarantee of 1/2. Moreover, two more sophisticated algorithms were derived with worst case ratios $\frac{2}{3}$ and $\frac{3}{4}$, respectively. Both of these sophisticated algorithms are based on presorting the items, and consequently are off-line algorithms. The greedy algorithm, however, is an on-line algorithm. Csirik and Totik [8] proved that in fact the greedy algorithm is a *best possible* on-line algorithm, since no on-line algorithm can have a worst case ratio that is strictly better than 1/2. Csirik, Frenk, Galambos and Rinnooy Kan [7] gave a probabilistic analysis of the one-dimensional bin covering and of the two-dimensional vector covering problem (for the case that all bins are of size $(1, 1)$). Gaizer [10] constructed an off-line approximation algorithm with worst case guarantee 1/2 for dimension $d = 2$. The article by Csirik and Frenk [6] summarizes all results on vector covering problems that were derived till 1990. The multi-dimensional vector covering problem was studied by Alon, Azar, Csirik, Epstein, Sevastianov, Vestjens and Woeginger in [1]. The paper considers the case where the only allowed bin is the "all 1" vector. This paper considers both on-line and off-line vector covering. [1] gives an on-line algorithm with competitive ratio $1/(2d)$ and an upper bound of $1/(d+0.5)$ on the ratio of any deterministic or randomized algorithm. Finally Woeginger and Zhang [11] studied variable sized one-dimensional vector covering. They define a ratio $r(B)$. Consider only bins of size 1/2 or more, sort them in non-decreasing order. Then $r(B)$ is the minimum of the following two values:
1. The minimum ratio of two consecutive bin sizes (this number is smaller than 1).
2. The inverse of twice the size of the minimum sized bin (among the bins which are larger or equal to 1/2).
They show this ratio $r(B)$ is optimal for a finite set of bins B. A survey on on-line packing and covering is given in [9].

Our results: We give some definitions in order to define the number $r(B, m)$. Let $B = \{b_1, \ldots, b_s\}$ be the set of allowed bin sizes, where $1 = b_1 > b_2 > \ldots > b_s$, and let m be an integer $m \geq 1$ such that $1/m$ is an upper bound on item sizes. For each $1 \leq i \leq s$, let $b_{i,j} = b_i/j$ and let $B_i(m)$ be the set of fractions of b_i between sizes $1/(2m)$ and $1/m$, that is $B_i(m) = \{b_{i,j} | 1 \leq j \leq 2m\} \cap [1/(2m), 1/m]$. We define a new set of bins by $C(m) = \cup_{1 \leq i \leq s} B_i(m)$. Enumerate the sizes of numbers in $C(m)$, $C(m) = \{c_1, \ldots, c_k\}$ where $1/m = c_1 > c_2 > \ldots > c_k = 1/(2m)$. For every element in $C(m)$, recall an original bin size that caused it to be inserted into $C(m)$. For c_i, let $b(c_i)$ be the smallest b_j so that there exists an integer y that satisfies $yc_i = b_j$. Now define $q(B, m) = \max\{c_i/c_{i+1} | 1 \leq i \leq k-1\}$. Note that $1 + 1/m \geq q(B, m) > 1$. Finally define $r(B, m) = 1/q(B, m)$.

We show the following results for bin covering.

- For every finite set of bins B and integer $m \geq 1$, we give a deterministic algorithm with competitive ratio $r(B, m)$.
- For every finite set of bins B and integer $m \geq 1$, we give an upper bound of $r(B, m)$ on the competitive ratio of any algorithm.

These results show that $r(B, m)$ is the best possible competitive ratio for the set of bins B, and the upper bound $1/m$. This result reduces to the result of Woeginger and Zhang [11] for the case $m = 1$, and to the results of [3, 8], if both $m = 1$ and $B = \{1\}$ hold. If only $B = \{1\}$ holds, then it follows from our result that the best competitive ratio is $m/(m + 1)$.

We show the following results for vector covering.

- An upper bound of $1/2^{d(1-o(1))}$ on the competitive ratio for the vector covering problem for arbitrary sets B.
- An algorithm with competitive ratio $1/2$ for the case where each bin in B is a standard basis vector.
- An upper bound of $1/2$ on the competitive ratio for this case
- For the case of unit prefix vectors, (i.e. bins are of the form $(1, \ldots, 1, 0, \ldots, 0)$ with i and $d-i$ ones and zeros), and $|B| = d$) we show that this model is harder than the case where B consists of all standard basis vector bins (for large enough values of d) by giving an upper bound of $O(1/\log d)$ on the competitive ratio of any algorithm.
- For the same case we show that the relation between models holds also for $d = 2$ by giving a upper bound of $4/9$ on the competitive ratio of any algorithm for $d = 2$.
- For the case where B is the set of all possible binary vectors, we show that this model is easier than the basic model of one bin type which is all 1 (for large enough values of d) by giving an algorithm of ratio $\Omega(1/\log d)$.
- For the same case we show that the relation between models holds also for $d = 2$ by giving an algorithm of ratio larger than 0.4.

All upper bounds (negative results) in this paper hold both for deterministic and for randomized algorithms. In order to prove upper bounds we use an adaptation of Yao's theorem [12] which states that an upper bound of a deterministic algorithm against a fixed distribution on the input is also an upper bound for randomized algorithms. For maximization problems the bound is given by $E\,(V_{on}/V_{opt})$ or by $E(V_{on})/E(V_{opt})$ [4, 5]. All upper bounds are proved for algorithms that are allowed to have additive constants (i.e. $\eta \neq 0$) . In order to prove upper bounds (negative results) for such algorithms, we show that all upper bounds hold for arbitrarily long sequences.

2 One-Dimensional Covering

In this section we show that for every set of one-dimensional bins B and integer $m \geq 1$, the ratio $r(B, m)$ is the best possible competitive ratio.

