ONLINE CAPACITATED INTERVAL COLORING

LEAH EPSTEIN[*], THOMAS ERLEBACH[†], AND ASAF LEVIN[‡]

Abstract. In the online capacitated interval coloring problem, a sequence of requests arrive online. Each request is an interval $I_j \subseteq \{1, 2, \ldots, n\}$ with bandwidth b_j. We are initially given a vector of capacities $(c_1, c_2, \ldots, c_n)$. 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 the general case without resource augmentation.

Key words. competitive analysis, lower bound, interval coloring with bandwidth

AMS subject classifications. 68W25, 68W40, 68Q17

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).

To elaborate the 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

[*]Department of Mathematics, University of Haifa, 31905 Haifa, Israel (lea@math.haifa.ac.il).
[†]Department of Computer Science, University of Leicester, UK (t.erlebach.mcs.le.ac.uk). Part of this work was done during a period of study leave granted by University of Leicester.
[‡]Chaya Fellow. Faculty of Industrial Engineering and Management, The Technion, 32000 Haifa, Israel (levinas@ie.technion.ac.il).

in trees with uniform edge capacities follows easily from the known results for the unit-capacity case [19], but nontrivial new techniques were needed to obtain a 4-approximation for the case with arbitrary capacities [7]. 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 [3]. For the offline version of these problems, Chakrabarti et al. remark in [6] 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. As another example, consider offline throughput maximization of connection requests in stars, each request being associated with a bandwidth, a start time and an end time. Initially a constant-factor approximation was achieved only for the case of uniform capacities [11], and additional techniques were necessary to extend this result to arbitrary capacities [2].

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*.

Without loss of generality we can assume (by scaling) that $\min_{i=1,2,\ldots,n} c_i = 1$. Therefore, we restrict ourselves to this special case.

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.

Online coloring of interval graphs has been intensively studied. Kierstead and Trotter [16] constructed an online algorithm that 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 an interval graph 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 [13]). Many papers studied the performance of First-Fit for this problem [14, 15, 18, 8]. It is shown in [8] that the performance of First-Fit is strictly worse than the one achieved by the algorithm of [16].

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 also studied in [17, 9, 4]. The best competitive ratio known for the problem is 10 [17, 4]. A lower bound strictly higher than 3 was shown in [9].

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 [6, 7] for the case where $b_{\max} \le c_{\min}$, 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, approximation ratio $O(\log \frac{b_{\max}}{c_{\min}})$ was achieved in [6]. Recently, a quasi-polynomial time approximation scheme was presented [5].

Our results: In §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 $b_{\max} \le c_{\min}$. (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 $b_j \le c_i$ for all $i \in I_j$.) This is an important special case that contains the interval coloring problem with bandwidth studied in [1, 17, 4, 9]. This restriction on the maximum bandwidth is common in work on demand flow problems as well, see e.g. [6, 7]. 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.

In §4 we address the general case, i.e., the case where $b_{\max}$ can be larger than $c_{\min}$. In §4.1 we design an $O(\log b_{\max})$-competitive algorithm (the ratio is $O(\log \frac{b_{\max}}{c_{\min}})$ if the capacities are not normalized). In §4.2 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 design a constant competitive algorithm (the ratio is a function of ε). Finally, in §4.3, 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 [12] 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.

2. Preliminaries. 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. In the case of interval coloring without bandwidth, the load of a link is simply the number of intervals containing that link.

As building blocks for our online algorithms we employ algorithm KT, the original 3-competitive interval coloring algorithm due to Kierstead and Trotter [16], and algorithm $KT_{\ell b}$, an adaptation of Algorithm KT that was formulated by Epstein and Levy [10], using ideas from [4] (see also [17, 9]) in the setting where intervals have bandwidths but all links of the line have the same capacity. The basic idea of KT is to assign each interval to a *class* depending on the load of previously processed intervals assigned to the same or smaller classes, and to color the intervals in each class simply using First-Fit. It turns out that the competitive ratio achieved for interval coloring in this way is better than if First-Fit is applied to all intervals without a prior partition into classes [8]. Algorithm $KT_{\ell b}$ is a natural adaptation of KT to the setting with bandwidth. The partition into classes is performed based on the load and depending on a parameter ℓ, and each class is colored using First-Fit. Typically ℓ is chosen in such a way that a single color is sufficient for the requests assigned to the same class. The competitive ratio of both algorithms can be analyzed by relating the number of classes to the maximum load of the requests and bounding the number of colors used for each class.

