Online Capacitated Interval Coloring

Leah Epstein[1], Thomas Erlebach[2], and Asaf Levin[3]

[1] Department of Mathematics, University of Haifa, 31905 Haifa, Israel,
`lea@math.haifa.ac.il`
[2] Department of Computer Science, University of Leicester, England,
`t.erlebach.mcs.le.ac.uk`
[3] Department of Statistics, The Hebrew University, Jerusalem, Israel,
`levinas@mscc.huji.ac.il`

Abstract. In the online capacitated interval coloring problem, a sequence of requests arrive online. Each of the requests is an interval $I_j \subseteq \{1, 2, \ldots, n\}$ with bandwidth b_j. Initially a vector of capacities $(c_1, c_2, \ldots, c_n)$ is given. Each color can support a set of requests such that the total bandwidth of intervals containing i is at most c_i. The goal is to color the requests using a minimum number of colors. We present a constant competitive algorithm for the case where the maximum bandwidth $b_{\max} = \max_j b_j$ is at most the minimum capacity $c_{\min} = \min_i c_i$. For the case $b_{\max} > c_{\min}$, we give an algorithm with competitive ratio $O(\log \frac{b_{\max}}{c_{\min}})$ and, using resource augmentation, a constant competitive algorithm. We also give a lower bound showing that constant competitive ratio cannot be achieved in this case without resource augmentation.

1 Introduction

Motivated by a routing problem in optical networks we consider the following problem. We are given a line network with links $1, 2, \ldots, n$ and a vector of base capacities $(c_1, c_2, \ldots, c_n)$. The requests arrive one by one, in an online fashion, and each request is identified by the interval of links that it uses $I_j = [s_j, t_j]$ where $1 \le s_j \le t_j \le n$. Moreover, the request I_j is associated with a bandwidth b_j that is the *bandwidth request* of I_j. Each time a request arrives, a color must be assigned to it before the next request is revealed. A restriction on the coloring is that the total bandwidth of all requests that are assigned a common color and contain link i is at most c_i. The goal is to use a minimum number of colors. Naturally, we assume $b_j \le c_i$ for all $i \in I_j$ (otherwise a feasible coloring would not exist). Without loss of generality we also assume (by scaling) that $\min_{i=1,2,\ldots,n} c_i = 1$.

As practical motivation of our study, consider an optical line network, where each color corresponds to a distinct frequency (this frequency is seen as a color as it is a frequency of light) in which the information flows. Different links along the line have different capacities, which are a function of intermediate equipment along the link (e.g., a link with an intermediate repeater may have reduced capacity for each color as a result of the repeater). Each request uses the same bandwidth on all links that this request contains. Moreover, requests arrive over time. As the number of distinct available frequencies is limited, minimizing the number of colors for a given sequence of requests is a natural objective. Changing the color allocation of a request causes a setup cost that we would like to avoid, and therefore we restrict ourselves to the online problem where once a request is allocated a color this color allocation cannot be changed.

From a theoretical point of view, the problem is interesting as it extends the previously studied case of uniform capacities ($c_i = 1$ for all i) to the setting with arbitrary capacities. For many problems of a similar flavor (both in the online and offline variants), the setting with arbitrary capacities is significantly more difficult to deal with than the uniform setting, and new techniques and ideas are often required. For example, a $\frac{3}{2}$-approximation algorithm for offline coloring of unit-bandwidth paths in trees with uniform edge capacities follows easily from the known results for the unit-capacity case [17], but nontrivial new techniques were needed to obtain a 4-approximation for the case with arbitrary capacities [6]. Similar observations can be made for the throughput version of such problems (i.e., maximizing the total bandwidth of requests that can be accepted with one available color). For example, the only known constant competitive algorithm for online throughput maximization in line or ring networks uses randomization and preemption and works only for the case of uniform edge capacities [2]. For the offline version of these problems, Chakrabarti et al. remark in [5] that most of the techniques that have been used for the uniform capacity case do not seem to extend to the case of arbitrary capacities.

In order to analyze our online algorithms for capacitated interval coloring, we use the common criterion of competitive analysis. 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 in advance is denoted by OPT. We consider the absolute competitive ratio that is defined as follows. The absolute 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 absolute competitive ratio of an online algorithm is at most $\mathcal{C}$ we say that the algorithm is $\mathcal{C}$-competitive.

The problem studied in this paper is a generalization of the classical online interval graph coloring problem. If all capacities are 1, all bandwidth requests are 1, and the number of links in the network is unbounded, we arrive at the standard interval coloring problem.

Coloring interval graphs has been intensively studied. Kierstead and Trotter [14] 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 [11]). Many papers studied the performance of First Fit for this problem [12, 13, 16, 7]. It is shown in [7] that the performance of First Fit is strictly worse than the one achieved by the algorithm of [14].