Theorem 2.1 *For every collection of bin sizes B, and parameter m, there exists an algorithm for bin covering with asymptotic worst case ratio $r(B, m)$. This result is best possible among all on-line randomized algorithms.*

We start with the algorithm. Later we prove the upper bound which shows its optimality.

For a given set of bins B, and parameter m, we will use q for $q(B,m)$, and r for $r(B,m)$. Let $t = \lceil -\log_2 m(q-1) \rceil$. Define a partition of the interval of items $(0, 1/m]$ into the following subintervals:

- For $1 \leq j \leq k-1$ and $0 \leq l \leq t-1$, let $I_{j,l} = (c_{j+1}/2^l, c_j/2^l]$.
- Let $I_{k,t} = (0, c_k/2^{t-1}]$.

We assign each interval $I_{j,l}$ a corresponding bin size $d_{j,l}$. We define $d_{k,t}$ to be $b_1 = 1$. For each other interval $I_{j,l}$ we define $d_{j,k} = b(c_{j+1})$. The algorithm keeps one open bin of for each interval, this bin is used only for packing items in the interval. Each new item is classified and assigned to the corresponding bin. When a bin is covered, it is closed, and a new bin of the same size is opened and used for this interval.

Lemma 2.1 *The competitive ratio of the above algorithm is at least r*

Proof: We show that each covered bin is covered by at most q times its size. For every first type interval $I_{j,l}$ we show that its corresponding closed bins are covered by at most c_j/c_{j+1} times their size. Let $b(c_{j+1}) = yc_{j+1}$, then each such bin is covered by exactly $y2^l$ items of size at most $c_j/2^l$, and the total size of the items that cover it is at most yc_j. Thus the ratio between the size of items that cover the bin and the size of the bin is at most c_j/c_{j+1} as required. Since $q \geq c_j/c_{j+1}$, the bin is covered by at most q times its size.

For the interval $I_{k,t}$, since all items are bounded by $1/(2^t m)$, a covered bin is covered by items with total size at most $1+1/(2^t m)$. Since $2^t \geq 1/(m(q-1))$, $1+1/(2^t m) \leq q-1+1 = q$, and each such bin is covered by at most q, which is q times a size of a unit bin.

The total number of bins that are never closed is $O(kt)$, which is $O(mst)$ where $s = |B|$, this number depends solely on B and m, and is an additive constant. ■

Lemma 2.2 *The competitive ratio of any deterministic or randomized algorithm for bin covering is at most r.*

Proof: We start with the deterministic case. Let j be an integer such that $c_j/c_{j+1} = q$. Let y be an integer such that $yc_j = b(c_j)$. Let N be a large integer.

In order to define a list of items, we define a set of sizes D. Let $D = B \cup C'$ where $C' = \{xc' | x \in \{1, \ldots, 2m\}, c' \in C(m)\} \cap [1/2m, 1]$. Enumerate the numbers in D, $D = \{d_1, \ldots, d_p\}$ where $d_1 > d_2 > \ldots > d_p$. Now define $\varepsilon > 0$ to be a number that satisfies $2m \cdot 2^N \varepsilon < b_s$ and for every $1 \leq j \leq p-1$, $2m \cdot 2^N \varepsilon < d_j - d_{j+1}$. The list contains $2^N y$ items with size ε (sand) followed by $2^{N-i} y$ items with size $c_j - 2^i \varepsilon$ for a fixed i, $0 \leq i \leq N$.

The optimal off-line assignment uses bins with size $b(c_j)$ to pack all items. Each bin contains y items with size $c_j - 2^i \varepsilon$ and $2^i y$ items with size ε. This covers 2^{N-i} bins and $V_{opt} = 2^{N-i} b(c_j) = 2^{N-i} y c_j$. We change the packing of the on-line algorithm without reducing the value of the on-line algorithm in such a way that the on-line algorithm will use only bins of weights c_j and c_{j+1}. Consider a bin of the on-line algorithm of size b which is covered, i.e. it contributes some amount to the on-line algorithm value. Since $y \leq 2m$, and $b \geq b_s > 2m2^N \varepsilon$,

the bin contains at least one big item. Let $a > 0$ be the number of big items in the bin. Assume that $a \leq 2m$, otherwise remove items and let $a = 2m$, since items are larger than $1/2m$ and the bin is of size at most 1, then the bin is still covered.

Let μ be the total size of small items in the bin. If $\mu \geq 2^i a\varepsilon$ replace the bin by a bins of size c_j containing each one big item, and $\lfloor \mu/(a\varepsilon) \rfloor$ small items. Otherwise, partition the items in the same way, but use bins of weight c_{j+1} instead.

We show that the new bins are covered and that the value of the on-line algorithm is not reduced, i.e. $b \leq ac_j$ in the first case and $b \leq ac_{j+1}$ in the second case. In the first case since $\mu/a \geq 2^i\varepsilon$, the total weight of items in a new bin is $(c_j - 2^i\varepsilon) + \lfloor \mu/(a\varepsilon) \rfloor \varepsilon \geq c_j$. Hence each new bin is covered. On the other hand $b \leq ac_j + y2^N\varepsilon$. According to the definition of D, $ac_j \in D$. Assume to the contrary that $b > ac_j$. Since $b \in B$ then $b \in D$, thus $b - ac_j > y2^N\varepsilon$ and hence the bin would not be covered. We conclude that $b \leq ac_j$, and the replacement of b by a bins of size c_j does not change the on-line algorithm value and each new bin of size c_j is covered by at least c_j. In the second case, If $\mu < 2^i a\varepsilon$, the weight of an item in a new bin is at least $c_j - 2^i\varepsilon > c_{j+1}$. Hence each new bin is covered. On the other hand $b < ac_j$. Assume to the contrary that $ac_{j+1} < b < ac_j$, then $b/a \in C$, contradicting the fact that c_j in the successor of c_{j+1} in C. We conclude that new bins are covered, the on-line algorithm value was not reduced, and now the on-line algorithm uses only bins of sizes c_j and c_{j+1}.

