Theoretical Computer Science 407 (2008) 231–241

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Bincoloring[☆]

Sven O. Krumke[a], Willem E. de Paepe[b], Jörg Rambau[c], Leen Stougie[d,e,*]

[a] Department of Mathematics, University of Kaiserslautern. Paul-Ehrlich-Str. 14, 67653 Kaiserslautern, Germany

[b] Capgemini Nederland B.V., Papendorpseweg 100, P.O. Box 2575, 3500GN Utrecht, The Netherlands

[c] Department of Mathematics, University of Bayreuth, 95440 Bayreuth, Germany

[d] Department of Mathematics and Computer Science, Technical University of Eindhoven, P.O. Box 513, 5600MB Eindhoven, The Netherlands

[e] Centre for Mathematics and Computer Science (CWI), P.O. Box 94079, NL-1090 GB Amsterdam, The Netherlands

ARTICLE INFO

Article history:
Received 20 August 2007
Received in revised form 11 March 2008
Accepted 20 May 2008
Communicated by A. Fiat

Keywords:
Competitive analysis
NP-hardness

ABSTRACT

We introduce a new problem that was motivated by a (more complicated) problem arising in a robotized assembly environment. The *bin coloring problem* is to pack unit size colored items into bins, such that the maximum number of different colors per bin is minimized. Each bin has size $B \in \mathbb{N}$. The packing process is subject to the constraint that at any moment in time at most $q \in \mathbb{N}$ bins are partially filled. Moreover, bins may only be closed if they are filled completely. We settle the computational complexity of the problem and design an approximation algorithm for a natural version which gives a solution whose value is at most one greater than the optimal one.

We also investigate the existence of competitive online algorithms, which must pack each item without knowledge of any future items. We prove an upper bound of $3q - 1$ and a lower bound of $2q$ for the competitive ratio of a natural greedy-type algorithm, and show that surprisingly a trivial algorithm which uses only one open bin has a strictly better competitive ratio of $2q - 1$. Moreover, we show that any deterministic algorithm has a competitive ratio $\Omega(q)$ and that randomization does not improve this lower bound even when the adversary is oblivious.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

One of the commissioning departments in the distribution center of Herlitz PBS AG, Falkensee, a main distributor of office supplies in Europe, is devoted to greeting cards. The cards are stored in parallel shelving systems. Order pickers on automated guided vehicles collect the orders from the storage systems, following a circular course through the shelves. At the loading zone, which can hold q vehicles, each vehicle is logically "loaded" with B orders which arrive online. The goal is to avoid congestion among the vehicles (see [1] for details). Since the vehicles are unable to pass each other and the "speed" of a vehicle is correlated to the number of different stops it must make, this motivates assigning the orders to vehicles in such a way that the vehicles stop as few times as possible.

The above situation motivates the following *bin coloring problem* (BCP): one receives a finite sequence of unit size items $\sigma = r_1, r_2, \ldots$ where each item has a *color* $r_i \in \mathbb{N}$, and is asked to pack them into bins of size B. The goal is to pack the items into the bins "most uniformly", that is, to minimize the maximum number of different colors assigned to a bin. The packing

☆ Supported by MRT Network ADONET (MRTN-CT-2003-504438), the Dutch BSIK-BRICKS project, the Dutch mathematics cluster DIAMANT.

* Corresponding author at: Department of Mathematics and Computer Science, Technical University of Eindhoven, P.O. Box 513, 5600MB Eindhoven, The Netherlands. Tel.: +31 40 247 2299.

E-mail addresses: krumke@mathematik.uni-kl.de (S.O. Krumke), willem.de.paepe@capgemini.nl (W.E. de Paepe), joerg.rambau@uni-bayreuth.de (J. Rambau), leen@win.tue.nl (L. Stougie).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.05.022

process is subject to the constraint that at any moment in time, at most $q \in \mathbb{N}$ bins may be partially filled. Bins may only be closed if they are filled completely. (Notice that without these strict bounded space constraints the problem is trivial since in this case each item can be packed into a separate bin.)

In the *online version*, the *online bin coloring problem* (OlBcp), each item must be packed without knowledge of any future items. Trivially, any online algorithm for the OlBcp is B-competitive, where B denotes the size of the bins.

We study the online and offline bin coloring problem. For the offline problem Bcp we derive complexity and approximability results. For the online problem OlBcp we investigate which competitive ratios are achievable. Our results reveal a curiosity of competitive analysis: a truly stupid algorithm achieves essentially a (non-trivial) best possible competitive ratio for the problem whereas a seemingly reasonable algorithm performs provably worse in terms of competitive analysis.

This paper is organized as follows. In Section 2 we formally define the Bcp and OlBcp. Section 3 investigates the complexity of the offline problem Bcp and show that for a special case which is still NP-complete, the fitting bincoloring problem, there is an approximation algorithm with additive error of one. In Section 4 we describe and analyze the obvious algorithm greedyfit. In Section 5 we introduce and analyze the trivial algorithm onebin which surprisingly obtains a better competitive ratio than greedyfit. Sections 6 and 7 contain general lower bounds for deterministic and randomized algorithms.

2. Problem definition

We start by defining the problems under study.

Definition 1 (*(Online) Bin Coloring Problem*). In the *bin coloring problem* ($\text{Bcp}_{B,q}$) with parameters $B, q \in \mathbb{N}$ ($B, q \geq 2$), one is given a sequence $\sigma = r_1, \ldots, r_m$ of unit size items (requests), each with a color $r_i \in \mathbb{N}$, and is asked to pack them into bins with size B, that is, each bin can accommodate exactly B items. The packing is subject to the following constraints:

(1) The items must be packed according to the order of their appearance, that is, item i must be packed before item k for all $i < k$.
(2) At most q partially filled bins may be open to further items at any point in time during the packing process.
(3) A bin may only be closed if it is filled completely, i.e., if it has been assigned exactly B items.

The objective is to minimize the maximum number of different colors assigned to a bin.

In the online bin coloring problem ($\text{OlBcp}_{B,q}$), an online algorithm must pack each item r_i (irrevocably) without knowledge of requests r_k with $k > i$.

In the sequel it will be helpful to use the following view on the bins used to process an input sequence σ. Each open bin has an *index* x, where the number x satisfies $x \in \{1, \ldots, q\}$. Each time a bin with index x is closed (since it is filled completely) and a new bin is opened the new bin will also have index x. If no confusion can occur, we will refer to a bin with index x as *bin* x.

3. Offline-bin coloring

We first consider the general offline problem, and show that it is NP-complete. Then we consider the problem in a slightly different setting, where the number of items is exactly qB, and items have to be packed in exactly q bins. We show that even this problem is NP-complete.

