Variable sized online interval coloring with bandwidth

Leah Epstein* Thomas Erlebach † Asaf Levin ‡

Abstract

We consider online coloring of intervals with bandwidth in a setting where colors have variable capacities. Whenever the algorithm opens a new color, it must choose the capacity for that color and cannot change it later. The goal is to minimize the total capacity of all the colors used. We consider the bounded model, where all capacities must be chosen in the range $(0, 1]$, and the unbounded model, where the algorithm may use colors of any positive capacity. For the absolute competitive ratio, we give an upper bound of 14 and a lower bound of 4.59 for the bounded model, and an upper bound of 4 and a matching lower bound of 4 for the unbounded model. We also consider the offline version of these problems and show that whereas the unbounded model is polynomially solvable, the bounded model is NP-hard in the strong sense and admits a 3.6-approximation algorithm.

1 Introduction

Online interval coloring received much attention recently. In the basic problem, the nodes of an interval graph arrive online, one by one, together with the interval representation. The goal is to find a proper vertex coloring (i.e., each pair of adjacent vertices, i.e. intersecting intervals, are assigned distinct colors) with a minimum number of colors. The coloring has to be determined online, i.e., each new interval must be assigned a color upon arrival.

A study of this standard problem was done by Kierstead and Trotter [KT81]. They constructed an online algorithm which uses at most $3\omega - 2$ colors where ω is the maximum clique size of the interval graph. They also presented a matching lower bound of $3\omega - 2$ on the number of colors in a coloring of an arbitrary online algorithm. Note that the chromatic number of interval graphs equals the size of a maximum clique, which is equivalent in the case of interval graphs to the largest number of intervals that intersect any point (see [JT95, Gol80]). Many papers studied the competitive ratio of First-Fit for this problem [Kie88, KQ95, PRV04, CS88]. The latter paper shows that the competitive ratio of First-Fit is strictly worse than the competitive ratio of the algorithm of [KT81].

Adamy and Erlebach [AE03] introduced the interval coloring with bandwidth problem and presented a 195-competitive algorithm. In this problem each interval has a bandwidth requirement in $(0, 1]$. The intervals are to be colored so that at each point, the sum of bandwidths of intervals colored by a certain color does not exceed 1. This problem was studied also in [Nar04], giving an improved competitive ratio of 10, and in [EL05a], showing a lower bound of 3.2609 for this model.

We study a variant of this problem, where colors are not necessarily of capacity 1 as in [AE03]. The input arrives as in this model, however, an algorithm may use colors of arbitrary capacity. In an online environment, the capacity of a color is determined when the color is first used. The coloring is valid if for every color a that is used with capacity C_a, at each point the sum of bandwidths of intervals colored by a does not exceed C_a. The cost of a coloring is the sum of the capacities of the colors used.

*Department of Mathematics, University of Haifa, 31905 Haifa, Israel. lea@math.haifa.ac.il.

†Department of Computer Science, University of Leicester, University Road, Leicester LE1 7RH, United Kingdom. te17@mcs.le.ac.uk.

‡Department of Statistics, The Hebrew University, Jerusalem, Israel. levinas@mscc.huji.ac.il.

We study two models. In the *Unbounded Model*, there is no restriction on the capacities of colors. In the *Bounded Model*, the capacities cannot exceed the value 1.

The interval coloring problem with bandwidth of Adamy and Erlebach [AE03] is a generalization of the well known bin packing problem (see e.g. [Ull71, dlVL81, KK82, LL85, CW98, CGJ97, Sei02]). In that problem, items of size in $(0, 1]$ are to be partitioned into subsets of sum not exceeding 1. These subsets are called bins. In the online problem the items are assigned one by one to bins. If all input intervals are intersecting, we get an input of the bin packing problem, where bins correspond to colors.

Our problem is related to variable sized bin packing (see [Mur87, FL86, Csi89, Sei01, SvSE03]), but does not generalize it. In the bin packing problem, allowing the usage of bins of any size (even if the sizes are bounded by 1) leads to a simple 1-competitive algorithm, which assigns every item a bin of the same size. In the variable sized bin packing problem, a set of allowed bin sizes is set in advance, and the algorithm can only use bins of this fixed set of sizes.

As mentioned in [AE03], the interval coloring problem with bandwidth arises in many applications. Most of the applications come from the field of communication networks. Consider a network with a line topology that consists of links, where each link has channels of constant capacity. A connection request is from one network node a to another node b and has a bandwidth associated with it. The set of requests assigned to a channel must not exceed the capacity of the channel on any of the links on the path $[a, b]$. The goal is to minimize the number of channels (colors) used. In our problem, we can choose the capacity of the channel, and therefore we pay a cost proportional to the capacity of the channel, rather than a fixed cost, that is charged in the case of unit capacity channels. A connection request from a to b corresponds to an interval $[a, b]$ with the respective bandwidth requirement and the goal is to minimize the sum of capacities of the channels used to serve all requests. In our model, we allow different capacities since not all channels are necessarily identical.

Another important application comes from scheduling. A requested job has a starting time, a duration, and a resource requirement during its execution. Jobs (intervals) arrive online and must be assigned to a machine (color) immediately. It is possible to pick a machine of any capability, which is fixed when the machine is ordered. The cost of the machine is proportional to its resource capacity. The objective is to minimize the sum of the costs of the machines used.

For an algorithm $\mathcal{A}$, we denote its cost by $\mathcal{A}$ as well. The cost of an optimal offline algorithm that knows the complete sequence of intervals is denoted by OPT. We consider the absolute competitive ratio and the absolute approximation ratio criteria. For an online algorithm we use the term competitive ratio whereas for an offline algorithm we use the term approximation ratio. The competitive ratio of $\mathcal{A}$ is the infimum $\mathcal{R}$ such that for any input, $\mathcal{A} \leq \mathcal{R} \cdot \mathrm{OPT}$. If the competitive ratio of an online algorithm is at most $\mathcal{C}$ we say that it is $\mathcal{C}$-competitive. If an algorithm has an unbounded competitive ratio, we say that it is not competitive. The approximation ratio of a polynomial time offline algorithm is defined similarly to be the infimum $\mathcal{R}$ such that for any input, $\mathcal{A} \leq \mathcal{R} \cdot \mathrm{OPT}$. If the approximation ratio of a polynomial time offline algorithm is at most $\mathcal{R}$ we say that it is an $\mathcal{R}$-approximation algorithm.