Generalizations of interval coloring received much attention recently. Adamy and Erlebach [1] introduced the interval coloring with bandwidth problem (which is also a special case of our problem). In this problem all capacities are 1 and each interval has a bandwidth requirement in $(0, 1]$. As in our problem, 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 the capacity, which is 1. This problem was studied also in [15, 8, 3]. The best competitive ratio known for that problem is 10 [15, 3]. A lower bound strictly higher than 3 was shown in [8].

Other previous work is concerned with the throughput version of related problems. In the demand flow problem on the line, each interval is associated with a profit, and the goal is to maximize the total profit of the accepted intervals without violating any edge capacity. This corresponds to maximizing the total profit of intervals that can receive the same color in our model. For the off-line version of that problem, constant-factor approximation algorithms have been presented in [5, 6] for the case where $b_{\max} \leq c_{\min}$, where $b_{\max} = \max_j b_j$ is the maximum requested bandwidth and $c_{\min} = \min_{i=1,\ldots,n} c_i$ is the minimum edge capacity. For the general case, an approximation ratio $O(\log \frac{b_{\max}}{c_{\min}})$ was achieved in [5]. Recently, a quasi-polynomial time approximation scheme was presented [4].

Our results: In Sect. 3 we present our first main result, a constant competitive algorithm for capacitated interval coloring for the case in which the maximum bandwidth request is at most the minimum capacity, i.e., the case where $b_{\max} \leq c_{\min}$. (Note that this restriction means that the minimum edge capacity *anywhere* on the line must be at least as large as the maximum bandwidth of any request. This is stronger than the standard requirement that $b_j \leq c_i$ for all $i \in I_j$.) This is an important special case that contains the interval coloring problem with bandwidth studied in [1, 15, 3, 8]. This restriction on the maximum bandwidth is common in work on demand flow problems as well, see e.g. [5, 6]. While our algorithm uses the standard technique of partitioning the requests into different types and dealing with each type separately, in our case the different types need to share colors and so the bandwidth sharing scheme for the colors needs to be designed very carefully.

We also consider the general case, i.e., the case where $b_{\max}$ can be larger than $c_{\min}$. First, we remark that it is not difficult to design an $O(n)$-competitive algorithm for this case. This can be done by partitioning the requests into at most n sets, each of which contains all requests for which the bottleneck link (i.e., the link of smallest capacity) is link i. We are left with n disjoint instances of bin packing, and we can run e.g. First-Fit on each set. In Sect. 4, we first design an $O(\log b_{\max})$-competitive algorithm (the ratio is $O(\log \frac{b_{\max}}{c_{\min}})$ if the capacities are not normalized). Then we show that for any amount ε of resource augmentation on the capacities (i.e. increasing capacities by a multiplicative factor of at most $1+\varepsilon$), we can obtain a constant competitive algorithm (the ratio is a function of ε). Finally, we give our second main result, a lower bound showing that no online algorithm can achieve constant competitive ratio in the general case without resource augmentation. The basic idea of our lower bound is to adapt the known logarithmic lower bound for online coloring of trees [10] to our problem. However, arbitrary trees cannot be represented as interval graphs, and hence we need to use the capacities and bandwidths in a very intricate way in order to encode the required tree structures. Furthermore, the construction must be such that the algorithm cannot benefit from the information that is conveyed by the encoding.

Several proofs are omitted due to space limitations.

2 Preliminaries

The following $KT_{\ell b}$ algorithm for the online interval coloring with bandwidth problem (where all edges have the same capacity) was studied by Epstein and Levy [8, 9] (see also [15, 3]). We are given an upper bound b on the maximum request bandwidth. 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 $1, 2, \ldots, m$, together with the additional new request, is at most $m\ell$. (For a set of requests, the *load* created on a link is the sum of the bandwidths of the requests containing that link, and the *maximum load* is the largest load of all links.) For an interval v_i that was allocated to class m, a *critical point of* v_i is a point q in v_i such 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 the interval v_i, has total load greater 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 $b+\ell$.*

Lemma 2. *For every m, the set A_m of intervals that were allocated to class m has a maximum load of at most $2(b+\ell)$.*

Note that the set A_m of intervals assigned to class m can be colored with a single color if its maximum load does not exceed the capacity of any edge (cf. [15]).