Theorem 2. *The decision version of* $\text{Bcp}_{B,q}$ *is NP-complete.*

Proof. We provide a polynomial time reduction from 3-Partition which is well known to be NP-complete in the strong sense [2, Problem SP15]. An instance of 3-Partition is given by a set of numbers $S = \{a_1, \ldots, a_{3t}\}$ with $\sum_i a_i = tM$, $M/4 < a_i < M/2$ for all i, and $a_i \in \mathbb{Z}^+$ for all i and a bound $M \in \mathbb{Z}^+$. The question asked is whether S can be partitioned into t disjoint sets $S_1, \ldots, S_t$ such that $\sum_{i:a_i \in S_j} a_i = M$.

We can assume that $M \geq 4$. We construct an instance of $\text{Bcp}_{B,q}$ with $q = t$ bins of size $B = M$ as follows. The input sequence σ starts with a_1 items of color 1, followed by a_2 items of color 2 and so on until a_{3t} items of color $3t$. Then follow $3t$ additional items, each with a new color not used before. Thus, there are $6t$ colors overall. We will show that the instance of $\text{Bcp}_{B,q}$ constructed has a solution of cost at most 3 if and only if the original instance of 3-Partition has a yes-answer.

Suppose that there is a yes-answer to the instance of 3-Partition and let $S_1, \ldots, S_t$ be sets such that $\sum_{i:a_i \in S_j} a_i = B$. We put all items corresponding to $a_i \in S_j$ into bin j, for all $i = 1, \ldots, 3t$. After this, each bin contains exactly B items of at most three different colors. Consequently, all bins will be closed, and q empty bins can be used for placing the last $3t$ items. Assigning three of the remaining items to every bin will give a solution of cost 3.

If there is no yes-answer for 3-Partition, one of the following is true. There is at least one non-empty bin after the first tB items, or there has been a bin that contained items of at least four different colors. In the latter case, we cannot have a solution of cost at most 3, so we can assume that all bins contain items of at most three colors, and there is at least one non-empty bin. Since $M/4 < a_i < M/2$ for all i, we know that the non-empty bins that contain items of only one different color have at least $M/2 + 1 = B/2 + 1 \geq 3$ empty places, and that the non-empty bins that contain items of two different colors have at least two empty places. Non-empty bins containing items of three different colors must have at least one empty place, otherwise the bin would have been replaced by an empty bin. In order to establish a solution of cost at most 3, none of the non-empty bins can be completely filled (and replaced) by assigning one or more of the last $3t$ items to it, since these items all have a different color. As a consequence, all $3t$ items must be distributed over the t currently open bins. Since at least one bin is non-empty, at least one bin ends up with four or more different colors. □

Now consider the problem where the sequence σ consists of exactly qB items, and all items must be assigned to no more than q bins. We call this the *fitting bincoloring problem*. Observe that for the fitting bincoloring problem the order in which the items are given is irrelevant. First we show that deciding whether or not there exists a solution of cost at most 2 is NP-compete.

Theorem 3. *The decision version of the fitting bincoloring problem is NP-complete.*

Proof. Again we use a reduction from 3-Partition. Given any instance of 3-Partition with $\sum_{i=1}^{3t} a_i = tM$ and $M/4 < a_i < M/2$, we construct an instance of the fitting bincoloring problem with $q = 3t$ bins with capacity $B = M/2$. We will provide an input sequence σ such that the fitting bincoloring instance has a solution of cost at most two if and only if the instance of 3-Partition has a solution.

The sequence σ consists of a set α_i of $1 \leq M/2 - a_i = B - a_i \leq B/2 - 1$ items of color i, for $i = 1, \ldots, 3t$, and of a set β_i of $M = 2B$ items of color $3t + i$, for $i = 1, \ldots, t$.

Suppose that the instance of 3-Partition has a solution $S_1, \ldots, S_t$. We first assign all items of α_i to bin i. Hence, bin i has exactly a_i free space left. Now, for $j = 1, \ldots, t$ we consider set $S_j = \{a_{j_1}, a_{j_2}, a_{j_3}\}$ in the solution of 3-Partition and we distribute the items of β_j over bins j_1, j_2 and j_3. Since $a_{j_1} + a_{j_2} + a_{j_3} = M = 2B$ this completely fills up the corresponding bins. Hence, we obtain a solution of cost 2.

Conversely, suppose that there is a solution of cost at most 2. Each set α_i has at least one item, and at most $B/2 - 1$ items, $i = 1, \ldots, 3t$. Hence each of these sets can fill only less than half of a bin, hence in any solution of cost at most 2 they have to be allocated to $3t$ separate bins.

To reach a solution of cost 2, the items of every set $\beta_j, j = 1, \ldots, t$, have to be assigned to bins such that those bins are completely filled by these items together with the already present α-items. It is easy to see that each set β_j must be distributed over exactly 3 bins and that this implies that the instance of 3-Partition has a solution. □

The fitting bincoloring problem has been investigated in [5], where the authors provided a polynomial time algorithm which for every instance finds a solution of cost at most $\mathrm{OPT}(\sigma) + 1$. Here, we give a much simpler two-step strategy roundrobin which achieves the same performance of $\mathrm{OPT}(\sigma) + 1$ for the fitting bincoloring problem. We note that it is trivial to find the optimal solution if there exists a solution of value 1. Theorem 3 implies a lower bound of 3/2 on the approximation ratio.

Theorem 4. *Given an instance of the fitting bincoloring problem with input sequence σ, strategy roundrobin finds a solution of cost at most $\mathrm{OPT}(\sigma) + 1$.*

Proof. Let $C_1, \ldots, C_k$ be the color sets in the given instance. Observe that $\mathrm{OPT}(\sigma) \geq k/q$. Suppose that at the moment that Step 1 of roundrobin ends, already the items $C_1, \ldots, C_s$ have been assigned to bins. At this moment, by construction the following properties are satisfied:

Algorithm roundrobin

Step 1: Group the items by color, and sort the sets by increasing cardinality. Let $C_1, \ldots, C_k$ be the monochromatic sets such that $|C_1| \leq |C_2| \leq \cdots \leq |C_k|$.

Fill the bins with monochromatic sets in a round robin fashion, i.e., bin 1 receives the items of set C_1, bin 2 gets the items of set C_2, etc., until the current set does not fit completely into the current bin. Notice that at every moment, a set of items is assigned to the bin with the least number of items.

Call all bins with B items *closed*. The remaining bins are divided into *available* and *unavailable* bins. A bin with the same amount of colors as the current bin is termed *available*, otherwise the bin is *unavailable*. Notice that the unavailable bins contain one more color than the available bins.