We first consider the online problem. We give tight bounds for the unbounded model, showing that the competitive ratio achieved by applying doubling is 4, and this is best possible. For the bounded model, we show that an adaptation of the algorithm in [Nar04] combined with doubling is 14-competitive. We prove that no algorithm has competitive ratio better than 4.59.

We further show that the offline unbounded problem can be solved using a simple polynomial algorithm, while the bounded problem is NP-hard in the strong sense. For that problem we design an approximation algorithm with ratio $\frac{18}{5} = 3.6$.

2 Preliminaries

The following $KT_{\ell b}$ algorithm for the online interval coloring with bandwidth problem was studied by Epstein and Levy [EL05a, EL05b] (see also [Nar04]). We are given an upper bound b on the maximum bandwidth request. We are also given a value of a parameter ℓ. The algorithm partitions the requests into classes and then colors each class using the First-Fit algorithm. The partition of the requests is performed online so that a request j is allocated to class m, where m is the minimum value so that the maximum load of the requests that were allocated to classes $1, 2, \ldots, m$ with the additional new request is at most $m\ell$. For an interval v_i that was allocated to class m a *critical point of* v_i is a point q in v_i so that the set of all the intervals that were allocated to classes $1, 2, \ldots, m-1$ prior to the arrival of v_i, together with v_i, has total load strictly larger than $(m-1)\ell$ in q (i.e., q prevents the allocation of v_i to class $m-1$). They proved the following lemmas.

Lemma 1 *Given an interval v_i that was allocated class m. For the set A_m of intervals that were allocated to class m, and for every critical point q of v_i the total load of A_m in q is at most $b + \ell$. If all intervals have the same bandwidth b, and ℓ is divisible by b, this total load is at most ℓ.*

Lemma 2 *For every m, the set A_m of intervals that were allocated to class m has a maximum load of at most $2(b + \ell)$. If all intervals have the same bandwidth, b, and ℓ is divisible by b, the set A_m of intervals that were allocated to class m has a maximum load of at most 2ℓ.*

Lemma 3 *The number of classes used by the algorithm is at most $\lceil \frac{\omega^*}{\ell} \rceil$, where ω^* is the maximum load.*

3 Online Algorithms

3.1 The unbounded model

Our algorithm for the unbounded model simply uses standard doubling (see [BKM+92, AAF+97]). I.e., we keep a current "guess" of the maximum load of the complete sequence, which is actually a lower bound on the load, and a single active color. On the arrival of the first interval, we initialize the guess to be the smallest (negative) power of 2 that is not larger than the bandwidth requirement of the interval. We open the first color with capacity which is twice the guess. Each time an interval arrives we color it with the active (i.e., last opened) color if possible. If a new interval arrives that cannot be colored with the active color, this means that the maximum load is at least twice larger than the current guess. We therefore update the guess to equal twice the current guess, and open a new color with its capacity equal to twice the new value of the guess. Repeat this process until the interval can be colored with the most recently opened color. This color becomes active.

Theorem 4 *The competitive ratio of the above algorithm is 4.*

Proof. If there is a single color used by the algorithm, then its capacity is at most twice the largest load, and the competitive ratio is bounded by 2. Otherwise, consider the last time a new color was opened by the algorithm. The value L that is the current guess of the maximum load at this time is a lower bound on OPT. The new color has capacity $2L$, and since each time a new color is opened its capacity is at least twice the previous capacity, we conclude that the total cost of the algorithm is at most $2L + L + \frac{L}{2} + \cdots + \frac{L}{2^i} + \cdots \leq 4L \leq 4\text{OPT}$. ■

Given a non-negative small value $0 < \varepsilon < \frac{1}{6}$, we next describe a modified procedure whose asymptotic competitive ratio is $2 + \varepsilon$. The algorithm runs the $KT_{\ell b}$ algorithm with "unit" capacity that is set to $\frac{1}{\varepsilon}$. In order to use the algorithm with unit capacities, we multiply the bandwidth of all input intervals by ε. In this way we get $b = \varepsilon$ and therefore we can use $\ell = \frac{1}{2} - \varepsilon$, so that each class of the algorithm can be packed using one color. The algorithm has the following performance guarantee:

Theorem 5 *There is an online algorithm that for each input sequence provides a solution with cost at most* $(2+\varepsilon)\mathrm{OPT} + O(\frac{1}{\varepsilon})$.

Proof. By Lemma 3, the number of colors that our algorithm uses is at most $\lceil \frac{\omega^*}{\ell} \rceil$. Each of them costs $\frac{1}{\varepsilon}$, and 1 after scaling, where ω^* is the maximum load of the scaled input. Note that $\omega^* \leq \mathrm{OPT} \cdot \varepsilon$ (due to the scaling) and therefore the cost of the solution of the algorithm is at most

$$\left\lceil \frac{\omega^*}{\ell} \right\rceil \cdot \frac{1}{\varepsilon} \quad \leq \quad \left(\frac{\varepsilon \mathrm{OPT}}{\ell} + 1 \right) \cdot \frac{1}{\varepsilon} = \frac{\mathrm{OPT}}{\frac{1}{2} - \varepsilon} + \frac{1}{\varepsilon} = \frac{2 \cdot \mathrm{OPT}}{1 - 2\varepsilon} + \frac{1}{\varepsilon} \leq 2(1+3\varepsilon) \cdot \mathrm{OPT} + \frac{1}{\varepsilon},$$

where the last inequality holds for $\varepsilon < \frac{1}{6}$. By scaling ε before the application of the algorithm we obtain an online algorithm that uses colors of total cost at most $(2+\varepsilon)\mathrm{OPT} + \frac{6}{\varepsilon}$ as claimed. ■

3.2 The bounded model

Our algorithm for this case is the following adaptation of the algorithm of Narayanaswamy [Nar04] for the online interval coloring problem with bandwidth. We partition the requests into three groups. *Large requests* are requests with bandwidth in the interval $\left(\frac{1}{2}, 1\right]$, *medium requests* are requests with bandwidth in the interval $\left(\frac{1}{4}, \frac{1}{2}\right]$, and *small requests* are requests with bandwidth at most $\frac{1}{4}$. We use disjoint colors for coloring requests of distinct groups. Our algorithm is different from the algorithm of [Nar04] mainly in the procedure for coloring the small requests.