For $0 \leq i < N$, let Z_i be the number of on-line algorithm bins with the amount of least $2^i\varepsilon$ weight of sand but less than $2^{i+1}\varepsilon$. Let Z_N be the number of on-line algorithm bins with weight of sand at least 2^N. We have $\sum_{i=0}^{N} Z_i 2^i \leq y2^N$. The on-line algorithm value would be $c_j(Z_i + Z_{i+1} + \ldots + Z_N) + c_{j+1}(2^{N-i}y - (Z_i + Z_{i+1} + \ldots + Z_N)) = yc_{j+1}2^{N-i} + (c_j - c_{j+1})(Z_i + Z_{i+1} + \ldots + Z_n)$.

We multiply the on-line and off-line algorithms values for every $i \geq 1$ by $2^{i-1}/2^N$, and the value for $i = 0$ by $1/2^N$. For the on-line algorithm we get

$$\begin{aligned}
& yc_{j+1}(\sum_{i=1}^{N} 2^{N-i}2^{i-1}/2^N + 2^N/2^N) + (c_j - c_{j+1})\sum_{i=0}^{N} Z_i 2^i/2^N \\
\leq\ & y(c_{j+1}(N+2)/2 + (c_j - c_{j+1})) \\
=\ & y/2((N+2)c_{j+1} + 2c_j - 2c_{j+1}) \\
=\ & y/2(Nc_{j+1} + 2c_j)
\end{aligned}$$

For the optimal off-line algorithm we get

$$\begin{aligned}
& \sum_{i=1}^{N} 2^{i-1}/2^N \cdot 2^{N-i}yc_j + 1/2 \cdot 2^N yc_j \\
=\ & (yc_j/2^N)(N+2)2^{N-1} \\
=\ & yc_j(N+2)/2\ .
\end{aligned}$$

If the competitive ratio is r and $q = 1/r$, then also for the convex sums $yc_j(N+2)/2 \leq qy/2(Nc_{j+1} + 2c_j)$ or $q \geq (N+2)/((N(c_{j+1}/c_j) + 2)$. Hence $r = c_{j+1}/c_j + \delta_N$, where δ_N is arbitrarily close to 0, for large enough values of N.

For the randomized case we use the adaptation to Yao's theorem, and the probabilities are the factors we multiplied by in the deterministic case. We get that $E(V_{on})/E(V_{opt}) \leq c_{j+1}/c_j$. ■

3 Multi-dimensional Vector Covering

In this section we consider variable sized vector covering. In the classical vector covering problem, B contains a single bin $\mathbf{1}= (1, 1, \ldots, 1)$. We study a simple extension of this model, and allow each bin to be a binary d-dimensional vector.

We start by showing an upper bound on the competitive ratio for arbitrary sets B.

Lemma 3.1 *The competitive ratio of any deterministic or randomized on-line algorithm for binary variable sized vector covering is at most* $1/2^{d(1-o(1))}$.

Proof: Let B be the set of all binary vectors v, such that $v_1 = 1$, and there are exactly $\lfloor (d+1)/2 \rfloor$ non-zero components in v. The sequence begins with k items with size $(1, 0, \ldots, 0)$. Next a vector $x \in B$ is chosen uniformly in random. Let $x^0 = x - (1, 0, \ldots, 0)$. Now k vector items with size x^0 arrive. Benefit can be gained only from bins of size x, opened in the first phase. It is easy to see that the competitive ratio is at most $1/|B| = 1/2^{d(1-o(1))}$. ■

Due to this negative result, we do not study the general model, but restrict ourselves to specific interesting cases of B. We show some relations between the different models.

3.1 Covering a single type of bin

We start with an easy model where B consists of a single type of bin b. Assume that the number of non-zero components in b is i, then we can reduce this problem to the case of vector covering bins of dimension i, where all components are 1 (the basic vector covering problem). This is true since both the on-line and off-line algorithms do not use the coordinates which are zero in the vector b. Thus the best competitive ratio for this case is $\Theta(1/i)$ [1].

3.2 Covering standard basis vector bins

In this model we let B consist of bins which have one non-zero component. We show that for any such set of bins, the best competitive ratio is 1/2. We consider this model since this is a case where all bins have the same bin weight.

The upper bound (negative result) follows from the bound of [8] by picking one $b \in B$, where the ith coordinate of b is non-zero, and giving a sequence where all vectors are d-dimensional, with all coordinates zero, except for the ith coordinate which is the same as in [8]. We give an easier proof for the case where $|B| > 1$:

Lemma 3.2 *If B consists of standard basis vector bins only, and $|B| > 1$, then the competitive ratio of any deterministic or randomized on-line algorithm is at most 1/2.*

Proof: We assume that all vectors are two-dimensional otherwise we can pick two bins $b_1, b_2 \in B$ which are non-zero in coordinates i and j respectively, and reduce the problem into two dimensions by always giving all other coordinates the value zero.

The upper bound sequence contains $2k$ items with size $(1-\varepsilon, 1-\varepsilon)$. Then an additional $2k$ items arrive. These items are all of size $(0, \varepsilon)$ with probability $\frac{1}{2}$, and of size $(\varepsilon, 0)$ with the same probability (all arriving items are of the same size).

The optimal off-line algorithm value in both cases would be $2k$ ($2k$ bins of either type (1,0) or type (0,1) are covered).