Step 2: If the number of different colors of the items yet to be assigned is at most the number of available bins, assign the remaining monochromatic sets to the bins as follows. If there are unavailable bins remaining, each set will be used to fill up at least one unavailable bin, and the remaining items of this color are assigned to an available bin, that will be marked as unavailable from that point. The bin that is filled is marked as closed. Continue like this until all items are assigned. If there are no unavailable bins left, use an available bin instead of an unavailable bin.

If the number of remaining colors is greater than the number of available bins, fill the bins in a round robin fashion, starting from the bin with the least number of items, and assigning the remaining items of a color set to the next bin.

- No bin is able to accommodate any of the remaining sets $C_{s+1}, \ldots, C_k$ completely.
- No more than q different colors are remaining.
- The available bins have $\lfloor s/q \rfloor$ different colors, the unavailable bins have $\lfloor s/q \rfloor + 1$ different colors.
- $\mathrm{OPT}(\sigma) \geq \lfloor s/q \rfloor + 1$.

If the number of remaining colors is at most the number of available bins, at most two new colors are assigned to any originally available bin during Step 2, and at most one new color is assigned to any originally unavailable bin. The maximum number of colors in a bin is therefore at most $\lfloor s/q \rfloor + 2 \leq \mathsf{OPT}(\sigma) + 1$.

If the number of remaining colors is greater than the number of available bins, then the total number of different colors is at least $(\lfloor s/q \rfloor + 1)q$, and hence $\mathsf{OPT}(\sigma) \geq \lfloor s/q \rfloor + 2$. Each bin gets at most two new colors, resulting in a solution of at most $\lfloor s/q \rfloor + 3 \leq \mathsf{OPT}(\sigma) + 1$. □

4. The Algorithm greedyfit

We now turn to the online problem OLBCP. In this section we introduce a natural greedy-type strategy, which we call greedyfit, and show that the competitive ratio of this strategy is at most $3q$ but no smaller than $2q$ (provided the capacity B is sufficiently large).

Algorithm greedyfit

If upon the arrival of request r_i the color r_i is already contained in one of the currently open bins, say bin b, then put r_i into bin b. Otherwise put item r_i into a bin that contains the least number of different colors (which means opening a new bin if currently less than q bins are non-empty). Ties are broken arbitrarily.

The analysis of the competitive ratio of greedyfit is essentially via a pigeon-hole principle argument. We first show a lower bound on the number of bins that *any* algorithm can use to distribute the items in a contiguous subsequence and then relate this number to the number of colors in the input sequence.

Lemma 5. *Let $\sigma = r_1, \ldots, r_m$ be any request sequence for the $\text{OLBCP}_{B,q}$ and let $\sigma' = r_i, \ldots, r_{i+\ell}$ be any contiguous subsequence of σ. Then any algorithm packs the items of σ' into at most $2q + \lfloor (\ell - 2q)/B \rfloor$ different bins.*

Proof. Let ALG be any algorithm and let $b_1, \ldots, b_t$ be the set of open bins for ALG just prior to the arrival of the first item of σ'. Denote by $f(b_j) \in \{1, \ldots, B-1\}$ the empty space in bin b_j at that moment in time. To close an open bin b_j, ALG needs $f(b_j)$ items. Opening and closing an additional new bin needs B items. To achieve the maximum number of bins ($\geq 2q$), ALG must first close each open bin and put at least one item into each newly opened bin. From this moment in time, opening a new bin requires B new items. Thus, it follows that the maximum number of bins ALG can use is bounded from above as claimed in the lemma. □

Theorem 6. *Algorithm greedyfit is c-competitive for the $\text{OLBCP}_{B,q}$ with $c = \min\{2q + \lfloor (qB - 3q + 1)/B \rfloor, B\}$.*

Proof. Let σ be any request sequence and suppose $\mathsf{greedyfit}(\sigma) = w$. It suffices to consider the case $w \geq 2$. Let s be the smallest integer such that $\mathsf{greedyfit}(r_1, \ldots, r_{s-1}) = w - 1$ and $\mathsf{greedyfit}(r_1, \ldots, r_s) = w$. By the construction of greedyfit, after processing $r_1, \ldots, r_{s-1}$ each of the currently open bins must contain exactly $w - 1$ different colors. Moreover, since $w \geq 2$, after processing additionally request r_s, greedyfit has exactly q open bins (where, as an exception, we count here the bin where r_s is packed as open even if by this assignment it is just closed). Denote those bins by $b_1, \ldots, b_q$.

Let bin b_j be the bin among $b_1, \ldots, b_q$ that has been opened last by greedyfit. Let $r_{s'}$ be the first item that was assigned to b_j. Then, the subsequence $\sigma' = r_{s'}, \ldots, r_s$ consists of at most $qB - (q-1)$ items, since between $r_{s'}$ and r_s no bin is closed and at the moment $r_{s'}$ was processed, $q - 1$ bins already contained at least one item. Moreover, σ' contains items with at least w different colors. By Lemma 5 OPT distributes the items of σ' into at most $2q + \lfloor (qB - 3q + 1)/B \rfloor$ bins. Consequently,

$$\mathsf{OPT}(\sigma) \geq \frac{w}{2q + \lfloor (qB - 3q + 1)/B \rfloor},$$

which proves the theorem. □

Corollary 7. *Algorithm greedyfit is c-competitive for the $\text{OLBCP}_{B,q}$ with $c = \min\{3q - 1, B\}$.* □

We continue to prove a lower bound on the competitive ratio of algorithm greedyfit. This lower bound stated in the next theorem shows that the analysis of the previous theorem is tight up to a (small) constant factor.

Theorem 8. *greedyfit has a competitive ratio greater or equal to $2q$ for the $\text{OLBCP}_{B,q}$ if $B \geq 2q^3 - q^2 - q + 1$.*

The proof of Theorem 8 is divided into a couple of steps. Before we go into the technical details, we briefly sketch the idea and the key properties of the lower bound construction.

We construct a request sequence σ that consists of a finite number M of phases in each of which qB requests are given. The sequence is constructed in such a way that after each phase the adversary has q empty bins.

Each phase consists of two steps. In the first step q^2 items are presented, each with a new color which has not been used before. In the second step $qB - q^2$ items are presented, all with a color that has occurred before. We will show that we can choose the items given in Step 2 of every phase such that the following properties hold for the bins of greedyfit:

Property 1. *The bins with indices $1, \ldots, q-1$ are never closed.*

Property 2. *The bins with indices $1, \ldots, q-1$ contain only items of different colors.*

Property 3. *There is an assignment of the items of σ such that no bin contains items with more than q different colors.*

Property 4. *There is an $M \in \mathbb{N}$ such that during Phase M greedyfit assigns for the first time an item with a new color to a bin that already contains items with $2q^2 - 1$ different colors.*

We analyze the behavior of greedyfit by distinguishing between the items assigned to the bin with index q and the items assigned to bins with indices 1 through $q - 1$. In the sequel we let L_k denote the set of colors of the items assigned to bins $1, \ldots, q - 1$ and let R_k be the set of colors assigned to bin q during Step 1 of Phase k.