For packing large requests we use unit capacity colors, and pack these requests using Kierstead and Trotter's algorithm [KT81] for online interval coloring (without bandwidth). This is equivalent to using the algorithm in Section 2 with $\ell = 1$ and ignoring bandwidth requirements. In this case the total load of a class is at most two requests at each point, and as explained in [KT81], each class requires at most three colors.

Lemma 6 *The total cost of the colors used by the large requests is at most* $6 \cdot \mathrm{OPT}$.

Proof. Denote by L the maximum number of large requests that have a common point, i.e., the largest clique of large requests consists of L large requests. Note that the total load of the large requests is larger than $\frac{L}{2}$. Hence $\mathrm{OPT} > \frac{L}{2}$. The algorithm uses three colors, each of unit capacity, to pack each class of Kierstead and Trotter's algorithm [KT81]. The number of classes used by the algorithm is L, and therefore the cost of the colors used by the large requests is at most $3L$, and therefore at most $6 \cdot \mathrm{OPT}$. ■

For packing medium requests we again use unit capacity colors, and pack these requests using the algorithm in Section 2, giving each interval bandwidth of $\frac{1}{2}$. This is similar to using Kierstead and Trotter's algorithm for online interval coloring (without bandwidth). Each class is packed using one color (and not three colors). This packing of each class is feasible by Lemma 2, since we use $b = \ell = \frac{1}{2}$

Lemma 7 *The total cost of the colors used by the medium requests is at most* $4 \cdot \mathrm{OPT}$.

Proof. Denote by L the maximum number of medium requests that have a common point (there is a clique of L medium requests). Note that the total load of the medium requests is at least $\frac{L}{4}$. Hence $\mathrm{OPT} \geq \frac{L}{4}$. The algorithm uses one unit capacity color to pack each class of Kierstead and Trotter's algorithm [KT81]. The number of classes used by the algorithm is L, and therefore the cost of the colors used by the medium requests is at most L, and therefore at most $4 \cdot \mathrm{OPT}$. ■

It remains to describe the packing of the small requests. We partition the small requests into type 1 requests and type 2 requests. A *type 1* request is a request such that upon its arrival, for each point within the request the total load of previously presented type 1 requests, plus the load of the new request, is at most $\frac{1}{2}$. A *type 2* (small) request is a small request that is not a type 1 request.

We use separate sets of colors for type 1 small requests and type 2 small requests. The type 1 small requests are packed using the doubling procedure (described in the unbounded model). Recall that in that procedure, the capacity of each color that we use is an integer power of 2. Therefore, the last opened color that we use for small requests of type 1 has a capacity of at most $\frac{1}{2}$.

The packing of type 2 small requests uses only colors with unit capacity and is carried out by applying algorithm $KT_{\ell b}$ for $\ell = \frac{1}{4}$ and $b = \frac{1}{4}$. More precisely, we apply algorithm $KT_{\ell b}$ to all small requests, but the requests that are assigned to the first two classes by $KT_{\ell b}$ are actually the type 1 small requests that are handled as explained above.

The purpose of this partition into types is that if the load caused by the small intervals is very low, then opening a color of capacity 1 right away might be an overkill for the small intervals. Specifically, we want to show an absolute competitive ratio of 4, which would be impossible if a unit capacity color was opened immediately.

Lemma 8 *The total cost of the colors used by the small requests is at most* $4 \cdot \mathrm{OPT}$.

Proof. If there is no type 2 small request, then the claim holds since the doubling procedure is a 4-competitive algorithm. Thus, we can assume that there is at least one type 2 small request. Note that in this case all colors that we use to color type 1 small requests have a total cost that is at most 1. Consider the execution of the algorithm $KT_{\ell b}$ for $\ell = \frac{1}{4}$ and $b = \frac{1}{4}$ on the complete input (i.e., already starting at the first interval). All intervals of the first two classes that would have been opened by $KT_{\ell b}$ are colored in our algorithm by the set of colors which are given capacities smaller than 1. To see this last property note that by the definition of $KT_{\ell b}$, the first two classes of the algorithm contain only intervals whose total load is at most $2\ell = \frac{1}{2}$. All these intervals are by definition type 1 small requests. Therefore, if we denote by ω^* the maximum total load of the small requests, then $\mathrm{OPT} \geq \omega^*$ and the number of unit capacity colors that the algorithm uses in order to pack the type 2 small requests is at most $\lceil \frac{\omega^*}{\ell} \rceil - 2 \leq 4\omega^* + 1 - 2 = 4\omega^* - 1$. Since the total cost of the type 1 small requests is 1, we conclude that the algorithm packs the small requests using colors with total cost that is at most $4\omega^* \leq 4 \cdot \mathrm{OPT}$. ■

Using Lemmas 6, 7 and 8, we establish the following theorem.

Theorem 9 *There exists a* 14*-competitive online algorithm for the bounded model.*

4 Lower bounds

4.1 The unbounded model

We next show that the competitive ratio of our algorithm for the unbounded model is best possible. To prove the lower bound, we again apply methods similar to [BKM+92].

Theorem 10 *Any online algorithm for the unbounded model has a competitive ratio of at least* 4.

Proof. Before we construct the lower bound we note that we assume for ease of presentation that bandwidth requirements can be numbers larger than 1. Clearly, the unbounded model is equivalent to any model where the bandwidths are bounded by some constant (not necessarily 1). Before presenting the sequence, we can compute a bound on the largest bandwidth needed for the proof, and thus our lower bound satisfies the model.