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

It was shown in [14] that using the above algorithm with $b = \ell = 1$ in the case where all intervals have unit bandwidth ($b_j = 1$ for all j) results in classes that have maximum load two and can be colored online with three colors per class (the first class can be colored using a single color), assuming unit edge capacity. (If the edges have capacity 2, one color suffices for each class. The same obviously holds also if $b = \ell = \frac{1}{2}$, all requests have bandwidth equal to $\frac{1}{2}$, and the edges have unit capacity.) We refer to this special case of algorithm $KT_{\ell b}$ as algorithm KT; it is the classical algorithm by Kierstead and Trotter that requires at most $3\omega - 2$ colors for coloring a set of intervals with maximum clique size ω.

3 Algorithm for the Case $\max_j b_j \leq \min_{i=1,2,\ldots,n} c_i$

The Algorithm. Since we assume that $\min_{i=1,2,\ldots,n} c_i = 1$, all bandwidth requests are at most 1. The *level* of request $I_j = [s_j, t_j]$ is $\lfloor \log_2 \min_{i \in I_j} c_i \rfloor$, i.e. the rounded down base 2 logarithm of the minimum capacity of any link along the request. A level $i > 0$ request is *small* if its bandwidth is at most 2^{i-3}, and a level 0 request is *small* if its bandwidth is at most $\frac{1}{4}$. A request that is not small is a *large request.* Note that large requests exist only in level 0, 1 and 2.

Our algorithm first rounds down all capacities to integer powers of 2, this does not change the classification into levels. Next it performs an online partition of the requests according to their levels. For all i, the small requests of level i are colored using an algorithm for online coloring along a line network with identical capacities, and these capacities are $\max\{1, 2^{i-1}\}$. For the coloring of these small requests we use the same set of colors for the requests of all levels. More specifically, requests of level 0 are allocated a capacity of 1 in each color, on every link. Requests of level $i > 0$ are allocated a capacity of 2^{i-1} in each color, on every link. To color the small requests, note that a small request has bandwidth at most 2^{i-3} for $i > 0$ and at most $\frac{1}{4}$ for level 0. Therefore we can apply the algorithm $KT_{\ell b}$ from Sect. 2, using $b = \ell = 2^{i-3}$ for $i > 0$ and $b = \ell = \frac{1}{4}$ for level 0. A new class is opened if a new request of some level opens a new class. Each class is colored using a single color, i.e., given color t, it is used for all requests assigned to class t, no matter which level they belong to. It is not difficult to show that this coloring is valid.

As for large requests we first define the following types. We define a *type 1 large request* to be a level 1 large request with bandwidth requirement that belongs to the interval $\left(\frac{1}{2}, 1\right]$. A large request that is not type 1 is called a *type 2 large request.* Each type of large request is packed independently using its own set of colors. We next describe the packing of each type of large requests.

Type 1 large requests. We round up all bandwidth requests to 1 and then apply algorithm KT, the online algorithm for interval coloring (without bandwidth) of Kierstead and Trotter [14]. However, unlike that algorithm, where each class was colored using three colors, we can use a single color for each class, similarly to the algorithm for coloring requests of bandwidth in $(\frac{1}{4}, \frac{1}{2}]$ in [15], see also Sect. 2.

Type 2 large requests. We partition the type 2 large requests into three subgroups according to their levels. For each new open color we allocate a total unit capacity for all the type 2 large requests of level 0. Moreover for each link whose rounded capacity is at least two we also allocate a unit capacity for all the type 2 large request of level 1. For each link whose rounded capacity is at least four we allocate two units of capacity for all the type 2 large requests of level 2. We then apply the following algorithms depending on the level of the large request.

A level 0 large request of type 2. We further partition these requests into two sub-families of requests according to their bandwidth request. The first sub-family consists of requests with bandwidth in $\left(\frac{1}{4}, \frac{1}{2}\right]$, and the second sub-family consists of requests with bandwidth in $\left(\frac{1}{2}, 1\right]$. For each sub-family we use its

own set of colors (note that all these colors can be used also by large requests of type 2 from levels 1 and 2). For each request in the first sub-family we round up its bandwidth request to $\frac{1}{2}$ and then apply algorithm KT, the online algorithm for interval coloring (without bandwidth) of Kierstead and Trotter, where each class can be packed into a common color, as is done for type 1. For the second sub-family we also round up its bandwidth request to 1 and afterwards apply algorithm KT, where each class is packed using three colors, exactly as in [14].
A level 1 large request of type 2. We recall that such a request has bandwidth at most $\frac{1}{2}$. We round up its bandwidth request to $\frac{1}{2}$ and then apply algorithm KT, where each class can be packed into a common color.
A level 2 large request of type 2. We round up its bandwidth request to 1 and apply algorithm KT, where each class can be packed into a common color.

Analysis. First, one can show that the solution returned by the algorithm is feasible. In fact, even the rounded capacity constraints are satisfied by the coloring produced by the algorithm. The next lemma is a trivial consequence of the fact that the colors used to color small requests of the different levels are shared among the levels.