We now describe the construction of the request sequence. During Step 1 of Phase k, $k = 1, 2, \ldots$, we give q^2 items each with a different color which has not occurred before. During Step 1 there are items with $|R_k|$ different colors assigned to bin q by greedyfit.

For the moment, suppose that $|R_k| \geq q$ (we will justify this condition in the next lemma). We then partition the at most q^2 colors in $|R_k|$ into q disjoint non-empty sets $S_1, \ldots, S_q$. We give $qB - q^2 \geq 2q^2$ items with colors from $|R_k|$ such that the number of items with colors from S_j is $B - q$ for every j, and the last $|R_k|$ items all have a different color. The sequence stops, if for the first time greedyfit assigns an item with a new color to a bin which contains already $2q^2 - 1$ different colors.

In order for our construction to work, we must make sure that $|R_k| \geq q$ for all $k \geq 1$. This claim clearly holds for $k = 1$, since after Step 1 of Phase 1 each bin of greedyfit contains exactly q colors, so $|R_1| \geq q$. The next lemma shows that (among other things) the condition $|R_k| \geq q$ is invariably true.

Lemma 9. *We can construct a request sequence, divided into phases, such that for $k \geq 1$ the following statements are true, provided greedyfit does not assign an item with a new color to a bin which contains already $2q^2 - 1$ different colors.*

(i) *At the beginning of Phase k bin q of greedyfit contains exactly the colors from R_{k-1} (where $R_0 := \emptyset$).*
(ii) *Bin q is not closed by greedyfit before the end of Step 1 of Phase k.*
(iii) *After Step 1 of Phase k, each of the bins $1, \ldots, q - 1$ of greedyfit contains at least $|R_k| + |R_{k-1}| - 1$ different colors.*
(iv) *In Step 2 of Phase k greedyfit packs all items into bin q.*
(v) $|R_k| \geq q$.
(vi) *At the end of Phase k, bin q of greedyfit contains exactly $B - \sum_{j \leq k} |L_j|$ items, and this number is at least q^2.*

Proof. The proof is by induction on k. All claims are easily seen to be true for $k = 1$. Hence, in the inductive step we assume that statements (i)–(vi) are true for all phases $1, \ldots, k$ and we consider Phase $k + 1$.

(i) By the induction hypothesis (iv) all items from Step 2 presented in Phase k were packed into bin q by greedyfit. Since at the end of Phase k bin q contains at least $q^2 \geq |R_k|$ items (see induction hypothesis (vi)) and the last $|R_k|$ items presented in Phase k had different colors, it follows that at the beginning of Phase $k + 1$, bin q contains *at least* all colors from R_k. On the other hand, since all the $Bq - q^2 > B$ items from Step 2 of phase k were packed into bin q by greedyfit(induction hypothesis (iv)), this bin was closed during this process and consequently can only contain colors from R_k.

(ii) By the induction hypothesis (vi) at the end of phase k, the empty space in bin q is $\sum_{j \leq k} |L_j|$. Since there are exactly q^2 items presented in Step 1 of the phase, the claim follows if we can show that $\sum_{j \leq k} |L_j| \geq q^2$ at the beginning of phase $k + 1$: in that case, there is even enough space in bin q to accommodate *all* items given in Step 1 without filling it. We show that $|L_1| + |L_2| \geq q^2$ which implies that $|\sum_{j \leq k} L_j| \geq q^2$.

After Phase 1, each bin of greedyfit contains q colors, which yields $|L_1| = q(q - 1)$. By induction hypothesis (iv) for $k = 1$ all items presented in Step 2 of the first phase are packed into bin q by greedyfit. By induction, hypothesis (vi), at the end of Phase 1 bin q contains at least q^2 items. Since the last $|R_k|$ items presented in Step 2 of phase k have all different colors (and all of these are packed into bin q by induction hypothesis (iv)) we can conclude that at the beginning of Phase 2 bin q of greedyfit already contains q colors. Thus, in Step 1 of Phase 2 greedyfit again puts q items into each of its bins. At this point, the total number of distinct colors in the first $q - 1$ bins is at least $(q - 1)q + (q - 1)q = 2q^2 - 2q \geq q^2$ for $q > 1$, so that $|L_1| + |L_2| \geq q^2$. As noted above, this implies the claim.

(iii) By (ii), bin q is not closed before the end of Step 1. After Step 1 all colors from R_{k+1} are already in bin q by construction. Since by (i) before Step 1 also all colors from R_k were contained in bin q, it follows that bin q contains at least $|R_k| + |R_{k+1}|$ different colors at the end of Step 1.

By construction of greedyfit each of the bins $1, \ldots, q - 1$ must then contain at least $|R_k| + |R_{k+1}| - 1$ different colors.

(iv) When Step 2 starts, all colors from R_{k+1} are already in bin q by construction. Therefore, greedyfit will initially pack items with colors from R_{k+1} into bin q as long as this bin is not yet filled up. We have to show that after bin q has been closed the number of colors in any other bin is always larger than in bin q. This follows from (iii), since by (iii) each of the bins $1, \ldots, q - 1$ has at least $|R_k| + |R_{k+1}| - 1$ colors after Step 2 of Phase $k + 1$ and by the induction hypothesis (v) the estimate $|R_k| \geq q$ holds, which gives

$$|R_k| + |R_{k+1}| - 1 \geq |R_{k+1}| + q - 1 > |R_{k+1}|.$$

(v) At the beginning of Phase $k + 1$, bin q contains exactly $|R_k|$ colors by (i). By the induction hypothesis (iii) and (iv) each of the bins $1, \ldots, q - 1$ contains at least $|R_k| + |R_{k-1}| - 1 \geq |R_k|$ colors. Hence, at the beginning of Phase $k + 1$, the minimum number of colors in bins $1, \ldots, q - 1$ is at least the number of colors in bin q. It follows from the definition of greedyfit that during Step 1 of Phase $k + 1$, bin q is assigned at least the $q^2/q = q$ colors. In other words, $|R_{k+1}| \geq q$.

(vi) After Phase $k+1$, exactly $(k+1)qB$ items have been given. Moreover, after k phases bins 1 through $q-1$ contain exactly $\sum_{j\le k}|L_j|$ items because the items of Step 2 are always packed into bin q by greedyfit. Thus, the number of items in bin q of greedyfit equals

$$kqB - \sum_{j\le k}|L_j| \mod B \equiv \underbrace{B - \sum_{j\le k}|L_j|}_{<B} \mod B.$$

We show that $B - \sum_{j\le k}|L_j| \ge q^2$. This implies that $B - \sum_{j\le k}|L_j| \mod B = B - \sum_{j\le k}|L_j|$.