More formally, the $KT_{\ell b}$ algorithm for the online interval coloring with bandwidth problem is defined as follows. We are given an upper bound $b > 0$ on the maximum request bandwidth and a parameter $\ell > 0$. 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 an interval v_i that was allocated to class m, a *critical point of* v_i is a point q (here, we use the terms *point* and *link* synonymously) 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 more than $(m-1)\ell$ in q (i.e., q prevents the allocation of v_i to class $m-1$).

The following lemmas have been proved in [10] for algorithm $KT_{\ell b}$.

LEMMA 2.1. *Consider an interval v_i that was allocated to 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 2.2. *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. [17]).

LEMMA 2.3. *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 [16] 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 capacities. (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.) Algorithm KT is exactly this special case of algorithm $KT_{\ell b}$; 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 $\max_j b_j \leq \min_{i=1,2,\ldots,n} c_i$.

3.1. The algorithm. As we assume that $\min_{i=1,2,\ldots,n} c_i = 1$, all bandwidth requests are at most 1. We define the *level* of request $I_j = [s_j, t_j]$ to be $\lfloor \log_2 \min_{i \in I_j} c_i \rfloor$, i.e. the rounded down base 2 logarithm of the minimum capacity of any link along the request (we also call such a link with minimum capacity a *bottleneck link* of the request).

The main idea of our algorithm is to partition the requests into different types depending on their level and bandwidth, and to apply an appropriate variant of $KT_{\ell b}$ to each type. As is usual in online problems of this kind, it appears difficult to handle small requests (having small bandwidth compared to the capacity of the bottleneck link) and large requests (large bandwidth compared to the capacity of the bottleneck link) together and still achieve a constant competitive ratio. Therefore, we treat these two kinds of requests separately using disjoint sets of colors. Furthermore, requests of the same level can, in a certain sense, be treated like requests on a line with uniform link capacities because of the way in which their maximum load can be used to derive a lower bound on the optimal number of colors (see Lemmas 3.5 and 3.6). Hence, we

TABLE 3.1
Overview of request classification

Level	Bandwidth	Classification
0	$\leq \frac{1}{4}$	small request
	$\in (\frac{1}{4}, 1]$	large request (type 2)
1	$\leq \frac{1}{4}$	small request
	$\in (\frac{1}{4}, \frac{1}{2}]$	large request (type 2)
	$\in (\frac{1}{2}, 1]$	large request (type 1)
2	$\leq \frac{1}{2}$	small request
	$\in (\frac{1}{2}, 1]$	large request (type 2)
$i > 2$	$\in (0, 1]$	small request

apply $KT_{\ell b}$ to the small requests of each level, and we carefully share the capacity of each link among the small requests of all levels so that the same colors can be used for small requests of different levels. For the large requests (which exist only on a small number of levels), one could achieve a constant competitive ratio by simply treating them as standard intervals and applying algorithm KT, but we obtain a better constant by distinguishing further types and partially sharing colors between different levels.

We now give a detailed description of the algorithm. Formally, small and large requests are defined as follows. 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. See Table 3.1 for an overview of the request classification.

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 §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. We later 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 [16]. 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 [17], see also §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 [16].

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 then apply algorithm KT, where each class can be packed into a common color.

3.2. Analysis. We first show that the solution returned by the algorithm is feasible, i.e., satisfies the capacity constraints. We actually show that already the rounded capacity constraints are satisfied. We first consider the colors of the small requests.

LEMMA 3.1. *The sum of bandwidths allocated to each link and to each color used by small requests (for all levels together) is at most its rounded capacity.*

Proof. Consider a specific link, and assume that its (rounded) capacity is 2^s. Note that there are intervals that use this link only in levels $0, 1, \ldots, s$. Given a level i, by Lemma 2.2, the total bandwidth of intervals colored by one color is at most $2(2^{i-3} + 2^{i-3}) = 2^{i-1}$ for $i > 0$ and at most $2(\frac{1}{4} + \frac{1}{4}) = 1$ for level 0. The total bandwidth of intervals with a common color from level $i \le s$ that use this link is therefore at most $\max\{1, 2^{i-1}\}$. The claim follows by noting that $1 + \sum_{i=1}^{s} 2^{i-1} = 2^s$. ☐

LEMMA 3.2. *The sum of bandwidths allocated to each link and to each color used by large requests of type 2 (for all levels together) is at most its rounded capacity.*

Proof. We fix a color that was used by a large request of type 2.