Lemma 4. *Let s_j be the number of colors used to color the small requests of level j. Then, the number of colors used by the algorithm for coloring the small requests is exactly* $\max_{j\geq 0} s_j$.

Furthermore, we can show that $\mathrm{OPT} \geq \frac{s_j}{32}$ for all j. Together with Lemma 4, this gives the following:

Corollary 1. *The number of colors used to color the small requests is at most* $32 \cdot \mathrm{OPT}$.

It remains to analyze the number of colors used by the large requests.

Lemma 5. *Let b be a fixed value that is either $\frac{1}{2}$ or 1 and let c be a fixed value that is either b or $2b$. Assume that we are given a subset $\mathcal{S}$ of large request of level i, $0 \leq i \leq 2$, each with bandwidth in the interval $\left(\frac{b}{2}, b\right]$ and we first round up the bandwidth to b and afterwards use Kierstead and Trotter's algorithm KT with color capacity c. Then, if $b < c$ the number of colors used to color all the requests of this family is at most $2 \cdot \left(\frac{2^{i+2}}{b} - 1\right) \cdot \mathrm{OPT}$, and otherwise (if $b = c$) the number of colors used to color all the requests of this family is at most $6 \cdot \left(\frac{2^{i+2}}{b} - 1\right) \cdot \mathrm{OPT}$.*

Lemma 5 implies, using $b = 1$, $c = 2$ and $i = 1$, that the number of colors that are used by the algorithm to color all type 1 large requests is at most $14 \cdot \mathrm{OPT}$.

Furthermore, we get that the number of colors that are used to color all type 2 large requests of level 0 is at most $32 \cdot \mathrm{OPT}$; this follows by using $b = \frac{1}{2}$, $c = 1$ and $i = 0$ for the requests with bandwidth in $\left(\frac{1}{4}, \frac{1}{2}\right]$, and $b = 1$, $c = 1$ and $i = 0$ for the requests with bandwidth in $\left(\frac{1}{2}, 1\right]$. Similarly, we get that the number of colors for type 2 large requests of level 1 is at most $30 \cdot \mathrm{OPT}$ (using

$b = \frac{1}{2}$, $c = 1$ and $i = 1$), and the number of colors for type 2 large requests of level 2 is at most $30 \cdot \mathrm{OPT}$ (using $b = 1$, $c = 2$ and $i = 2$). As the colors used to color type 2 large requests of different levels are shared among the levels, the number of colors used to color all type 2 large requests is the maximum among the numbers of colors used to color type 2 large requests of level i for $i = 0, 1, 2$. By considering the different cases above, this maximum is at most $32 \cdot \mathrm{OPT}$.

Theorem 1. *The algorithm is* 78*-competitive.*

Proof. Each color is used to either color small requests, or to color large requests of type 1, or to color large requests of type 2. By Corollary 1 there are at most $32 \cdot \mathrm{OPT}$ colors that are used to color small requests. As discussed above, at most $14 \cdot \mathrm{OPT}$ colors are used to color large requests of type 1, and at most $32 \cdot \mathrm{OPT}$ colors are used to color large requests of type 2. The claim follows since $32 \cdot \mathrm{OPT} + 14 \cdot \mathrm{OPT} + 32 \cdot \mathrm{OPT} = 78 \cdot \mathrm{OPT}$. □

4 Algorithms and Lower Bound for the General Case

An $O(\log b_{\max})$-Competitive Algorithm. Recall that $b_{\max}$ denotes the maximum bandwidth of a request. We now deal with the general case where $b_{\max}$ can be larger than $c_{\min}$. We still assume that the edge capacities are normalized so that $c_{\min} = 1$. In order to present an $O(\log b_{\max})$-competitive algorithm, we first note that the algorithm of the previous section is designed in such a way that it can handle small requests even if they have bandwidth requests which are larger than 1 and provides a solution whose cost is at most $32 \cdot \mathrm{OPT}$ for these requests. Therefore, it suffices to consider the large requests. Recall that for $i > 0$, a level i request is large if its bandwidth is at least 2^{i-3}, and it has a link with capacity that is smaller than 2^{i+1}.

In our algorithm we perform an online partition of large requests into levels, and pack the large requests of each level separately using colors that are dedicated to the level. To pack the large requests of level i, we disregard the capacities and bandwidth of the requests, and we pack the requests using Kierstead and Trotter's algorithm KT assuming unit capacities and unit bandwidths. This completes the definition of the algorithm, and it remains to analyze it.