Since $k < M$ we know that each of the bins 1 through $q-1$ contains at most $2q^2-1$ colors. Thus, $\sum_{j\le k}|L_j| \le (2q^2-1)(q-1) = 2q^3 - 2q^2 - q + 1$. It follows from the assumption on B that $B - \sum_{j\le k}|L_j| \ge q^2$. □

At this this point we have shown that we can actually construct the sequence as suggested. Let us now bound the optimal offline cost and prove that the first three of the intended properties are indeed satisfied:

Lemma 10. *Properties 1–3 hold for the sequence constructed, as long as no bin from* greedyfit *contains* $2q^2$ *colors.*

Proof. Properties 1 and 2 are an immediate consequence of the fact that bins $1, \dots, q-1$ receive only bins in the first step of each phase (see Lemma 9).

We now address Property 3: in each phase, the adversary assigns the items of Step 1 such that every bin receives q items, and the items with colors in the color set S_j go to bin j. Clearly, the items in every bin have no more than q different colors. By construction of the sequence, the items given in Step 2 can be assigned to the bins of the adversary such that all bins are completely filled, and the number of different colors per bin does not increase. □

As a final step, we need to prove that there is a number $M \in \mathbb{N}$ such that after M phases there is a bin from greedyfit that contains items with $2q^2$ different colors (Property 4). We will do this by establishing the following lemma:

Lemma 11. *In every two subsequent Phases k and $k+1$, either $|L_k \cup L_{k+1}| > 0$ or bin q contains items with $2q^2$ different colors during one of the two phases.*

Proof. Suppose that there is a Phase k in which $|L_k| = 0$. This means that all q^2 items given in Step 1 are assigned to bin q ($|R_k| = q^2$). By Lemma 9(i), at the beginning of Phase $k+1$, bin q still contains q^2 different colors. If in Step 1 of Phase $k+1$ again all q^2 items are assigned to bin q, bin q contains items with $2q^2$ different colors (recall that bin q is never closed before the end of Step 1 by Lemma 9(ii)). If fewer than q^2 items are assigned to bin q then one of the other bins gets at least one item, and $|L_{k+1}| > 0$.

We can conclude from Lemma 11 that at least once every two phases the number of items in the bins 1 through $q-1$ grows. Since these bins are never closed (Property 1), and all items have a unique color (Property 2), after a finite number M of phases, one of the bins of greedyfit must contain items with $2q^2$ different colors. This completes the proof of the Theorem 8.

5. The trivial algorithm onebin

This section is devoted to arguably the simplest (and most trivial) algorithm for the OlBcp, which surprisingly has a better competitive ratio than greedyfit. Moreover, as we will see later this algorithm achieves essentially the best competitive ratio for the problem.

Algorithm onebin

The algorithm uses only at most one open bin at any point in time. The next item r_i is packed into the open bin. A new bin is opened only if the previous item has closed the bin by filling it up completely.

The proof of the upper bound on the competitive ratio of onebin is along the same lines as that of greedyfit.

Lemma 12. *Let $\sigma = r_1, \dots, r_m$ be any request sequence. Then for $i \ge 0$ any algorithm packs the items $r_{iB+1}, \dots, r_{(i+1)B}$ into at most* $\min\{2q-1, B\}$ *bins.*

Proof. It is trivial that the B items $r_{iB+1}, \dots, r_{(i+1)B}$ can be packed into at most B different bins. Hence we can assume that $2q-1 \le B$, which means $q \le (B-1)/2 \le B$.

Consider the subsequence $\sigma' = r_{iB+1}, \dots, r_{(i+1)B}$ of σ. Let ALG be any algorithm and suppose that just prior to the arrival of the first item of σ', algorithm ALG has t open bins. If $t = 0$, the claim of the lemma trivially follows, so we can assume for the rest of the proof that $t \ge 1$. Denote the open bins by $b_1, \dots, b_t$. Let $f(b_j) \in \{1, \dots, B-1\}$ be the number of empty places in bin b_j, $j = 1, \dots, t$. Notice that

$$\sum_{j=1}^{t} f(b_j) \equiv 0 \mod B. \tag{1}$$

Suppose that ALG uses at least $2q$ bins to distribute the items of σ'. By arguments similar to those given in Lemma 5, ALG can maximize the number of bins used only by closing each currently open bin and put at least one item into each of the newly opened bins. To obtain at least $2q$ bins at least $\sum_{j=1}^{t} f(b_j) + (q-t) + q$ items are required. Since σ' contains B items and $t \leq q$ it follows that

$$\sum_{j=1}^{t} f(b_j) + q \leq B. \tag{2}$$

Since by (1) the sum $\sum_{j=1}^{t} f(b_j)$ is a multiple of B and $q \geq 1$, the only possibility that the left hand side of (2) can be bounded from above by B is that $\sum_{j=1}^{t} f(b_j) = 0$. However, this is a contradiction to $f(b_j) \geq 1$ for $j = 1, \ldots, t$. □

As a consequence of the previous lemma we obtain the following bound on the competitive ratio of onebin.

Theorem 13. *Algorithm onebin is c-competitive for the* OLBCP$_{B,q}$ *where* $c = \min\{2q-1, B\}$.

Proof. Let $\sigma = r_1, \ldots, r_m$ be any request sequence for the OLBCP$_{B,q}$ and suppose that onebin$(\sigma) = w$. Let $\sigma' = r_{iB+1}, \ldots, r_{(i+1)B}$ of σ be the subsequence on which onebin gets w different colors. Clearly, σ' contains items with exactly w colors. By Lemma 12 OPT distributes the items of σ' into at most $\min\{2q-1, B\}$ different bins. Hence, one of those bins must be filled with at least $\frac{w}{\min\{2q-1,B\}}$ colors. □

The competitive ratio proved in the previous theorem is tight as the following example shows. Let $B \geq 2q-1$. First we give $(q-1)B$ items. The items have q different colors, every color but one occurs $B-1$ times, one color occurs only $q-1$ times. After this, in a second step q items with all the different colors used before are requested. Finally, in the third step $q-1$ items with new (previously unused) colors are given.

After the first $(q-1)B$ items by definition onebin has only empty bins. The adversary assigns all items of the same color to the same bin, using one color per bin. When the second set of of q items arrives, the adversary can now close $q-1$ bins, still using only one color per bin. onebin ends up with q different colors in its bin.

The adversary can assign every item given in the third step to an empty bin, thus still having only one different color per bin, while onebin puts these items in the bin where already q different colors where present.

6. A general lower bound for deterministic algorithms

In this section we prove a general lower bound on the competitive ratio of any deterministic online algorithm for the OLBCP. We establish a lemma which immediately leads to the desired lower bound but which is even more powerful. In particular, this lemma will allow us to derive essentially the same lower bound for randomized algorithms in Section 7.