The on-line algorithm may assign one or two big items into a bin of type (1,0) or (0,1). Let x_1 be the number of (1,0) bins, used by the on-line algorithm for one item, and x_2 be the number of (0,1) bins with one big item. Let z be the number of bins that were covered by the on-line algorithm after the arrival of big items.

The competitive ratio in each case is $\frac{(x_i+z)}{2k}$. Since both cases are equally likely, $r \leq \frac{1}{2}\left(\frac{x_1+z}{2k}\right) + \frac{1}{2}\left(\frac{x_2+z}{2k}\right) = \frac{1}{4k}(x_1+x_2+2z)$. By the definitions of x_1, x_2 and z, $x_1+x_2+2z \leq 2k$; hence $r \leq \frac{2k}{4k} = \frac{1}{2}$. ∎

We give an optimal greedy algorithm. We assume that $|B| = d$ and it contains all bins of the standard basis. Otherwise we just cancel the coordinates that do not appear in any bin (are zero in all bins), since no algorithm can benefit from these coordinates, and treat the input as a lower dimension vector.

The algorithm has at most d open bins at a time. Each arriving vector is classified to a type among $\{1, 2, \ldots, d\}$ according to its largest component.

Denote a bin whose ith coordinate is 1 by b_i. After an arriving item is classified to a type j, then if there is an open bin of size b_j, the item is assigned to it, and if after the item is assigned to a bin, the bin is covered, we close the bin (and never use it again).

If no bin of size b_j is open, open a new one and continue in the same way.

Theorem 3.1 *The competitive ratio of the greedy algorithm is at least* $\frac{1}{2}$.

Proof: For the optimal off-line algorithm, define by S_{opt}, the active area of items, that is, the area of items that was used to cover bins (i.e. the area that is not wasted, which is in the case of standard basis vectors, 1 for each covered bin). Clearly, $V_{opt} = S_{opt}$. Since all bins have only one non-zero coordinate, the active area of each item is in one coordinate. For an item a, let $a_{\max} = \max_{1 \leq i \leq d} a_i$. Clearly, $S_{opt} \leq \sum_{a \in A} a_{\max}$, where A is the sequence of items.

On the other hand, consider the area $\sum_{a \in A} a_{\max}$, the on-line algorithm packed according to the maximum component.

Let α_i be the number of bins of type b_i that the on-line algorithm managed to cover. For each such bin, the ith coordinate is covered by less than 2 (since at the time the last item was assigned there, the coordinate was less than 1, and the new item adds at most 1). There is at most one open bin which is never closed, and its ith coordinate is also less than 1. Hence $\sum_{a \in A_i} a_{\max} \leq 2\alpha_i + 1$, where A_i are items that were classified to type i. Summing over all $1 \leq i \leq d$ we get: $\sum_{a \in A} a_{\max} \leq 2V_{on} + d$, which gives $V_{on} \geq \frac{1}{2}V_{opt} - \frac{d}{2}$. ∎

3.3 Covering unit prefix bins

We define a model which is similar to the previous one. Let B_1 be a set that contains exactly d possible bin sizes. Bin i, which also has bin weight i, is a d-dimensional vector whose i leftmost coordinates are 1, and all others are 0. We show that using this set of allowed bins instead of d standard basis vector bins, makes the problem of on-line covering harder.

Theorem 3.2 *The competitive ratio of any deterministic or randomized on-line algorithm for covering the increasing set of bins has competitive ratio of at most* $O(1/\log d)$.

Proof: The upper bound sequence consists of $2k$ items with size $a = (1-\varepsilon, 0, 0, \ldots, 0)$ and then with probability p_i (which we fix later), $2k$ items with size $(\varepsilon, 1, \ldots, 1, 0, \ldots, 0)$ arrive ($d-i$ zeros), and for $2 \le i \le d$, with probability q_i, (which is also fixed later), k items with size $(0, 1, \ldots, 1, 0, \ldots, 0)$ arrive ($d-i$ zeros). q_1 is the probability that no further items (except the first $2k$ items) arrive. Hence the probabilities should be fixed so that $\sum_{i=1}^{d}(p_i + q_i) = 1$.

Let α_i be the number of bins of bin weight i that the on-line algorithm opens for one item of size a, and β_i to be the number of bins of bin weight i it opens, assigning two items with size a to each.

Let us calculate the competitive ratio for each case: Consider the cases where items with $d-i$ rightmost zero coordinates arrive. With probability p_i, the on-line algorithm manages to cover all bins of weight of at most i that it opened.

The optimal off-line assignment would use only bins of bin weight i, and would assign one item of each type to every bin. Thus $C_{opt} = 2ki$, and $C_{on} = \sum_{j=1}^{i}(\alpha_j + \beta_j) \cdot j$. On the other hand, with probability q_i, the on-line algorithm would cover only bins with at least two items with size a (otherwise the leftmost component is not covered).

The optimal off-line assignment would use only bins of bin weight i, and would assign two items with size a, and one item of the other type to each bin. Thus $C_{opt} = ki$ and $C_{on} = \sum_{j=1}^{i} \beta_j \cdot j$. According to the initial on-line assignment, we have $\sum_{i=1}^{d}(\alpha_i + 2\beta_i) \le 2k$. Let $p_i = q_i = \frac{1}{2i(i+1)}$ for $i < d$ and $p_d = q_d = \frac{1}{2d}$.