First, it is not difficult to verify that the algorithm produces a feasible solution. The number of levels is $O(\log b_{\max})$, as our algorithm uses colors to color large requests of level i only if there is at least one large request of level i. Furthermore, we can show that for each i our algorithm uses $O(\mathrm{OPT})$ colors to color all the large requests of level i. We thus obtain the following theorem.

Theorem 2. *There exists an $O(\log b_{\max})$-competitive algorithm for the general case of the capacitated interval coloring problem.*

Note that we have assumed $c_{\min} = 1$ without loss of generality. In the case where $c_{\min}$ is not normalized to 1, the ratio becomes $O(\log \frac{b_{\max}}{c_{\min}})$.

Resource Augmentation Algorithm. Given a fixed positive number $0 < \varepsilon < 1$ such that $\frac{1}{\varepsilon}$ is an integer, we allow the online algorithm to use colors such that the total bandwidth of requests that are assigned a common color and contain the link i is at most $(1+\varepsilon)c_i$. I.e., the online algorithm is allowed to use slightly larger capacities than the offline algorithm is allowed. Let $\delta = \frac{\varepsilon}{3}$.

We perform an online partition of the requests into large requests and small requests. We pack the small requests similarly to the previous section with at most $32 \cdot \mathrm{OPT}$ colors. We next describe the algorithm to obtain a coloring of the large requests.

Let $\tilde{c_j}$ denote c_j rounded up to the nearest integer power of $(1+\delta)$. We now define the level of a request $[s_i, t_i]$ to be the logarithm with respect to the base $(1+\delta)$ of the minimum rounded capacity of a link along this request, i.e., $\log_{1+\delta} \min_{s_i \le j \le t_i} \tilde{c_j}$. For each level i we use algorithm KT to compute a packing of its large requests into colors, using a capacity of $(1+\delta)^i$ on each link. For each i, one can show that the algorithm uses $O(\mathrm{OPT})$ colors to color all the large requests of level i.

For each i, we define the type of i to be $i \bmod \frac{1}{\delta^2}$. Therefore, there are exactly $\frac{1}{\delta^2}$ types. For all levels with a common type we use the same set of colors, whereas for different types we use disjoint sets of colors. Therefore, the total number of colors used by our algorithm is at most $O(\frac{1}{\delta^2}) \cdot \mathrm{OPT}$, and this provides a constant competitive ratio for all constant values of δ. Furthermore, we can show that the edge capacities are violated at most by a factor of $(1+\varepsilon)$: Given a color c that is used to color large requests of type i, and a link j whose capacity is c_j, the total bandwidth of requests that are colored c and contain j is at most $(1+3\delta)\, c_j = (1+\varepsilon)c_j$. We obtain the following theorem.

Theorem 3. *For every constant $\varepsilon > 0$, there is a constant competitive algorithm for the general case of the capacitated interval coloring problem with resource augmentation by a factor of $1+\varepsilon$.*

Competitive Lower Bound. We finally outline a lower bound construction showing that no deterministic algorithm can achieve constant competitive ratio in the general case (without resource augmentation). Let $\mathcal{A}$ be any deterministic online algorithm for the problem. We imagine the links of the line numbered from left to right, starting with link 1 as the leftmost link. The capacity of link j is set to 3^j, for all $j \ge 1$. We identify colors with positive integers. Whenever $\mathcal{A}$ uses a new color, and it has used $i-1$ distinct colors prior to using that color, the new color is defined to be color i.

In the adversary construction, each newly presented interval has its left endpoint strictly to the right of all left endpoints of previously presented intervals, and it has a strictly larger bandwidth than all previously presented intervals. In fact, an interval with leftmost link L has bandwidth at least $3^L - 3^{L-1} > 3^{L-1}$. Furthermore, the set of all presented intervals can be colored optimally with two colors.

The adversary strategy makes use of a component (i.e., a subroutine that is used as part of the construction) denoted by $C_F(\ell)$, where F can be any set of positive integers (the set of forbidden colors) and ℓ can be any positive integer. The goal of $C_F(\ell)$ is to force the algorithm to use a color that is not in F. Furthermore, the interval I on which $\mathcal{A}$ uses a color not in F is the last interval presented in the component. The length of I is at least ℓ. A component $C_F(\ell)$ is placed on a part of the line with leftmost link L (i.e., no interval presented in $C_F(\ell)$ contains a link to the left of L). An instance of $C_F(\ell)$ with leftmost link L is also called a $C_F(\ell)$ *at* L. Note that different incarnations of $C_F(\ell)$ may contain different (non-isomorphic) sets of intervals, as the intervals presented by the adversary depend on the actions of the on-line algorithm $\mathcal{A}$. The construction of $C_F(\ell)$ for $|F| > 1$ is recursive and makes use of smaller components $C_{F'}(\ell')$ for $F' \subset F$.