(i) For a link whose rounded capacity is one, we use this link and this color only for level 0 requests, and for these requests we allocate at most one unit of capacity on all the links (that is its rounded capacity).

(ii) For a link whose rounded capacity is two, we use this link and this color only for level 0 and level 1 requests. The sum of bandwidths allocated to this link and this color of level 0 requests is at most one unit of capacity. Similarly, the sum of bandwidths allocated to this link and this color of level 1 requests is at most one unit of capacity. Therefore, in this case also the sum of bandwidths allocated to each link and each color used by large requests of type 2 (for all levels) is at most its rounded capacity.

(iii) For a link whose rounded capacity is at least four, we note that the total allocated bandwidth of this color in a fixed link is at most four units of capacity, since the requests of level 2 are allocated a bandwidth of at most 2 per color.
This completes the proof. ∎

By Lemmas 3.1 and 3.2, using the fact that a color used by large requests of type 1 clearly satisfies the capacity constraints, we conclude the following corollary.

COROLLARY 3.3. *The algorithm constructs a feasible solution.*

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 3.4. *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* $\max_{j\geq 0} s_j$.

We let a *critical link of level j* be a link with rounded capacity 2^j. It is clear that each interval of level j must contain a critical link of level j. Furthermore, note that a critical link is not the same as a critical point (defined in §2).

LEMMA 3.5. *Let p be a link along the line such that the total load of requests of level i is maximized in p. Then, the load in p is at most twice the maximum load of a critical link of level i. Moreover, if there are L level i requests that use p, then there is a critical link q of level i such that there are at least $\frac{L}{2}$ level i requests that use q.*

Proof. Consider the set S_p of intervals that contain p. If p is a critical link of level i, then the largest load on any critical link of level i is simply the load on p. Next, consider the situation where this is not the case. If there are no critical links of level i on the left-hand side of p, then p is contained in a minimal interval $[p, p+1, \ldots, b]$, where b is a critical link of level i. All intervals of level i that contain link p must contain link b, since they must contain a critical link. Similarly, if there are no critical links of level i on the right-hand side of p, then p is contained in a minimal interval $[a, a+1, \ldots, p]$, where a is a critical link of level i. All intervals of level i that contain link p must contain link a. In both cases, again the load caused by level i intervals on the critical link is at least the load on p. Finally, if there is at least one critical link on each side of p, p is contained in a minimal interval $[a, a+1, \ldots, b]$ such that both a and b are critical links of level i. Each interval in S_p contains either a or b or both of them. All intervals in S_p that contain a are added to S_a, and we let $S_b = S_p \setminus S_a$. The set S_a contributes to the load in a, and the set S_b contributes to the load in b. Therefore, the load in p is at most twice the maximum load of a critical link of level i (and a similar conclusion holds with respect to the number of requests instead of the load). ∎

LEMMA 3.6. *Assume that $s_i = \max_{j\geq 0} s_j$. Then,* $\mathrm{OPT} \geq \frac{s_i}{32}$.

Proof. Assume $i > 0$, and let p be a link along the line such that the total load of requests of level i is maximized in p. Since color s_i is used, the load in p is greater than $2^{i-3}(s_i - 1)$ (as a new class is determined by $\ell = 2^{i-3}$). Therefore, by Lemma 3.5, the load of some critical link of level i is greater than $2^{i-4}(s_i - 1)$ (half the load of p). Denote by OPT_i the minimum number of colors that are necessary to color the small requests of level i. Since the original capacity (before the rounding) of a critical link of level i is less than 2^{i+1}, we conclude that $\mathrm{OPT}_i > \frac{2^{i-4}(s_i-1)}{2^{i+1}} = \frac{s_i-1}{32}$. Therefore, $32 \cdot \mathrm{OPT} \geq 32 \cdot \mathrm{OPT}_i > s_i - 1$, and since OPT is integer, we conclude that $32 \cdot \mathrm{OPT} \geq s_i$. For $i = 0$, a similar analysis shows the claim. ∎

Using Lemmas 3.4 and 3.6, we establish the following:

COROLLARY 3.7. *The number of colors used to color the small requests is at most* $32 \cdot \mathrm{OPT}$.