The expectation of the optimal off-line value is

$$\begin{aligned} E(V_{opt}) &= \sum_{i=1}^{d}(p_i 2ki + q_i ki) = 3k\Big(\sum_{i=1}^{d-1}\frac{1}{2(i+1)} + \frac{1}{2}\Big) \\ &= \frac{3}{2}k\Big(\sum_{i=1}^{d-1}\frac{1}{i+1} + 1\Big) = \frac{3}{2}k\Big(\sum_{i=1}^{d}\frac{1}{i}\Big) \ge \frac{3}{2}k\ln d \end{aligned}$$

The expectation of the on-line value is

$$\begin{aligned} E(V_{on}) &\le \sum_{i=1}^{d}\sum_{j=1}^{i}(p_i(\alpha_j + \beta_j)j + q_i\beta_j j) \\ &= \sum_{i=1}^{d-1}\frac{1}{2i(i+1)}\sum_{j=1}^{i}(2\beta_j + \alpha_j)j + \sum_{j=1}^{d}\frac{1}{2d}(2\beta_j + \alpha_j)j \end{aligned}$$

$$
\begin{aligned}
&= \sum_{j=1}^{d}\Big(\sum_{i=j}^{d-1}\frac{1}{2i(i+1)}j+\frac{1}{2d}j\Big)(2\beta_j+\alpha_j)\\
&= \sum_{j=1}^{d}\frac{j}{2}\Big(\sum_{i=j}^{d-1}\Big(\frac{1}{i}-\frac{1}{i+1}\Big)+\frac{1}{d}\Big)(2\beta_j+\alpha_j)=\sum_{j=1}^{d}\frac{j}{2}\cdot\frac{1}{j}(2\beta_j+\alpha_j)\le k
\end{aligned}
$$

The probabilities are valid since

$$
\sum_{i=1}^{d}(p_i+q_i)=\sum_{i=1}^{d-1}\frac{1}{i(i+1)}+\frac{1}{d}=\sum_{i=1}^{d-1}\Big(\frac{1}{i}-\frac{1}{i+1}\Big)+\frac{1}{d}=1
$$

Thus the competitive ratio is at most $\frac{k}{\frac{3}{2}k\ln d}=\frac{2k}{3k\ln d}=O\Big(\frac{1}{\log d}\Big)$. ■

We show that this problem is significantly harder than standard basis vector bins, not only asymptotically but for $d=2$ as well (in this case $B=\{(1,1),(1,0)\}$).

Lemma 3.3 *The competitive ratio of any deterministic or randomized algorithm for $d=2$ and unit prefix bins is at most $4/9$.*

Proof: The Lemma follows from a more careful examination of the sequence given in the proof of Theorem 3.2 for $d=2$. Consider the following sequence. First $2k$ items with size $a=(1-\varepsilon,0)$ arrive. Next there are four cases (we fix the probability of each case later).

- No more items arrive
- An additional $2k$ items with size $(\varepsilon,0)$ arrive.
- $2k$ items with size $(\varepsilon,1)$ arrive.
- k items with size (0,1) arrive.

Let x_1 be the number of (1,1) bins that the on-line algorithm opens with one item of size a, and x_2 with two items. Let y_1 be the number of (1,0) bins that the on-line algorithm opens with one item of size a and y_2 with two items. The optimal off-line algorithm value is k in the first case (bins of size (1,0) with pairs). $2k$ in the second case (one item of each size in (1,0) bins). $4k$ in the third case (one item of each size in (1,1) bins) and $2k$ in the last case (all items are packed in bins of size $(1,1)$, each bin contains two items with size a and one item with size $(0,1)$).

The on-line algorithm cannot use only late-arriving items to cover bins. Its value is y_2 in the first case, y_1+y_2 in the second, $2x_1+2x_2+y_1+y_2$ in the third and y_2+2x_2 in the last case.

According to the definitions of x_i and y_i we have $x_1+2x_2+y_1+2y_2\le 2k$. We use the probabilities $\frac{1}{4}$ for all cases. The expectation of the off-line algorithm value is $E(V_{opt})=\frac{9k}{4}$. The expectation of the on-line algorithm value is

$$
\begin{aligned}
E(V_{on}) &\le \frac{1}{4}(2x_1+4x_2+2y_1+4y_2)\\
&\le 2\frac{2k}{4}=2\frac{k}{2}=k
\end{aligned}
$$

and thus the competitive ratio is at most $\frac{4}{9}$.

■

3.4 Covering all possible bins

In this model, B consists of $2^d - 1$ bins, i.e. each binary d-dimensional bin may be used. We show that the basic covering model (the only allowed bin is the all 1 bin) is harder than this model. We define the following algorithm.

Let $k = \lceil \log_2(d^2) \rceil$. We define type vectors by vectors in the set A^d where $A = \{0\} \cup \{\frac{1}{2^i} \mid 0 \leq i \leq k\}$. A^d consists of $(k+2)^d$ different type vectors. The algorithm keeps one open bin of a corresponding size for each vector type. To determine the bin vector for a certain type vector we do the following:
For each $0 \leq s \leq k$, let α_s be the sum of sizes of all components of size $\frac{1}{2^s}$ of the type vector, which is $\frac{m}{2^s}$, where m is the number of $\frac{1}{2^s}$ components. Let s_1 be an integer for which α_s is maximized. The corresponding bin would have zeros in all coordinates which are less than $\frac{1}{2^{s_1}}$ in the type vector, and 1 in all coordinates which are at least $\frac{1}{2^{s_1}}$.

We keep one open corresponding bin for each possible type vector. On arrival of an item vector a, a is identified with a type vector, then it is assigned to a bin that corresponds to this type. A covered bin is closed and replaced by a new bin of the same size.

For each arriving item vector a, compute its type vector in the following way:
Let i be the coordinate such that $a_i = \max_{1 \leq j \leq d} a_j$. Let a' be the vector $\frac{a}{a_i}$. Replace each coordinate a'_j by the maximum number $\frac{1}{2^s}$, such that $a'_j \geq \frac{1}{2^s}$. If $s > k$, then replace a'_j by zero. The new vector a'' is the type vector of a. Let $a''' = a'' \cdot a_i$. a is assigned to the open bin which corresponds to the type vector a''.

Theorem 3.3 *The above algorithm has competitive ratio* $\Omega(\frac{1}{\log d})$.