A component $C_F(\ell)$ at L requires a part of the line consisting of $g(\ell, |F|)$ links, for a suitable function g. Note that $g(\ell, |F|) \geq \ell$ must always hold, since already the last interval of $C_F(\ell)$ has length at least ℓ.

The adversary construction satisfies the following invariants.

Invariant 1: When the adversary presents a $C_F(\ell)$ at L, the total bandwidth of all previously presented intervals containing L is at most $\beta_L := 3^{L-1}$.

Invariant 2: Let R' be the leftmost among the rightmost ℓ links of the last interval I of the component $C_F(\ell)$ at L presented by the adversary (i.e., $R' = R - \ell + 1$ if R is the rightmost link of I). The construction ensures that the total bandwidth of intervals from $C_F(\ell)$ that contain R' is at most $3^{R'-1} - 3^{L-1}$.

After a $C_F(\ell)$ at L has been presented, only intervals with left endpoint R' (as defined in Invariant 2) or further to the right will be presented. Note that Invariant 2, together with Invariant 1, implies that the bandwidth of intervals starting to the left of R' and containing R' is at most $3^{L-1} + (3^{R'-1} - 3^{L-1}) = 3^{R'-1}$, so that Invariant 1 automatically holds again for components placed at R' or further to the right.

For $F = \emptyset$, a $C_F(\ell)$ at L consists of a single interval of length $\ell + 1$ with leftmost link L and bandwidth $3^L - \beta_L$. The length of the part of the line required for a $C_F(\ell)$ with $|F| = 0$ is thus $g(\ell, 0) = \ell + 1$. As another simple case to start with, consider the case $F = \{f_1\}$ for some positive integer f_1. The adversary first presents an interval I_1 of length $\ell + 1$ with leftmost link L and bandwidth $3^L - \beta_L$. If $\mathcal{A}$ assigns a color different from f_1 to I_1, the component $C_F(\ell)$ is finished (and I_1 is the last interval of that component). If $\mathcal{A}$ assigns color f_1 to I_1, the adversary next presents an interval I_2 of length $\ell + 1$ whose leftmost link is the rightmost link R of I_1. The bandwidth of I_2 is $3^R - \beta_L$. Algorithm $\mathcal{A}$ must color I_2 with a color different from f_1, because I_1 and I_2 cannot receive the same color (their bandwidths add up to $3^L - \beta_L + 3^R - \beta_L = 3^R + (3^L - 2 \cdot 3^{L-1}) > 3^R$ and both intervals contain link R). The component $C_F(\ell)$ is finished, and I_2 is its last interval. Note that I_1 has rightmost link R and hence does not overlap the rightmost ℓ links of I_2. Therefore, the bandwidth occupied by this $C_F(\ell)$ on its rightmost ℓ links (starting with link $R + 1$) is bounded by $3^R - \beta_L$, showing that Invariant 2 is satisfied. The length of the part of the line required for the $C_F(\ell)$ with $|F| = 1$ is thus $g(\ell, 1) = 2\ell + 1$.

Let $|F| = k$ for some $k > 1$. The idea underlying the construction of $C_F(\ell)$ is to repeatedly use components $C_{F'}(\ell')$ for suitable subsets $F' \subset F$ to force $\mathcal{A}$ to use all colors from F on intervals that all intersect on a common link; then, a new interval that contains that link and is in conflict with the previous intervals containing that link is presented and must receive a color outside F. On the other hand, if the algorithm already uses a color outside F to color an interval presented in one of the recursive constructions $C_{F'}(\ell')$, the construction of $C_F(\ell)$ finishes right away. We can assume (by induction) that Invariant 2 has been shown to hold for the recursive constructions $C_{F'}(\ell')$ that are used in the construction of $C_F(\ell)$, and we will show that Invariant 2 holds again for $C_F(\ell)$.

Assume that $F = \{f_1, f_2, \ldots, f_k\}$. We will show how to construct $C_F(\ell)$ on a part of the line with leftmost link L. The construction proceeds in rounds. There will be at most k rounds, and after the last round one additional final interval may be presented.

For round 0, let $F_0 = \emptyset$. First, the adversary presents a $C_{F_0}(\ell_0)$ for suitable $\ell_0 \geq \ell$ starting at L. If $\mathcal{A}$ assigns a color outside F to the last (and only) interval of $C_\emptyset(\ell_0)$, the construction of $C_F(\ell)$ is finished. Otherwise, we can assume w.l.o.g. that $\mathcal{A}$ assigns color f_1 to the last interval I_0 of $C_\emptyset(\ell_0)$. Let R_0 be the rightmost link of I_0. Let $R'_0 = R_0 - \ell_0 + 1$. The remaining rounds up to round $k-1$ will take place inside the rightmost ℓ_0 links of I_0; only the final interval that may be presented after round $k-1$ extends beyond the right end of I_0.