Our construction of the lower bound for the unbounded model is based on instances in which OPT equals the maximum load, whereas the algorithm tries to guess an upper bound on the maximum load, and pays the sum of all its guesses. We consider input sequences with the following structure. The first interval is $[0,1]$ with a unit bandwidth request. Given an arbitrary prefix of intervals for which the algorithm opened the set of colors with capacities $c_1 \le c_2 \le \cdots \le c_k$ the next interval is disjoint to all the previous intervals with bandwidth request $c_k + \varepsilon$ for a sufficiently small value of ε. Then, the algorithm needs to open another color with capacity at least $c_{k+1} \ge c_k + \varepsilon$. Note that at this step OPT $= c_k + \varepsilon$ as all the intervals are disjoint and therefore they all fit into a common color with capacity $c_k + \varepsilon$, whereas the algorithm pays $\sum_{j=1}^{k+1} c_j$.

Given a fixed value of ρ that is strictly smaller than 4, we will show that if our input sequence is long enough an online algorithm cannot pay at each step k at most ρ times the cost of OPT at this step (the sequence can be stopped at any point, preventing all future intervals from arriving). Assume that this does not hold, and that there is a ρ-competitive online algorithm with $\rho = 4 - \delta$ for some $\delta > 0$. Denote this algorithm by $\mathcal{A}$. Assume that given the above input sequence for the value of ε that satisfies $\frac{1}{1+\varepsilon} = 1 - \delta^2$, $\mathcal{A}$ opens colors with capacities $c_1 < c_2 < \cdots < c_k < \cdots$. Then, since $\mathcal{A}$ is ρ-competitive the following inequalities must hold:

$$\sum_{j=1}^{k+1} c_j \le \rho(c_k + \varepsilon) \text{ and } c_1 \le \rho.$$

Let $r_{k+1} = 4 - \delta - \frac{\sum_{j=1}^{k} c_j}{c_k+\varepsilon}$, for $k \ge 1$. The inequality above implies $\frac{c_{k+1}}{c_k+\varepsilon} \le r_{k+1}$. Note that if $r_{k+1} < 1$, $\mathcal{A}$ cannot open a color of sufficient capacity in step $k+1$ without violating the assumption that its competitive ratio is ρ. We will show that the values r_{k+1} for $k = 1, 2, \ldots$ form a decreasing sequence so that r_{k+1} must be strictly less than 1 for some large enough value of k (depending only on ε). This is a contradiction to $r_{k+1} \ge 1$ and shows that such a sequence of c_k's cannot exist, hence no algorithm can achieve competitive ratio $4 - \delta$ for any $\delta > 0$.

First, we observe that $r_2 = 4 - \delta - \frac{c_1}{c_1+\varepsilon} \le 4 - \delta$. Next, we will show that $r_{k+2} \le r_{k+1}/(1+\gamma)$ for all $k \ge 1$ (as long as $r_{k+1} \ge 1$), where $\gamma > 0$ is a constant. Assuming that $r_{k+1} \le 4 - \delta$ was shown by induction, we can bound r_{k+2} as follows.

$$\begin{aligned}
r_{k+2} &= 4 - \delta - \frac{\sum_{j=1}^{k+1} c_j}{c_{k+1} + \varepsilon} = 4 - \delta - \frac{\sum_{j=1}^{k} c_j}{c_k + \varepsilon} \cdot \frac{c_k + \varepsilon}{c_{k+1} + \varepsilon} - \frac{c_{k+1}}{c_{k+1} + \varepsilon} \\
&= 4 - \delta - (4 - \delta - r_{k+1}) \cdot \frac{c_k + \varepsilon}{c_{k+1} + \varepsilon} - \frac{c_{k+1}}{c_{k+1} + \varepsilon} \\
&\le 4 - \delta - (4 - \delta - r_{k+1}) \cdot \frac{c_k + \varepsilon}{r_{k+1}(c_k + \varepsilon) + \varepsilon} - \frac{1}{1 + \varepsilon} \\
&\le 4 - \delta - (4 - \delta - r_{k+1}) \cdot \frac{c_k + \varepsilon}{(r_{k+1} + \varepsilon)(c_k + \varepsilon)} - \frac{1}{1 + \varepsilon} \\
&= 4 - \delta - (4 - \delta - r_{k+1}) \cdot \frac{1}{r_{k+1} + \varepsilon} - \frac{1}{1 + \varepsilon} \\
&= 4 - \delta - (4 - \delta) \cdot \frac{1}{r_{k+1} + \varepsilon} + \frac{r_{k+1}}{r_{k+1} + \varepsilon} - \frac{1}{1 + \varepsilon} \\
&\le 5 - \delta - \frac{1}{1 + \varepsilon} - \frac{4 - \delta}{r_{k+1} + \varepsilon} \le 5 - \delta - \frac{1}{1 + \varepsilon} - \frac{4 - \delta}{r_{k+1}(1 + \varepsilon)}
\end{aligned}$$

We claim that

$$5 - \delta - \frac{1}{1 + \varepsilon} - \frac{4 - \delta}{r_{k+1}(1 + \varepsilon)} \le \frac{r_{k+1}}{1 + \gamma} \tag{1}$$

where γ is chosen in such a way that $\frac{4}{1+\gamma} \ge 4 - \delta^2$ is satisfied. We see that (1) is equivalent to

$$\frac{1}{1 + \gamma} r_{k+1}^2 - (5 - \delta - \frac{1}{1 + \varepsilon}) r_{k+1} + \frac{4 - \delta}{1 + \varepsilon} \ge 0.$$

As this is a quadratic inequality of the form $ax^2 + bx + c \geq 0$ with $a > 0$, it suffices to show that the discriminant $b^2 - 4ac$ is negative. For $\delta < 0.1$ we can calculate as follows.

$$\begin{aligned} b^2 - 4ac &= (5 - \delta - \frac{1}{1+\varepsilon})^2 - \frac{4}{1+\gamma} \cdot \frac{4-\delta}{1+\varepsilon} \leq (4 - \delta + \delta^2)^2 - (4-\delta^2)(4-\delta)(1-\delta^2) \\ &= \delta^5 - 3\delta^4 - 7\delta^3 + 29\delta^2 - 4\delta \leq 0.0001\delta + 2.9\delta - 4\delta \leq -\delta < 0 \end{aligned}$$

Hence, (1) holds, and the proof is complete. ■

4.2 The bounded model

In order to construct the lower bound, we use as a black box the lower bound of Kierstead and Trotter [KT81] given originally for the standard online interval coloring problem. They designed for any integer k a lower bound sequence where the clique size is at most k, whereas any online algorithm is forced to use $3k - 2$ colors. In [EL05a] it was shown that this construction can be adapted to the case where the value k or bounds on it are known in advance to the algorithm.