In the sequel we will have to refer to the "state" of (the bins managed by) an algorithm ALG after processing a prefix of a request sequence σ. To this end we introduce the notion of a *$\mathcal{C}$-configuration*.

Definition 14 (*$\mathcal{C}$-Configuration*). Let $\mathcal{C}$ be a set of colors. A *$\mathcal{C}$-configuration* is a packing of items with colors from $\mathcal{C}$ into at most q bins. More formally, a $\mathcal{C}$-configuration can be defined as a mapping $K : \{1, \ldots, q\} \to \mathcal{S}_B$, where

$$\mathcal{S}_B = \{S : S \text{ is a multiset over } \mathcal{C} \text{ containing at most } B \text{ elements from } \mathcal{C}\}$$

with the interpretation that $K(j)$ is the multiset of colors contained in bin j. We omit the reference to the set $\mathcal{C}$ if it is clear from the context.

We are now ready to prove the key lemma which will be used in our lower bound constructions.

Lemma 15. *Let $B, q, s \in \mathbb{N}$ be numbers such that $s \geq 1$ and the inequality $B/q \geq s-1$ holds. There exists a finite set $\mathcal{C}$ of colors and a constant $L \in \mathbb{N}$ with the following property: For any deterministic algorithm ALG and any $\mathcal{C}$-configuration K there exists an input sequence $\sigma_{ALG,K}$ of* OLBCP$_{B,q}$ *such that*

(i) *The sequence $\sigma_{ALG,K}$ uses only colors from $\mathcal{C}$ and $|\sigma_{ALG,K}| \leq L$, that is, $\sigma_{ALG,K}$ consists of at most L requests.*
(ii) *If ALG starts with initial $\mathcal{C}$-configuration K then $ALG(\sigma_{ALG,K}) \geq (s-1)q$.*
(iii) *If OPT starts with the empty configuration (i.e., all bins are empty), then $OPT(\sigma_{ALG,K}) \leq s$. Additionally, OPT can process the sequence in such a way that at the end again the empty configuration is attained.*

Moreover, all of the above statements remain true even in the case that the online algorithm is allowed to use $q' \geq q$ bins instead of q (while the offline adversary still only uses q bins). In this case, the constants $|\mathcal{C}|$ and K depend only on q' but not on the particular algorithm ALG.

Proof. Let $\mathcal{C}$ be a set of $(s-1)^2q^2q'$ colors and ALG be any deterministic online algorithm which starts with some initial $\mathcal{C}$-configuration K. The construction of the request sequence $\sigma_{ALG,K}$ works in *phases*, where at the beginning of each phase the offline adversary has all bins empty.

During the run of the request sequence, a subset of the currently open bins of ALG will be *marked*. We will denote by P_k the subset of marked bins at the beginning of Phase k. Then, $P_1 = \emptyset$ and we are going to show that during some Phase M, one bin in P_M will contain at least $(s-1)q$ colors. In order to assure that this goal can in principle be achieved, we keep the invariant that each bin $b \in P_k$ has the property that the number of different colors in b plus the free space in b is at least $(s-1)q$. In other words, each bin $b \in P_k$ could potentially still be forced to contain at least $(s-1)q$ different colors. For technical reasons, P_k is only a subset of the bins with this property.

For bin j of ALG we denote by $n(j)$ the number of different colors currently in bin j and by $f(j)$ the space left in bin j. Then every bin $j \in P_k$ satisfies $n(j) + f(j) \geq (s-1)q$. By $\min P_k := \min_{j \in P_k} n(j)$ we denote the minimum number of colors in a bin from P_k.

The idea of the construction is the following (cf. Claim 17): We will force that in each phase either $|P_k|$ or $\min P_k$ increases. Hence, after a finite number of phases we must have $\min P_k \geq (s-1)q$. On the other hand, we will ensure that the optimal offline cost remains bounded by s during the whole process.

We now describe Phase k with $1 \leq k \leq q(s-1)q'$. The adversary selects a set of $(s-1)q$ new colors $C_k = \{c_1, \ldots, c_{(s-1)q}\}$ from $\mathcal{C}$ not used in any phase before and starts to present one item of each color in the order

$$c_1, c_2, \ldots, c_{(s-1)q}, c_1, c_2, \ldots, c_{(s-1)q}, c_1, c_2, \ldots \tag{3}$$

until one of the following cases appears:

Case 1 ALG puts an item into a bin $p \in P_k$.
In this case we let $Q := P_k \setminus \{ j \in P_k : n(j) < n(p) \}$, that is, we remove all bins from P_k which have less than $n(p)$ colors.
Notice that $\min_{j \in Q} n(j) > \min P_k$, since the number of different colors in bin p increases.

Case 2 ALG puts an item into some bin $j \notin P_k$ which satisfies

$$n(j) + f(j) \geq (s-1)q. \tag{4}$$

In this case we set $Q := P_k \cup \{j\}$ (that is, we tentatively add bin j to the set P_k).

Notice that after a finite number of requests one of these two cases must occur: Let $b_1, \ldots, b_t$ be the set of currently open bins of ALG. If ALG never puts an item into a bin from P_k then at some point all bins of $\{b_1, \ldots, b_t\} \setminus P_k$ are filled and a new bin, say bin j, must be opened by ALG by putting the new item into bin j. But at this moment bin j satisfies $n(j) = 1$, $f(j) = B - 1$ and hence $n(j) + f(j) = B \geq (s-1)q$ which gives (4).

Since the adversary started the phase with all bins empty and during the current phase we have given no more than $(s-1)q$ colors, the adversary can assign the items to bins such that no bin contains more than $s-1$ different colors (we will describe below how this is done precisely). Notice that due to our stopping criterions from above (Case 1 and Case 2) it might be the case that in fact we have presented less than $(s-1)q$ colors so far.

In the sequel we imagine that each currently open bin of the adversary has an index x, where $1 \leq x \leq q$. Let $\varphi\colon C_k \to \{1, \ldots, q\}$ be any mapping of the colors from C_k to the offline bin index such that $|\varphi^{-1}(\{x\})| \leq s-1$ for $j = 1, \ldots, q$. We imagine color c_r to "belong" to the bin with index $\varphi(c_r)$ even if no item of this color has been presented (yet). For those items presented already in Phase k, each item with color c_r goes into the currently open bin with index $\varphi(c_r)$. If there is no open bin with index $\varphi(c_r)$ when the item arrives a new bin with index $\varphi(c_r)$ is opened by the adversary to accommodate the item.