For round 1, let $F_1 = \{f_1\}$. The adversary presents a $C_{F_1}(\ell_1)$ starting at $L_1 = R'_0$. Observe that the total bandwidth of intervals presented earlier that contain R'_0 is bounded by $3^{R'_0-1}$: bandwidth at most 3^{L-1} from intervals presented before the current $C_F(\ell)$ (by Invariant 1), and bandwidth at most $3^{R'_0-1} - 3^{L-1}$ from the $C_{F_0}(\ell_0)$ that was presented in round 0. The last interval I_1 of $C_{F_1}(\ell_1)$ receives some color c. If $c \notin F$, the construction of $C_F(\ell)$ is finished. If $c \in F$, we can assume w.l.o.g. that $c = f_2$.

In general, assume that round j of the construction of $C_F(\ell)$ has finished. The last interval I_j of the $C_{F_j}(\ell_j)$ presented in round j has received color f_{j+1}. Let R_j be the rightmost link of I_j. Let $R'_j = R_j - \ell_j + 1$. Arguing as above, we know that the total bandwidth of intervals containing R'_j that were presented so far is at most $3^{R'_j-1}$. Let $F_{j+1} = \{f_1, f_2, \ldots, f_{j+1}\}$. The adversary presents a $C_{F_{j+1}}(\ell_{j+1})$ starting at $L_{j+1} = R'_j$. This component will be placed completely inside the rightmost ℓ_j links of I_j. The last interval I_{j+1} of $C_{F_{j+1}}(\ell_{j+1})$ receives some color c. If $c \notin F$, the construction of $C_F(\ell)$ is finished. If $c \in F$, we can assume w.l.o.g. that $c = f_{j+2}$. This finishes round $j+1$.

After round $k-1$, either the construction has finished early and we are done, or the algorithm has used colors $f_1, f_2, \ldots, f_k$ on the intervals $I_0, I_1, \ldots, I_{k-1}$ that were the last intervals of the components $C_{F_j}(\ell_j)$ for $j = 0, \ldots, k-1$. In the latter case, let R_{k-1} be the rightmost link of I_{k-1}. Note that R_{k-1} is also contained in $I_0, \ldots, I_{k-2}$. The adversary presents an interval I_k with leftmost link R_{k-1}, length ℓ_k, and bandwidth $3^{R_{k-1}} - \beta_L$. Note that I_k is in conflict with $I_0, \ldots, I_{k-1}$ on link R_{k-1}, as each of $I_0, \ldots, I_{k-1}$ has bandwidth at least $3^L - 3^{L-1} = 3^L - \beta_L > \beta_L$. Therefore, the algorithm $\mathcal{A}$ must assign a color outside F to I_k, and the construction of $C_F(\ell)$ is finished. ℓ_k is chosen in such a way that the interval I_k extends ℓ links further to the right than any of the previous intervals presented as part of this component $C_F(\ell)$. Note that no other interval (other than I_k) from this $C_F(\ell)$ overlaps the rightmost ℓ links of I_k. Let R_k be the rightmost link of I_k, and let $R'_k = R_k - \ell + 1$. The total bandwidth of intervals from this $C_F(\ell)$ that overlap the rightmost ℓ links of I_k is equal to the bandwidth of I_k, which is less than $3^{R'_k-1} - 3^{L-1}$. Therefore, Invariant 2 is satisfied for this $C_F(\ell)$.

As we know that previously presented intervals of total bandwidth at most 3^{L-1} contain the link L (by Invariant 1), we can conclude that the total bandwidth of intervals overlapping the rightmost ℓ links of I_k is bounded by $3^{L-1} + 3^{R'_k-1} - \beta_L = 3^{R'_k-1}$, so that Invariant 1 continues to hold for components placed at R'_k or further to the right. One can also show that the construction of $C_F(\ell)$ ensures that Invariant 2 is maintained. Furthermore, it is clear that the component $C_F(\ell)$ forces the algorithm to use a color outside the set F.

The length $g(\ell, k)$ of the part of the line that is needed to place a $C_F(\ell)$, for $\ell > 0$, with $|F| = k$ can be calculated to be $g(\ell, k) = \ell + 1$ for $k = 0$ and $g(\ell, k) = a_k(\ell + 1) - 1$ for $k > 0$. Here, the sequence a_n for $n \geq 0$ is defined by $a_0 = 1$ and $a_{n+1} = 1 + \prod_{i=0}^{n} a_i$. We have $a_0 = 1$, $a_1 = 2$, $a_2 = 3$, $a_3 = 7$, $a_4 = 43$, etc. This sequence is known as Sylvester's sequence or the sequence of Euclid numbers. For $n \geq 1$, it satisfies $a_{n+1} = a_n^2 - a_n + 1$. It is known that $a_n = \lfloor c^{2^{n-1}} \rfloor + 1$, where $c \approx 1.59791$ (see [18], sequences A000058 and A007018).