Theorem 11 *Any online algorithm for the bounded model has a competitive ratio of at least* 4.5.

Proof. Let k be a large enough integer. We are going to have at most two such constructions, where there is no overlap between the intervals of the two constructions. Let $\varepsilon > 0$ be a small value, such that $P = \frac{1}{2\varepsilon}$ is an integer. We start with such a construction where all intervals have bandwidth $\frac{1}{2} + \varepsilon$. Since the largest capacity of a color can be 1, no two overlapping intervals can receive the same color, and therefore the algorithm is forced to use $3k - 2$ colors, whereas an optimal offline algorithm can use at most k colors, each of capacity $\frac{1}{2} + \varepsilon$.

The second construction will use intervals of bandwidth $\frac{1}{2} + j\varepsilon$ for some $2 \leq j \leq P$. In this construction as well the algorithm is forced to use $3k - 2$ colors of capacity at least $\frac{1}{2} + j\varepsilon$, whereas the construction is k-colorable. An optimal offline algorithm uses k colors of capacity $\frac{1}{2} + j\varepsilon$ each, and these colors are used to color all intervals of the first construction as well. Consider the $3k - 2$ colors with largest capacity opened by the algorithm for the first construction. Let s be the number of colors out of these colors whose capacity is strictly smaller than $\frac{1}{2} + j\varepsilon$. The algorithm has to open at least s new colors of capacity $\frac{1}{2} + j\varepsilon$.

Already in the first construction, the algorithm only needs to open colors whose capacities are in the set $\{\frac{1}{2} + \varepsilon, \frac{1}{2} + 2\varepsilon, \ldots, \frac{1}{2} + P\varepsilon = 1\}$. Consider only the $3k - 2$ colors of largest capacities that are opened for the first construction. Let X_j for $1 \leq j \leq P$ be the number of colors of capacity $\frac{1}{2} + j\varepsilon$.

Let C be the competitive ratio. The cost of the algorithm for the first construction is at least $\sum_{j=1}^{P} (\frac{1}{2} + j\varepsilon)X_j$. Note that according to the definition of the values X_j, $\sum_{j=1}^{P} X_j = 3k - 2$, therefore we can write this lower bound on the cost as $\frac{3k}{2} - 1 + \varepsilon \sum_{j=1}^{P} jX_j$. Since the optimal cost is $(\frac{1}{2} + \varepsilon)k$, we get $\frac{3k}{2} - 1 + \varepsilon \sum_{j=1}^{P} jX_j \leq C(\frac{1}{2} + \varepsilon)k$. This is equivalent to

$$\sum_{j=1}^{P} jX_j \leq CP(1 + 2\varepsilon)k - 3kP + 2P \; . \tag{2}$$

For every $2 \leq j \leq P$ we get a lower bound on the cost of the algorithm for the second construction of

$$
\begin{aligned}
& \sum_{i=1}^{P}(\frac{1}{2}+i\varepsilon)X_i + (3k-2-\sum_{i=j}^{P}X_i)(\frac{1}{2}+j\varepsilon) \\
= \;& \frac{3k-2}{2} + \varepsilon\sum_{i=1}^{P} iX_i + \frac{3k-2}{2} + j\varepsilon(3k-2) - \frac{1}{2}\sum_{i=j}^{P}X_i - j\varepsilon\sum_{i=j}^{P}X_i \\
= \;& (3k-2)(1+j\varepsilon) + \varepsilon\sum_{i=1}^{j-1} iX_i + \varepsilon\sum_{i=j}^{P}(i-j)X_i - \frac{1}{2}\sum_{i=j}^{P}X_i .
\end{aligned}
$$

Therefore,

$$(3k-2)(1+j\varepsilon) + \varepsilon\sum_{i=1}^{j-1} iX_i + \varepsilon\sum_{i=j}^{P}(i-j)X_i - \frac{1}{2}\sum_{i=j}^{P}X_i \le Ck(\frac{1}{2}+j\varepsilon) \; ,$$

or

$$2P(3k-2)(1+j\varepsilon) + \sum_{i=1}^{j-1} iX_i + \sum_{i=j}^{P}(i-j)X_i - P\sum_{i=j}^{P}X_i \le 2PCk(\frac{1}{2}+j\varepsilon) = PCk + jCk \; .$$

We get

$$\sum_{i=1}^{j-1} iX_i + \sum_{i=j}^{P}(i-j)X_i - P\sum_{i=j}^{P}X_i \le P(C-6)k + 4P + j(C-3)k + 2j \; . \tag{3}$$

For each $1 \le j \le P$, we multiply the inequality for j by the coefficient a_j, and add up the resulting inequalities. The values of the coefficients are $a_1 = \frac{P+1}{2}$ (for equation (2)), and for $j > 1$, $a_j = 1$.