Our goal now is to clear all open offline bins so that we can start a new phase. During our clearing loop the offline bin with index x might be closed and replaced by an empty bin multiple times. Each time a bin with index x is replaced by an empty bin, the new bin will also have index x. The bin with index x receives a color not in $\varphi^{-1}(\{x\})$ at most once, ensuring that the optimum offline cost still remains bounded from above by s. The clearing loop works as follows:

(1) (Start of clearing loop iteration) Choose a color $c^* \in C_k$ which is not contained in any bin from Q. If there is no such color, goto the "good end" of the clearing loop (Step 4).

(2) Let $F \leq qB$ denote the current total empty space in the open offline bins. Present items of color c^* until one of the following things happens:

Case (a): At some point in time ALG puts the ℓth item with color c^* into a bin $j \in Q$ for some $1 \leq \ell < F$. Notice that the number of different colors in j increases. Let

$$Q' := Q \setminus \{b \in Q : n(b) < n(j)\},$$

in other words, we remove all bins b from Q which currently have less than $n(j)$ colors. This guarantees that

$$\min_{b \in Q'} n(b) > \min_{b \in Q} n(b) \geq \min P_k. \tag{5}$$

The adversary puts all ℓ items with color c^* into bins with index $\varphi(c^*)$. Notice that during this process the open bin with index $\varphi(c^*)$ might be filled up and replaced by a new empty bin with the same index.

Set $Q := Q'$ and go to the start of the next clearing loop iteration (Step 1). Notice that the number of colors from C_k which are not contained in Q decreases by one, but $\min_{b \in Q} n(b)$ increases.

Case (b): F items of color c^* have been presented, but ALG has not put any of these items into a bin from Q.

In this case, the offline adversary processes these items differently from Case (a): The F items of color c^* are used to fill up the exactly F empty places in all currently open offline bins. Since up to this point, each offline bin with index x had received colors only from the $s-1$ element set $\varphi^{-1}(\{x\})$, it follows that no offline bin has contained more than s different colors.

We close the clearing loop by proceeding as specified at the "standard end" (Step 3).

(3) (Standard end of clearing loop iteration)

In case we have reached this step, we are in the situation that all offline bins have been cleared (we can originate only from Case (b) above). We set $P_{k+1} := Q$ and end the clearing loop and the current Phase k.

(4) (Good end of clearing loop iteration)

Stop the current phase and issue additional requests such that all offline bins are closed without increasing the offline cost. After this, end the sequence.

We analyze the different possible endings of the clearing loop. First we show that in case of a "good end" we have successfully constructed a sufficiently bad sequence for ALG.

Claim 16. *If the clearing loop finishes with a "good end", then one bin in Q contains at least $(s-1)q$ different colors.*

Proof. If the clearing loop finishes with a "good end", then we have reached the point that all colors from C_k are contained in a bin from Q. Before the first iteration, exactly one color from C_k was contained in Q. The number of colors from C_k which are contained in bins from Q can only increase by one (which is in Case (a) above) if $\min_{b\in Q} n(b)$ increases. Hence, if all colors from C_k are contained in bins from Q, $\min_{b\in Q} n(b)$ must have increased $(s-1)q-1$ times, which implies $\min_{b\in Q} n(b) = (s-1)q$. In other words, one of ALG's bins in Q contains at least $(s-1)q$ different colors. □

What happens if the clearing loop finishes with a "standard end"?

Claim 17. *If the clearing loop of Phase k completes with a "standard end", then $\min P_{k+1} > \min P_k$ or $|P_{k+1}| > |P_k|$.*

Before we prove Claim 17, let us show how this claim implies the result of the lemma. Since the case $|P_{k+1}| > |P_k|$ can happen at most q' times, it follows that after at most q' phases, $\min P_k$ must increase. On the other hand, since $\min P_k$ never decreases by our construction and the offline cost always remains bounded from above by s, after at most $q(s-1)q'$ phases we must be in the situation that $\min P_k \geq (s-1)q$, which implies a "good end". Since in each phase at most $(s-1)q$ new colors are used, it follows that our initial set $\mathcal{C}$ of $(s-1)^2q^2q'$ colors suffices to construct the sequence $\sigma_{\mathsf{ALG},K}$. Clearly, the length of $\sigma_{\mathsf{ALG},K}$ can be bounded by a constant L independent of ALG and K.

Proof (*Proof of Claim 17*). Suppose that the sequence (3) given at the beginning of the phase was ended because Case 1 occurred, i.e., ALG put one of the new items into a bin from P_k. In this case $\min_{b\in Q} n(b) > \min P_k$. Since during the clearing loop $\min_{b\in Q} n(b)$ can never decrease and P_{k+1} is initialized with the result of Q at the "standard end" of the clearing loop, the claim follows.

The remaining case is that sequence (3) was ended because of a Case 2-situation. Then $|Q| = |P_k \cup \{j\}|$ for some $j \notin P_k$ and hence $|Q| > |P_k|$. During the clearing loop Q can only decrease in size if $\min_{i\in Q} n(i)$ increases. It follows that either $|P_{k+1}| \geq |P_k| + 1$ or $\min P_{k+1} > \min P_k$ which is what we claimed. □

This completes the proof of Lemma 15. □

As an immediate consequence of Lemma 15 we obtain the following lower bound result for the competitive ratio of any deterministic algorithm:

Theorem 18. *Let $B, q, s \in \mathbb{N}$ such that $s \geq 1$ and the inequality $B/q \geq s-1$ holds. No deterministic algorithm for $\textsc{OlBcp}_{B,q}$ can achieve a competitive ratio less than $\frac{s-1}{s} \cdot q$. Consequently, the competitive ratio of any deterministic algorithm for fixed B and q is at least $\left(1 - \frac{q}{B+q}\right) q$. In particular, for the general case with no restrictions on the relation of the capacity B to the number of bins q, there can be no deterministic algorithm for $\textsc{OlBcp}_{B,q}$ that achieves a competitive ratio less than q.*

All of the above claims remain valid, even if the online algorithm is allowed to use an arbitrary (but fixed) number $q' \geq q$ of open bins. □

7. A general lower bound for randomized algorithms

In this section we show lower bounds for the competitive ratio of any randomized algorithm against an oblivious adversary for the $\textsc{OlBcp}_{B,q}$. To this end we first recall Yao's Principle:

Theorem 19 (*Yao's Principle*). *Let $\{ ALG_y : y \in \mathcal{Y} \}$ denote the set of deterministic online algorithms for an online minimization problem. If $\bar{X}$ is a probability distribution over input sequences $\{ \sigma_x : x \in \mathcal{X} \}$ and $\bar{c} \geq 1$ is a real number such that*

$$\inf_{y\in\mathcal{Y}} \mathbb{E}_{\bar{X}} \left[ALG_y(\sigma_x) \right] \geq \bar{c}\, \mathbb{E}_{\bar{X}} \left[OPT(\sigma_x) \right], \tag{6}$$

then $\bar{c}$ is a lower bound on the competitive ratio of any randomized algorithm against an oblivious adversary. □

Theorem 20. *Let $B, q, s \in \mathbb{N}$ such that $s \geq 1$ and the inequality $B/q \geq s-1$ holds. Then no randomized algorithm for* OLBCP$_{B,q}$ *can achieve a competitive ratio less than $\frac{s-1}{s} \cdot q$ against an oblivious adversary.*

In particular for fixed B and q, the competitive ratio against an oblivious adversary is at least $\left(1 - \frac{q}{B+q}\right) q$.