Proof: Consider each vector type separately. Denote by $V(a)$, the weight of item a, and by W, the total weight of all items. The item weight of a'' is at least $\frac{V(a)}{2}(1 - \frac{1}{d})$. This is true since the components which are at least $\frac{1}{2^k}$, are divided by at most 2. The smaller components altogether sum to at most $d \cdot \frac{1}{d^2} \cdot V(a)$.

We assume that instead of covering with the real vectors, each vector a is replaced by the corresponding a'''; this assumption can only reduce the competitive ratio. Consider a certain type vector b. Let $1/2^s$ be the size of the smallest component of b. For each component i of a bin which is corresponds to b, component i is covered by at most 2 (since all items assigned to this bin have equal weight in the smallest components). Thus the weight that is gained from a vector a (in the on-line algorithm value) is at least $\frac{1}{2}\frac{1}{k+1}\frac{V(a)}{2}(1 - \frac{1}{d}) \geq \frac{1}{2\log_2 d+2}\frac{V(a)}{8}$. There is only one bin for each type that is never covered. Thus the value gained on items of type vector T is at least $\left(\sum_{a \in T} \frac{1}{2\log_2 d+2} \frac{V(a)}{8}\right) - d$. Summing over all types we get at least

$$V_{on} \geq \sum_a \frac{1}{2\log_2 d + 2} \frac{V(a)}{8} - d \cdot (k+2)^d \ .$$

Since $V_{opt} \leq W$ we get $V_{on} \geq \frac{1}{16}\frac{1}{(\log_2 d+1)} V_{opt} - C(d)$, where $C(d)$ is an additive constant that depends only on d. ■

	$[0,\frac{1}{8})$	$[\frac{1}{8},\frac{1}{7})$	$[\frac{1}{7},\frac{1}{6})$	$[\frac{1}{6},\frac{1}{5})$	$[\frac{1}{5},\frac{1}{4})$	$[\frac{1}{4},\frac{1}{3})$	$[\frac{1}{3},\frac{1}{2})$	$[\frac{1}{2},\frac{4}{5})$	$[\frac{4}{5},1]$
$[0,\frac{1}{8})$	1	1	1	2	4	4	4	9	16
$[\frac{1}{8},\frac{1}{7})$	1	1	1	3	3	3	8	9	16
$[\frac{1}{7},\frac{1}{6})$	1	1	1	6	6	6	8	9	16
$[\frac{1}{6},\frac{1}{5})$	2	3	6	5	7	7	8	9	16
$[\frac{1}{5},\frac{1}{4})$	4	3	6	7	5	12	10	13	17
$[\frac{1}{4},\frac{1}{3})$	4	3	6	7	12	5	11	14	17
$[\frac{1}{3},\frac{1}{2})$	4	8	8	8	10	11	5	15	15
$[\frac{1}{2},\frac{4}{5})$	9	9	9	9	13	14	15	5	5
$[\frac{4}{5},1]$	16	16	16	16	17	17	15	5	5

Table 1: The partition into classes

We show that the problem with all possible bins is easier for two dimensions as well. For that, we give an algorithm for two-dimensional bin covering with $B = \{(0,1),(1,0),(1,1)\}$ that achieves a competitive ratio strictly higher than 0.4.

We partition $[0,1]^2$ into several classes and run a greedy algorithm on each class separately. Some classes are partitioned further as explained below. The basic partition is given in Table 3.4. The table shows the class that each item belongs to, according to the magnitude of its both coordinates. A trivial argument will show that the algorithm is 3/8 competitive. By using more careful analysis, we will increase this to $53/132 \approx 0.401515$, and with one final idea further increase the competitive ratio to $103/252 \approx 0.40873$.

We discuss each class separately, focusing on the amount of bins that need to be simultaneously open for each class. We specify how many items are required to cover the bin. We give an analysis of the ratio between bin weight and the items weight for covered bins in each class, this depends on the item weight that the bin has in each component. Denote this ratio for class i by r_i.

1. Items with both coordinates in $[0,\frac{1}{6})$. We keep at most three open bins for this size, i.e. we partition this set further: Let (α,β) be an item such that $\alpha < \frac{1}{6}$ and $\beta < \frac{1}{6}$. We pack items such that $\alpha < \frac{\beta}{2}$ into bins of size (0,1), items such that $\beta < \frac{\alpha}{2}$ into bins of size (1,0), and all other items ($\frac{\beta}{2} \le \alpha \le \beta$ or $\frac{\alpha}{2} \le \beta \le \alpha$) into bins of size (1,1).

   The items weight in each covered bin can be bounded in the following way:
   For (1,1) bins, the smaller coordinate is bounded by $1+\frac{1}{6}$, and the larger by twice the smaller. For (1,0) or (0,1) bins, the covered coordinate is at most $1+\frac{1}{6}$ and the other is at most half of it. Hence $r_1 \le \max\{3.5/2, 7/6 \cdot 3/2\} = 1.75$.

2. One coordinate is in $[0,\frac{1}{8})$, and the other is in $[\frac{1}{6},\frac{1}{5})$. We keep at most two open bins for this case, one bin of size $(0,1)$ for items (a,b) such that $a < b$, and the other of size $(1,0)$ for items (a,b) such that $a > b$. Every bin is covered by exactly 6 items. Hence $r_2 \le 6/5 + 6/8 = 1.95$.

3. One coordinate is in $[\frac{1}{8},\frac{1}{7})$ and the other is in $[\frac{1}{6},\frac{1}{3})$. We have at most one open bin (of size (1,1)) for this range of sizes. Every 8 items cover a bin. Hence $r_3 \le (\frac{8}{7}+\frac{8}{3})/2 = \frac{40}{21}$

4. For $2 \leq i \leq 4$, we consider the case that one coordinate is in $[0, \frac{1}{8})$ and the other is in $[\frac{1}{i+1}, \frac{1}{i})$. For every i we have at most one open bin of size (0,1) and one of (1,0) for the symmetric case. Each covered bin contains exactly $i+1$ items, and thus $r_4 \leq \frac{i+1}{i} + \frac{i+1}{8} \leq 1.875$.