Next, we compute the coefficient of each value X_i, $1 \le i \le P$, in the resulting inequality. Given a value X_i, its coefficient in the inequality (2) is i. Its coefficient in the inequality (3) for $j > i$ is i and for $j \le i$ is $i - j - P$. Therefore, we get

$$
\begin{aligned}
& \frac{P+1}{2}i + \sum_{j=2}^{i}(i-j-P) + \sum_{j=i+1}^{P} i = i\left(\frac{P+1}{2} + P - 1\right) - P(i-1) - \left(\frac{i(i+1)}{2} - 1\right) \\
& = \frac{iP}{2} + \frac{i}{2} + iP - i - Pi + P - \frac{i^2}{2} - \frac{i}{2} + 1 \ge (P-i)\cdot\frac{i+2}{2} \ge 0 \; .
\end{aligned}
$$

Therefore the left hand side of the resulting inequality is non-negative. Next, consider the right hand side. It is equal to

$$
\begin{aligned}
& \frac{P+1}{2}(CP(1+2\varepsilon)k - 3kP + 2P) \quad + \quad \sum_{j=2}^{P}(P(C-6)k + 4P + j(C-3)k + 2j) \\
= & \frac{P+1}{2}(CP(1+2\varepsilon)k - 3kP + 2P) \quad + \quad (P-1)(P(C-6)k + 4P) + ((C-3)k + 2)(\frac{P(P+1)}{2} - 1) \; .
\end{aligned}
$$

Letting k tend to infinity, we get the following inequality on C.

$$0 \le (P^2+P)(C(\frac{1}{2}+\varepsilon) - \frac{3}{2}) + (P^2-P)(C-6) + (C-3)\frac{P^2+P-2}{2} \; .$$

Next, we let P tend to infinity and get $0 \le \frac{C-3}{2} + (C-6) + \frac{C-3}{2} = 2C - 9$. This gives a lower bound of 4.5 on C. ■

Remark 12 *Running a linear program using Matlab for* $P = 400$ *we can get a lower bound of* 4.591 *on* C.

5 Offline problems

In this section we show that the unbounded model is polynomially solvable (offline) whereas the bounded model is NP-hard and we provide a 3.6-approximation algorithm for it.

5.1 The unbounded model

The unbounded model problem is clearly polynomially solvable. The algorithm computes the maximum load, and then opens a single color with capacity equal to the maximum load. Clearly, all the intervals can be colored using this color, and we obtain a feasible solution whose cost equals the maximum load, which is a lower bound on the optimal cost. Hence, we conclude the following:

Proposition 13 *The offline problem of the unbounded model is polynomially solvable.*

5.2 The bounded model

We first show that the resulting offline problem for the bounded model is NP-hard.

Theorem 14 *The offline problem of the bounded model is NP-hard in the strong sense.*

Proof. We show a reduction from the 3-Partition problem defined as follows (see problem [SP15] in [GJ79]): We are given a set of $3m$ positive numbers $s_1, s_2, \ldots, s_{3m}$ such that $\sum_{j=1}^{3m} s_j = mB$ and each s_i satisfies $\frac{B}{4} < s_i < \frac{B}{2}$. The goal is to find out whether there exists a partition of the numbers into m sets such that the sum of elements of each set is exactly B. The 3-Partition problem is known to be NP-hard in the strong sense.

Given such an instance of the 3-Partition problem we define an instance of the variable size interval coloring problem as follows: We are given a set of m intervals each with bandwidth request 1 and all of them are the interval $[0, 1]$. Moreover, we are given another set of $3m$ intervals $[2, 3]$, where the i-th interval of this set has a bandwidth request $\frac{s_i}{B}$. We claim that there is a feasible solution to this variable size interval coloring problem of cost at most m if and only if the 3-Partition instance is a YES instance.

To see this last claim first note that the variable size interval coloring problem must color with distinct colors the intervals of the first set (since we are considering the bounded model), and we can color with the same color intervals from the second set as long as their total bandwidth request is at most 1. I.e., there is a solution to the variable size interval coloring of cost at most m, if and only if the intervals of the second set (the intervals of $[2, 3]$) can be partitioned into m sets such that the total bandwidth request of the intervals of a set is at most 1. This means that the sum of elements of each such set is at most B, i.e., the 3-Partition instance is a YES instance. ■

Because of the fact that the bounded model problem is NP-hard, we turn our focus to designing an approximation algorithm for this problem. We define a *small request* to be a request with bandwidth that is at most $\frac{1}{2}$, and a *large request* to be a request whose bandwidth is strictly larger than $\frac{1}{2}$. Our algorithm uses disjoint sets of colors to color the small requests and the large requests.

For small requests we sort the intervals in non-decreasing order of their left end-point. Then, we use colors with maximum capacity 1 and color the intervals according to the First-Fit algorithm. After we color all the small requests, we compute the maximum load of the last color that is opened and we change its capacity to be this value of the maximum load.

Lemma 15 *The cost of colors that our algorithm uses to color the small requests is at most* $2 \cdot \mathrm{OPT}$.

Proof. Assume that the algorithm uses c colors to color the small requests and that the maximum load in the last color equals a. Then, the cost of the colors that our algorithm uses to color the small requests is $c-1+a$. Note that if $c=1$ then this cost is at most the maximum load (which equals the maximum load of the small requests) and therefore $a \leq \text{OPT}$. So we assume that $c \geq 2$. Consider the leftmost point in which the load of the last color is a, and denote this point by p. Then, p is the left end-point of an interval that our algorithm colors with the last color. Denote by I this interval, and recall that the bandwidth request of I is at most $\frac{1}{2}$. Since our algorithm decided not to place I in the first $c-1$ colors we conclude that the load of each of these colors in the point p is at least $\frac{1}{2}$ (this is so because the intervals are sorted from left to right). Therefore, the total load of all the small requests at point p is at least $\frac{c-1}{2}+a$. Therefore, $\text{OPT} \geq \frac{c-1}{2}+a \geq \frac{c-1+a}{2}$. Since the algorithm pays at most $c-1+a$, this cost is at most $2 \cdot \text{OPT}$. ■

It remains to consider the large requests. Before presenting our algorithm, we consider the following algorithm. We sort the large requests in non-decreasing order of their left end-point. Then, we use colors with capacity 1 and color the intervals according to the First-Fit algorithm. We note that using First-Fit minimizes the number of colors that are used to color the large requests (when the intervals are sorted) and since the capacity of each color is 1 whereas in the optimal solution the capacity of each color is at least $\frac{1}{2}$, we conclude that this algorithm uses colors with total cost of at most $2 \cdot \text{OPT}$.