Next, we consider the optimal coloring of $C_F(\ell)$. Consider a $C_F(\ell)$ placed at some link L. Let R be the rightmost link of its last interval I. Let $R' = R - \ell + 1$ be the link at which later components could potentially be placed. Call the set of intervals from $C_F(\ell)$ that contain R' and are different from I the *siblings* of I. We can prove by induction on the size of F that every $C_F(\ell)$ can be colored with 2 colors in such a way that all intervals from the $C_F(\ell)$ containing R' (these are the last interval of $C_F(\ell)$ and its siblings) are assigned the same color. Furthermore, the coloring is such that in each of the two color classes, there is a free capacity of at least 3^{L-1} on all links of the component.

For any $k \geq 1$, we can let $F = \{1, 2, \ldots, k-1\}$ and place a $C_F(1)$ starting at link 1. By the discussion above, the on-line algorithm $\mathcal{A}$ uses a color $\geq k$ on this instance, while the optimum can color all intervals with 2 colors. This shows that $\mathcal{A}$ cannot have competitive ratio better than $k/2$. As k is arbitrary, we obtain the following theorem. Note that the number of links needed to place a $C_F(1)$ is $g(1, k) = 2a_k - 1 = 2(\lfloor c^{2^{k-1}} \rfloor + 1) - 1$, where $c \approx 1.59791$. Thus $k = \Theta(\log \log n)$, where n is the length of the line, and $k = \Theta(\log \log \log c_{\max})$, since the capacity of link i is 3^i.

Theorem 4. *There is no deterministic on-line algorithm for capacitated interval coloring with non-uniform capacities with constant competitive ratio. Moreover, the competitive ratio of any deterministic on-line algorithm for the problem is at least* $\Theta(\log \log n)$ *for lines of length* n *and at least* $\Theta(\log \log \log c_{\max})$ *for lines with maximum edge capacity* $c_{\max}$ *and minimum edge capacity* 1.

References

1. 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.
2. R. Adler and Y. Azar. Beating the logarithmic lower bound: Randomized preemptive disjoint paths and call control algorithms. In *Proceedings of the 10th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'99)*, pages 1–10, 1999.
3. Y. Azar, A. Fiat, M. Levy, and N. Narayanaswamy. An improved algorithm for online coloring of intervals with bandwidth. *Theoretical Computer Science*, 363(1):18–27, 2006.
4. N. Bansal, A. Chakrabarti, A. Epstein, and B. Schieber. A quasi-PTAS for unsplittable flow on line graphs. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC'06)*, pages 721–729, 2006.
5. A. Chakrabarti, C. Chekuri, A. Gupta, and A. Kumar. Approximation algorithms for the unsplittable flow problem. In *Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX'02)*, LNCS 2462, pages 51–66, 2002.
6. C. Chekuri, M. Mydlarz, and F. Shepherd. Multicommodity demand flow in a tree. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP'03)*, LNCS 2719, pages 410–425, 2003.
7. 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.
8. 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.
9. 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.
10. A. Gyárfás and J. Lehel. On-line and first-fit colorings of graphs. *Journal of Graph Theory*, 12:217–227, 1988.
11. T. R. Jensen and B. Toft. *Graph coloring problems*. Wiley, 1995.
12. H. A. Kierstead. The linearity of first-fit coloring of interval graphs. *SIAM Journal on Discrete Mathematics*, 1(4):526–530, 1988.
13. H. A. Kierstead and J. Qin. Coloring interval graphs with First-Fit. *SIAM Journal on Discrete Mathematics*, 8:47–57, 1995.
14. H. A. Kierstead and W. T. Trotter. An extremal problem in recursive combinatorics. *Congressus Numerantium*, 33:143–153, 1981.
15. 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)*, pages 329–338, 2004.
16. S. V. Pemmaraju, R. Raman, and K. R. Varadarajan. Buffer minimization using max-coloring. In *Proceedings of 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'04)*, pages 562–571, 2004.
17. P. Raghavan and E. Upfal. Efficient routing in all-optical networks. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC'94)*, pages 134–143, 1994.
18. N. J. A. Sloane. On-line encyclopedia of integer sequences, 1996–2007. Available on-line at http://www.research.att.com/∼njas/sequences/Seis.html.