It remains to analyze the cost caused by the large requests.

LEMMA 3.8. *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 $6 \cdot \left(\frac{2^{i+2}}{b} - 1\right) \cdot \mathrm{OPT}$.*

Proof. Consider the maximum number L of requests from $\mathcal{S}$ that share a common link and denote this link by p.

(i) If $b < c$ then $b = \frac{c}{2}$ and we can color each class within Kierstead and Trotter's algorithm using one color. Therefore, the number of colors that are used by the algorithm for coloring the requests of $\mathcal{S}$ is at most the largest clique size, which is L.

(ii) If $b = c$, then we can color each class within Kierstead and Trotter's algorithm using three colors. Therefore, the number of colors that are used by the algorithm for coloring the requests of $\mathcal{S}$ is at most $3L$.

We next show a lower bound on OPT. By Lemma 3.5, there is a critical link q of level i with at least $\frac{L}{2}$ requests from $\mathcal{S}$ that contain q. Since the (original) capacity of q is less than 2^{i+1} and each request from $\mathcal{S}$ has a bandwidth of at least $\frac{b}{2}$, we conclude that each color in the optimal solution may be used for at most $\frac{2^{i+2}}{b} - 1$ requests from $\mathcal{S}$ that contain q. Since there are at least $\frac{L}{2}$ intervals that contain the critical link q, we conclude that $\mathrm{OPT} \geq \frac{L}{2 \cdot \left(\frac{2^{i+2}}{b} - 1\right)}$. Since if $b < c$ the algorithm uses at most L colors to color $\mathcal{S}$ and otherwise it uses at most $3L$ colors, we conclude that the claim of the lemma holds. ∎

LEMMA 3.9. *The number of colors that are used by the algorithm to color all type 1 large requests is at most $14 \cdot \mathrm{OPT}$.*

Proof. By Lemma 3.8, using $b = 1$, $c = 2$ and $i = 1$. ∎

LEMMA 3.10. *The number of colors that are used by the algorithm to color all type 2 large requests of level 0 is at most $32 \cdot \mathrm{OPT}$.*

Proof. By Lemma 3.8, using $b = \frac{1}{2}$, $c = 1$ and $i = 0$ we conclude that the number of colors used by the algorithm to color all type 2 large requests of level 0 with bandwidth in $\left(\frac{1}{4}, \frac{1}{2}\right]$ is at most $14 \cdot \mathrm{OPT}$. By Lemma 3.8, using $b = 1$, $c = 1$ and $i = 0$, the number of colors used by the algorithm to color all type 2 large requests of level 0 with bandwidth in $\left(\frac{1}{2}, 1\right]$ is at most $18 \cdot \mathrm{OPT}$. ∎

LEMMA 3.11. *The number of colors that are used by the algorithm to color all type 2 large requests of level 1 is at most $30 \cdot \mathrm{OPT}$.*

Proof. By Lemma 3.8, using $b = \frac{1}{2}$, $c = 1$ and $i = 1$. ∎

LEMMA 3.12. *The number of colors that are used by the algorithm to color all type 2 large requests of level 2 is at most $30 \cdot \mathrm{OPT}$.*

Proof. By Lemma 3.8, using $b = 1$, $c = 2$ and $i = 2$. ∎

LEMMA 3.13. *The number of colors that are used by the algorithm to color all type 2 large requests is at most $32 \cdot \mathrm{OPT}$.*

Proof. The colors used to color type 2 large requests of different levels are shared among the levels. Therefore, 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 Lemmas 3.10, 3.11 and 3.12, this maximum is at most $32 \cdot \mathrm{OPT}$. □

THEOREM 3.14. *The algorithm is* 78*-competitive.*

Proof. Each color used by the algorithm 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 3.7 there are at most $32 \cdot \mathrm{OPT}$ colors that are used to color small requests. By Lemma 3.9 there are at most $14 \cdot \mathrm{OPT}$ colors used to color large requests of type 1. By Lemma 3.13, there are at most $32 \cdot \mathrm{OPT}$ colors that 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. In this section, we deal with the general case where $b_{\max}$ can be larger than $c_{\min}$.