All of the above claims remain valid, even if the online algorithm is allowed to use an arbitrary (but fixed) number $q' \geq q$ of open bins.

Proof. Let $\mathcal{A} := \{\, \mathsf{ALG}_y : y \in \mathcal{Y}\,\}$ be the set of deterministic algorithms for the OLBCP$_{B,q}$. We will show that there is a probability distribution X over a certain set of request sequences $\{\, \sigma_x : x \in \mathcal{X}\,\}$ such that for any algorithm $\mathsf{ALG}_y \in \mathcal{A}$

$$\mathbb{E}_X\left[\mathsf{ALG}_y(\sigma_x)\right] \geq (s-1)q,$$

and, moreover,

$$\mathbb{E}_X\left[\mathsf{OPT}(\sigma_x)\right] \leq s.$$

The claim of the theorem then follows by Yao's Principle.

Let us recall the essence of Lemma 15. The lemma establishes the existence of a finite color set $\mathcal{C}$ and a constant L such that for a fixed $\mathcal{C}$-configuration K, any deterministic algorithm can be "fooled" by one of at most $|\mathcal{C}|^L$ sequences. Since there are no more than $|\mathcal{C}|^{qB}$ configurations, a *fixed finite* set of at most $N := |\mathcal{C}|^{L+qB}$ sequences $\Sigma = \{\sigma_1, \ldots, \sigma_N\}$ suffices to "fool" *any* deterministic algorithm provided the initial configuration is known.

Let X be a probability distribution over the set of finite request sequences

$$\{\sigma_{i_1}, \sigma_{i_2}, \ldots, \sigma_{i_k} : k \in \mathbb{N}, 1 \leq i_j \leq N\}$$

such that σ_{i_j} is chosen from Σ uniformly and independently of all previous subsequences $\sigma_{i_1}, \ldots, \sigma_{i_{j-1}}$. We call subsequence σ_{i_k} the *kth phase*.

Let $\mathsf{ALG}_y \in \mathcal{A}$ be arbitrary. Define ϵ_k by

$$\epsilon_k := \Pr_X\left[\begin{array}{c}\mathsf{ALG}_y \text{ has at least one bin with} \\ \text{at least } (s-1)q \text{ colors during Phase } k\end{array}\right].$$

The probability that ALG_y has one bin with at least $(s-1)q$ colors on any given phase is at least $1/N$, whence $\epsilon_k \geq 1/N$ for all k. Let

$$p_k := \Pr_X\left[\mathsf{ALG}_y(\sigma_{i_1} \ldots \sigma_{i_{k-1}}\sigma_{i_k}) \geq (s-1)q\right].$$

Then the probabilities p_k satisfy the following recursion:

$$p_0 = 0 \tag{7}$$
$$p_k = p_{k-1} + (1 - p_{k-1})\epsilon_k. \tag{8}$$

The first term in (8) corresponds to the probability that ALG_y has already cost at least $(s-1)q$ after Phase $k-1$, the second term accounts for the probability that this is not the case but cost at least $(s-1)q$ is achieved in Phase k. By construction of X, these events are independent. Since $\epsilon_k \geq 1/N$ we get that

$$p_k \geq p_{k-1} + (1 - p_{k-1})/N. \tag{9}$$

It is easy to see that any sequence of real numbers $p_k \in [0, 1]$ satisfying (7) and (9) must converge to 1.

Hence, also the expected cost $\mathbb{E}_X\left[\mathsf{ALG}_y(\sigma_x)\right]$ converges to $(s-1)q$. On the other hand, the offline costs remain bounded by s by the choice of the σ_{i_j} according to Lemma 15. □

8. Remarks

We have studied the bin coloring problem, which was motivated by applications in a robotized assembly environment. The investigation of the online problem from a competitive analysis point of view revealed a number of oddities (see Table 1 for an overview of our results). A natural greedy-type strategy greedyfit achieves a competitive ratio strictly worse than arguably the most stupid algorithm (onebin). Moreover, no algorithm can be substantially better than the trivial strategy (onebin). Even more surprising, neither randomization nor "resource augmentation" helps to overcome the $\Omega(q)$ lower bound on the competitive ratio. This is in contrast to [4,6] where the concept of resource augmentation was applied successfully to scheduling problems. Intuitively, the strategy greedyfit should perform well "on average" (which was confirmed in preliminary experiments with random data, see [3]).

An open problem remains the existence of a deterministic (or randomized) algorithm which achieves a competitive ratio of q (matching the lower bound of Theorems 18 and 20). However, the most challenging issue raised by our work seems to be an investigation of OLBCP from an average-case analysis point of view.

Table 1
Results for the OLBCP

Problem	Competitive ratios	Lower bounds
OLBCP	onebin: $\min\{2q-1, B\}$ (Theorem 13) greedyfit: $\min\{3q, B\}$ (Corollary 7)	Deterministic algorithms: $\left(1-\frac{q}{B+q}\right)q$ (Theorem 18) Randomized algorithms: $\left(1-\frac{q}{B+q}\right)q$ (Theorem 20)

References

[1] N. Ascheuer, M. Grötschel, S.O. Krumke, J. Rambau, Combinatorial online optimization, in: Proceedings of the International Conference of Operations Research, OR'98, Springer, 1998, pp. 21–37.
[2] M.R. Garey, D.S. Johnson, Computers and Intractability (A Guide to the Theory of NP-Completeness), W.H. Freeman and Company, New York, 1979.
[3] A. Höft, Online-optimierung einer halbautomatischen glückwunschkarten kommissionierungsanlage, Diplomarbeit, Technische Universität, Berlin, 2001.
[4] B. Kalyanasundaram, K. Pruhs, Speed is as powerful as clairvoyance, J. ACM 47 (4) (2000) 617–643.
[5] M. Lin, Z. Lin, J. Xu, Almost optimal solutions for bin coloring problems, in: Proceedings of the 16th International Symposium on Algorithms and Computation, in: Lecture Notes in Computer Science, vol. 3827, 2005, pp. 82–91.
[6] C.A. Phillips, C. Stein, E. Torng, J. Wein, Optimal time-critical scheduling via resource augmentation, Algorithmica 32 (2) (2002) 163–200.