5. For $1 \leq i \leq 5$, we consider items where both coordinates are in $[\frac{1}{i+1}, \frac{1}{i})$ (for $i = 1$ the range is $[\frac{1}{2}, 1]$). We have at most one bin open for each i (bin of size (1,1)). Each covered bin contains exactly $i+1$ items, giving $r_5 \leq \frac{i+1}{i}$.

6. Items with one coordinate in $[\frac{1}{7}, \frac{1}{6})$ and the other in $[\frac{1}{6}, \frac{1}{3})$ are assigned into bins of size (1,1). There is at most one open bin for this range, each covered bin containing 7 items and $r_6 \leq (\frac{7}{6} + \frac{7}{3})/2 = 1.75$.

7. Items with one coordinate in $[\frac{1}{6}, \frac{1}{5})$ and the other in $[\frac{1}{5}, \frac{1}{3})$ are assigned into bins of size (1,1). Similarly to previous cases $r_7 \leq 1.6$.

8. Items with one coordinate in $[\frac{1}{8}, \frac{1}{5})$ and the other in $[\frac{1}{3}, \frac{1}{2})$. We keep two open bins for this class of sizes (0,1) and (1,0). Each closed bin contains exactly 3 items, and $r_8 \leq 2.1$.

9. Items with one coordinate in $[\frac{1}{2}, \frac{4}{5})$ and the other in $[0, \frac{1}{5})$. There are at most two open bins (of sizes (1,0) and (0,1)) for this range. Each closed bin contains two items and $r_9 \leq 2$.

10. One coordinate in $[\frac{1}{3}, \frac{1}{2})$ and the other in $[\frac{1}{5}, \frac{1}{4})$; each bin is of size (1,1), and 5 items cover a bin. Hence $r_{10} \leq 1.875$.

11. One coordinate in $[\frac{1}{3}, \frac{1}{2})$ and the other in $[\frac{1}{4}, \frac{1}{3})$; each bin is of size (1,1), and covered by 4 items which gives $r_{11} \leq 5/3 \approx 1.667$.

12. One coordinate in $[\frac{1}{4}, \frac{1}{3})$ and the other in $(\frac{1}{5}, \frac{1}{4})$; similar to the previous case we get $r_{12} \leq 35/24 \approx 1.458$.

13. One coordinate in $[\frac{1}{5}, \frac{1}{4})$ and the other in $[\frac{1}{2}, \frac{4}{5})$. We use bins of weight 1, two items per bin, and get $r_{13} \leq 2.1$.

14. One coordinate in $[\frac{1}{4}, \frac{1}{3})$ and the other in $[\frac{1}{2}, \frac{4}{5})$. Here we use bins of weight 2, a bin is covered by 4 items, which gives a ratio of at most $r_{14} \leq 34/15 \approx 2.2667$.

15. One coordinate in $[\frac{1}{3}, \frac{1}{2})$ and the other in $[\frac{1}{2}, 1]$. $r_{15} \leq 2.25$.

16. Items with one coordinate in $[\frac{4}{5}, 1]$ and the other in $[0, \frac{1}{5})$, $r_{16} \leq 2.4$.

17. One coordinate in $[\frac{1}{5}, \frac{1}{3})$ and the other in $[\frac{4}{5}, 1]$. This is the only problematic case; each bin is of weight 1, and contains exactly 2 items. $r_{17} \leq 2.667$. We discuss this range again in the proof, referring to those items as "last range items".

Theorem 3.4 *The competitive ratio of the above algorithm is strictly above* 0.4.

Proof: It is easy to see that the algorithm is 3/8 competitive, since the ratio of items weight to bin weight of every closed bin is at most 8/3. To increase the competitive ratio to $r = 53/132$, we do the following steps.

We change the sequence in such a way that neither the on-line algorithm value nor the optimal off-line algorithm value changes. Consider all off-line algorithm bins which contain some number of last range items. We consider several cases.

- The off-line algorithm bin is of weight 1. We call a coordinate of a last range item "passive" if it covers a zero coordinate of a bin. If the small coordinate of a last range item is passive, replace it by 0. If the large coordinate of a last range item is passive, replace it by 4/5.
- The off-line algorithm bin is of weight 2 and it contains at least 2 last range items. In this case, replace all large coordinates (of size at least 4/5) of such last range items by 4/5. There are two cases, according to whether both components are covered by at least one large coordinates of item, or not. Decreasing the large coordinate would not change the off-line algorithm value, since both components of the off-line algorithm bin stay covered. This is true for both cases.
- The off-line algorithm bin is of weight 2, and contains one last range item, z. Reduce any coordinate of any other item in the bin which is larger than 4/5 to 4/5. If possible, decrease also the large coordinate of z to 4/5 (i.e. if the total size of other items in that coordinate is at least 1/5). The off-line algorithm bin stays covered since both coordinates of z are at least 1/5.

These changes do not change the value of the on-line algorithm. Items with a coordinate of size at least 4/5 are packed in pairs. Hence reducing them to 4/5 does not harm the packing. A small coordinate of a last range item is passive in the packing of the on-line algorithm, hence reducing it to zero also does not harm the packing.

We ignore all the bins used by the on-line algorithm, that are not completely covered. We still allow the optimal off-line algorithm to use them. This may only decrease the competitive ratio. Since the number of such bins is constant, this only adds an additive constant.