Let $\varepsilon > 0$ be a given constant such that $k = \frac{1}{2\varepsilon} - 1$ is an integer to be selected afterwards. Our algorithm for the large requests computes $k+1$ solutions and picks the cheapest solution among these. The first solution is to pack all the large requests with a minimum number of unit capacity colors (using First-Fit on the sorted list of large requests). For each $j = 1, 2, \ldots, k$ we define $a_j = \frac{1}{2} + j\varepsilon$ and our $(j+1)$-th solution is constructed as follows. We partition the large requests into two classes: the first class consists of all large requests with bandwidth at most a_j, and the second class consists of all the remaining large requests. Each class is packed separately using its own set of colors. The capacity of the colors that are used for the first class is a_j, whereas the capacity of the colors that are used for the second class is 1. Each class is packed optimally using the minimum number of colors (using First-Fit on the sorted list of intervals from this class). We next show that the cheapest solution among the $k+1$ solutions has a cost of at most $\left(\frac{8}{5} + O(\varepsilon)\right) \cdot \text{OPT}$.

Lemma 16 *The cheapest solution among the $k+1$ solutions has a cost of at most $\left(\frac{8}{5-2\varepsilon}\right) \cdot \text{OPT}$.*

Proof. We prove that the algorithm colors the large requests with total cost of at most $\frac{8}{5-2\varepsilon} \cdot \text{OPT}$ and the approximation ratio of the algorithm is at most $\frac{8}{5-2\varepsilon}$. Let $a_0 = \frac{1}{2}$ and $a_{k+1} = 1$. Let ρ be the competitive ratio of the algorithm, we prove that $\rho \leq \frac{8}{5-2\varepsilon}$. Denote by X_j the number of colors that OPT opens with capacity in the interval $(a_j, a_{j+1}]$, for $j = 0, 1, 2, \ldots, k$. Then, $\text{OPT} \geq \sum_{j=0}^{k} a_j \cdot X_j$. We assume that the cheapest solution among the $k+1$ solutions costs at least $\rho \cdot \text{OPT}$.

Since two intersecting large requests cannot be colored by the same color in any solution, we can compute upper bounds on the number of colors of each capacity used by the algorithm in each one of the cases. Note that our first solution can pack all the large requests using at most $\sum_{j=0}^{k} X_j$ colors and therefore the cost of this solution is at most $\sum_{j=0}^{k} X_j$. Since we assume that the cheapest solution among the $k+1$ solutions costs at least $\rho \cdot \text{OPT}$, we conclude that $\sum_{j=0}^{k} X_j \geq \rho \cdot \text{OPT}$. Next, consider the $(j+1)$-th solution for $j \geq 1$. The intervals of the first class can be colored using at most $\sum_{i=0}^{k} X_i$ colors each with capacity a_j (since this amount of colors suffices to color all the large requests). The intervals of the second class can be colored using at most $\sum_{i=j}^{k} X_i$ unit capacity colors. Therefore, the cost of the $(j+1)$-th solution is at most $a_j \cdot \left(\sum_{i=0}^{k} X_i\right) + \sum_{i=j}^{k} X_i$. Since we assume that the cheapest solution among the $k+1$ solutions costs at least $\rho \cdot \text{OPT}$, we conclude that $a_j \cdot \left(\sum_{i=0}^{k} X_i\right) + \sum_{i=j}^{k} X_i \geq \rho \cdot \text{OPT}$.

We next consider the following set of inequalities (these inequalities hold by our assumption):

$$\mathrm{OPT} \geq \sum_{j=0}^{k} a_j \cdot X_j \tag{4}$$

$$\sum_{j=0}^{k} X_j \geq \rho \cdot \mathrm{OPT} \tag{5}$$

$$a_j \cdot \left(\sum_{i=0}^{k} X_i\right) + \sum_{i=j}^{k} X_i \geq \rho \cdot \mathrm{OPT} \qquad \forall j = 1, 2, \ldots, k. \tag{6}$$

We construct the following inequality: we multiply (5) by $y_0 = a_0 - \sum_{i=1}^{k}(a_i - a_{i-1}) \cdot a_i$, and for each $j = 1, 2, \ldots, k$ we multiply the j-th constraint of (6) by $y_j = a_j - a_{j-1} = \varepsilon$, and we add up all the resulting inequalities. The left hand side of the resulting inequality is exactly $\sum_{j=0}^{k} a_j \cdot X_j$. This is so because the coefficient of X_j in the resulting inequality is $y_0 + \sum_{i=1}^{j} y_i(a_i + 1) + \sum_{i=j+1}^{k} y_i a_i = a_0 - \sum_{i=1}^{k}(a_i - a_{i-1}) \cdot a_i + \sum_{i=1}^{k}(a_i - a_{i-1})a_i + \sum_{i=1}^{j}(a_i - a_{i-1}) = a_j$. By (4), we conclude that the left hand side of the resulting inequality is at most OPT. The right hand side of the resulting inequality is $\rho \cdot \mathrm{OPT} \cdot \sum_{i=0}^{k} y_i$. We note also that the coefficients y_j are non-negative. To see this last claim note that for $j \geq 1$, $y_j = \varepsilon > 0$ and for $j = 0$, $y_0 = a_0 - \sum_{i=1}^{k}(a_i - a_{i-1}) \cdot a_i = \frac{1}{2} - \sum_{i=1}^{k} \varepsilon \cdot \left(\frac{1}{2} + i\varepsilon\right) = \frac{1-k\varepsilon}{2} - \varepsilon^2 \sum_{i=1}^{k} i = \frac{1-k\varepsilon}{2} - \varepsilon^2 \cdot \frac{k(k+1)}{2} \geq \frac{1-\frac{1}{2\varepsilon}\varepsilon}{2} - \varepsilon^2 \cdot \frac{1}{8\varepsilon^2} = \frac{1}{8} > 0$. Therefore, the inequality $\mathrm{OPT} \geq \sum_{j=0}^{k} a_j \cdot X_j \geq \rho \cdot \mathrm{OPT} \cdot \sum_{i=0}^{k} y_i$ holds. Therefore,

$$\rho \leq \frac{1}{\sum_{i=0}^{k} y_i} = \frac{1}{\frac{1-k\varepsilon}{2} - \varepsilon^2 \cdot \frac{k(k+1)}{2} + k\varepsilon} = \frac{2}{1 + k\varepsilon - k(k+1)\varepsilon^2} = \frac{2}{1 + \frac{1}{2} - \varepsilon - (\frac{1}{4} - \frac{\varepsilon}{2})} = \frac{8}{5 - 2\varepsilon}.$$

This completes the proof. ■

By Lemma 16, we obtain a solution that colors the large requests with colors of total cost at most $\left(\frac{8}{5} + O(\varepsilon)\right) \cdot \mathrm{OPT}$. We would like to argue that by picking ε as an infinitesimally small positive number we obtain an $\frac{8}{5}$ approximation algorithm. However, picking such a value of ε will increase dramatically the time complexity of our algorithm. To avoid these bad consequences we note the following lemma.

Lemma 17 *There is a polynomial time algorithm that emulates the solution returned by our previous algorithm for infinitesimally small value of ε.*

Proof. Our previous algorithm constructs solutions by partitioning the set of large intervals into two classes, and then solving the minimum coloring of each class by itself. The cost of each color of each class is bounded by the largest bandwidth request of intervals of this class (and this bound can be better than the bound we used in the proof of Lemma 16). Since the partitioning into two classes is done by deciding upon a threshold value τ and all requests smaller than τ are assigned to the first class and all other requests are assigned to the second class, we conclude that even for infinitesimally small value of ε we have to consider at most n ways to partition the large intervals, where n is the total number of large intervals. Therefore, we can execute the algorithm for an infinitesimally small value of ε by computing at most n solutions, and this is a polynomial time algorithm whose running time is independent of ε. ■

The following corollary is a direct consequence of Lemmas 16 and 17.

Corollary 18 *There is a polynomial time algorithm that colors the large requests with colors of total cost at most $\frac{8}{5} \cdot \mathrm{OPT}$.*

We next summarize the results for small requests and large requests.

Theorem 19 *There is an approximation algorithm with ratio $\frac{18}{5} = 3.6$ for the variable sized interval coloring problem in the bounded model.*

References

[AAF+97] J. Aspnes, Y. Azar, A. Fiat, S. A. Plotkin, and O. Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *Journal of the ACM*, 44(3):486–504, 1997.

[AE03] U. Adamy and T. Erlebach. Online coloring of intervals with bandwidth. In *Proceedings of the First International Workshop on Approximation and Online Algorithms (WAOA'03)*, LNCS 2909, pages 1–12, 2003.

[BKM+92] S. K. Baruah, G. Koren, D. Mao, B. Mishra, A. Raghunathan, L. E. Rosier, D. Shasha, and F. Wang. On the competitiveness of on-line real-time task scheduling. *Real-Time Systems*, 4(2):125–144, 1992.

[CGJ97] E. G. Coffman, M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing: A survey. In D. Hochbaum, editor, *Approximation algorithms*. PWS Publishing Company, 1997.

[CS88] M. Chrobak and M. Ślusarek. On some packing problems relating to dynamical storage allocation. *RAIRO Journal on Information Theory and Applications*, 22:487–499, 1988.

[Csi89] J. Csirik. An online algorithm for variable-sized bin packing. *Acta Informatica*, 26:697–709, 1989.

[CW98] 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, LNCS 1442, pages 147–177, 1998.

[dlVL81] W. Fernandez de la Vega and G. S. Lueker. Bin packing can be solved within $1 + \varepsilon$ in linear time. *Combinatorica*, 1:349–355, 1981.

[EL05a] L. Epstein and M. Levy. Online interval coloring and variants. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, LNCS 3580, pages 602–613, 2005.

[EL05b] L. Epstein and M. Levy. Online interval coloring with packing constraints. In *Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science (MFCS'05)*, LNCS 3618, pages 295–307, 2005.

[FL86] D. K. Friesen and M. A. Langston. Variable sized bin packing. *SIAM Journal on Computing*, 15:222–230, 1986.

[GJ79] M. R. Garey and D. S. Johnson. *Computer and Intractability*. W. H. Freeman and Company, New York, 1979.

[Gol80] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, 1980.

[JT95] T. R. Jensen and B. Toft. *Graph coloring problems*. Wiley, 1995.

[Kie88] H. A. Kierstead. The linearity of first-fit coloring of interval graphs. *SIAM Journal on Discrete Mathematics*, 1(4):526–530, 1988.

[KK82] N. Karmarkar and R. M. Karp. An efficient approximation scheme for the one-dimensional bin-packing problem. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS'82)*, pages 312–320, 1982.

[KQ95] H. A. Kierstead and J. Qin. Coloring interval graphs with First-Fit. *SIAM Journal on Discrete Mathematics*, 8:47–57, 1995.

[KT81] H. A. Kierstead and W. T. Trotter. An extremal problem in recursive combinatorics. *Congressus Numerantium*, 33:143–153, 1981.

[LL85] C. C. Lee and D. T. Lee. A simple online bin packing algorithm. *Journal of the ACM*, 32(3):562–572, 1985.

[Mur87] F. D. Murgolo. An efficient approximation scheme for variable-sized bin packing. *SIAM J. Comput.*, 16(1):149–161, 1987.

[Nar04] N. S. Narayanaswamy. Dynamic storage allocation and online colouring interval graphs. In *Proceedings of the 10th Annual International Conference on Computing and Combinatorics (COCOON'04)*, LNCS 3106, pages 329–338, 2004.

[PRV04] S. V. Pemmaraju, R. Raman, and K. R. Varadarajan. Buffer minimization using max-coloring. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'04)*, pages 562–571, 2004.

[Sei01] S. S. Seiden. An optimal online algorithm for bounded space variable-sized bin packing. *SIAM Journal on Discrete Mathematics*, 14(4):458–470, 2001.

[Sei02] S. S. Seiden. On the online bin packing problem. *Journal of the ACM*, 49(5):640–671, 2002.

[SvSE03] S. S. Seiden, R. van Stee, and L. Epstein. New bounds for variable-sized online bin packing. *SIAM Journal on Computing*, 32(2):455–469, 2003.

[Ull71] J. D. Ullman. The performance of a memory allocation algorithm. Technical Report 100, Princeton University, Princeton, NJ, 1971.