Consider a bin that the on-line algorithm packed for range 17, that contains an item whose size was decreased. The decreased item had either a zero small coordinate (and then its item weight is at most 1), or a 4/5 (or zero) large coordinate (and then its item weight it at most $4/5 + 1/3 = 17/15$). The item weight of the second item is at most 4/3. Hence the ratio between items weight and bin weight is at most 37/15.

Consider on-line algorithm bins with items weight of more than $1/r$. These may be only bins created for range 17. There are no changed last range items in these bins since $37/15 < 1/r$.

Let A_1 be the total current weight of items packed in such bins. All non-modified last range items share an off-line bin with only items of size in $([0, \frac{1}{5}), [0, \frac{4}{5}]) \cup ([0, \frac{4}{5}], [0, \frac{1}{5}))$. Let A_2 be the total current weight of items with size in $([0, \frac{1}{5}), [0, \frac{4}{5}]) \cup ([0, \frac{4}{5}], [0, \frac{1}{5}))$, which were not in range 17 originally. A_2 can include some items that were originally in range 16. Let A_3 be the weight of all other items, that are not included in $A_1 \cup A_2$.

Note that all items whose weight is included in A_2 have a ratio of items weight to bin weight of at most 2.2. This is true since for items that were not modified, the ratio is at most 2.1. Items that were modified, are packed in pairs in range 16. Moreover, such a bin contains exactly two items, and for at least one of them, its big coordinate was reduced to 4/5. A packed bin which contains at least one modified item has items weight of at most 2.2.

Since every item in A_1 has a weight of at most $\frac{4}{3}$, and in its off-line algorithm bin there are items in A_2 of weight of at least $\frac{2}{3}$, then $A_2 \geq \frac{1}{2}A_1$.

The total off-line algorithm value is at most $A_1 + A_2 + A_3 + \eta$, where η is the total items weight of items that the on-line algorithm assigned to bins that were never closed. The on-line algorithm value is at least $V_{on} \geq \frac{3}{8}A_1 + \frac{10}{22}A_2 + rA_3 \geq r(A_1 + A_2 + A_3) + \frac{7}{132}A_2 - \frac{7}{264}A_1 \geq rV_{opt}$ since $A_1 \leq 2A_2$. Thus since for all A_3 items, each bin has ratio of items weight to bin weight of at most $1/r$, the inequalities hold and the algorithm is $r = 53/132$ competitive.

To improve the bound it is possible to change the on-line algorithm packing of the bins for each of the last two ranges, so that a pair is together in a bin if either both items, or neither of them were changed. This cannot be done on-line, but is done only for the sake of proof. There would be at most one bin in each range which would not suit this arrangement, and it is possible to ignore these two bins, adding a small amount to the additive constant. In this way a bin of the last range with two changed items has items weight of at most $34/15 \approx 2.26667 < 252/103$, and a bin with two changed items of the range before has items weight of at most 2. Hence we can do the same calculation as before, using 2.1 instead of 2.2. This proves that the algorithm is 103/252 competitive. ■

4 Open Problems

Some problems for multi-dimensional covering were not answered in this paper.

For the model of unit prefix bins, we gave an upper bound of $O(1/\log d)$. It is not known whether it is possible to design an algorithm with competitive ratio $\Omega(1/\log d)$, or if it possible to give a smaller negative result?

For the model where B contains all binary vectors, we gave an algorithm with competitive ratio $\Omega(1/\log d)$. Is it possible to give an algorithm with higher competitive ratio, e.g. constant competitive ratio, or is it possible to prove an upper bound of $O(1/\log d)$? This raises the question whether the model of unit prefix bins is harder than the model where B contains all binary vectors.

Acknowledgments: I would like to thank Gerhard Woeginger for introducing me to some of the problems discussed in this paper. I would also like to thank Yossi Azar for helpful discussions.

References

[1] N. Alon, Y. Azar, J. Csirik, L. Epstein, S.V. Sevastianov, A.P.A. Vestjens, and G.J. Woeginger. On-line and off-line approximation algorithms for vector covering problems. *Algorithmica*, 21:104–118, 1998.

[2] S.F. Assmann. Problems in discrete applied mathematics. Technical report, Doctoral Dissertation, Mathematics Department, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1983.

[3] S.F. Assmann, D.S. Johnson, D.J. Kleitman, and J.Y.-T. Leung. On a dual version of the one-dimensional bin packing problem. *J. Algorithms*, 5:502–525, 1984.

[4] A. Borodin and R. El-Yaniv. On randomization in online computations. In *Computational Complexity*, 1997.

[5] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[6] J. Csirik and J.B.G. Frenk. A dual version of bin packing. *Algorithms Review*, 1:87–95, 1990.

[7] J. Csirik, J.B.G. Frenk, G. Galambos, and A.H.G. Rinnooy Kan. Probabilistic analysis of algorithms for dual bin packing problems. *J. Algorithms*, 12:189–203, 1991.

[8] J. Csirik and V. Totik. On-line algorithms for a dual version of bin packing. *Discr. Appl. Math.*, 21:163–167, 1988.

[9] J. Csirik and G.J. Woeginger. On-line packing and covering problems. In *A. Fiat and G. J. Woeginger, editors,* Online Algorithms: The State of the Art, volume 1442 of *LNCS*, pages 154–177. Springer-Verlag, 1998.

[10] T. Gaizer. An algorithm for the 2d dual bin packing problem. Unpublished manuscript, University of Szeged, Hungary, 1989.

[11] Gerhard J. Woeginger and Guochuan Zhang. Optimal on-line algorithms for variable-sized bin covering. *Operations Research Letters*, 25:47–50, 1999.

[12] A.C.C. Yao. Probabilistic computations: Towards a unified measure of complexity. In *Proceedings of the 18th ACM Symposium on Theory of Computing*, pages 222–227, 1977.