4.1. An $O(\log b_{\max})$-competitive algorithm. We denote by $b_{\max}$ the maximum bandwidth of a request. Let small requests be defined as in §3.1, i.e., as requests with bandwidth at most $\frac{1}{4}$ on level 0 and with bandwidth at most 2^{i-3} on level i for $i > 0$. We further define *medium* requests as those whose bandwidth b_j satisfies $\frac{1}{4} < b_j \leq \frac{1}{2}$ on level 0 and $2^{i-3} < b_j \leq 2^{i-2}$ on level i for $i > 0$. Requests that are neither small nor medium are called *large* requests.

In order to obtain 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. This is because the proofs of Lemmas 3.1 and 3.6 do not use the fact that the maximum bandwidth is at most 1. Therefore, it suffices to consider the medium and large requests. It turns out that we can let the medium requests of different levels share colors in a similar way as the small requests, and we will see that they can be colored with $30 \cdot \mathrm{OPT}$ colors. We will show that the large requests of each level can be colored with $42 \cdot \mathrm{OPT}$ colors using algorithm KT, but we cannot share colors between levels for the large requests, and hence the overall competitive ratio is $O(\log b_{\max})$, since there are $O(\log b_{\max})$ levels of large requests.

Now we describe the algorithm for medium and large requests in detail. For the medium requests, we do the following. We use a separate set of colors for medium requests, but this set of colors is shared by the medium requests of all levels. For each such color, we set aside different parts of the capacity of each link for each level of medium requests. On each link, we set aside 1 unit of capacity for medium requests of level 0. On each link with capacity at least 2^i, for $i > 0$, we additionally set aside 2^{j-1} units of capacity for requests of level j, for each $1 \leq j \leq i$. This does not exceed the capacity of any link. We apply Kierstead and Trotter's algorithm KT (ignoring bandwidth requests and capacities) to the medium requests of each level, but using a single color for each class created by the algorithm; this is possible because each class has maximum clique size 2, and the bandwidths of two medium requests of level i, $i > 0$, add up to at most 2^{i-1}, which is the capacity that has been set aside for medium requests of level i in each color (and for $i = 0$, the reasoning is analogous).

LEMMA 4.1. *The number of colors that the algorithm uses for medium requests of level i, for any $i \geq 0$, is at most* $30 \cdot \mathrm{OPT}$.

Proof. Consider the medium requests of level i. Let L be the maximum number of such requests that contain the same link. Algorithm KT opens L classes for this set of requests. As our algorithm uses a single color for each class, L colors suffice.

By Lemma 3.5, we know that there is a critical link p of level i that is contained in at least $L/2$ requests. The capacity of p is less than 2^{i+1}, and each medium request of level i has a bandwidth request larger than 2^{i-3}. Therefore, even the optimal coloring can assign the same color to less than $2^{i+1}/2^{i-3} = 16$ requests and thus needs at least $L/(2 \cdot 15)$ colors. So the number of colors used by the algorithm is at most $30 \cdot \mathrm{OPT}$. ∎

For the large requests, we again use a separate set of colors. Furthermore, the large requests of each level use their own set of colors (not shared between levels). To color the large requests of level i, we disregard the capacities and bandwidth of the requests, and we color the requests using Kierstead and Trotter's algorithm KT assuming unit capacities and unit bandwidths.

LEMMA 4.2. *For each $i \geq 0$, the algorithm uses at most $42 \cdot \mathrm{OPT}$ colors to color all the large requests of level i.*

Proof. Consider the maximum number L of large requests of level i that share a common link and denote this link by p. Algorithm KT produces L classes and colors each class using at most three colors. Therefore, the number of colors that are used by the algorithm for coloring the large requests of level i is at most $3L$.

We next show a lower bound on OPT. By Lemma 3.5, there is a critical link q of level i with at least $\frac{L}{2}$ large requests of level i that contain q. Since the capacity of q is less than 2^{i+1} and each large request of level i has a bandwidth of at least 2^{i-2}, we conclude that each color in the optimal solution may be used for at most 7 large requests of level i that contain q. Therefore, $\mathrm{OPT} \geq \frac{L}{2 \cdot 7}$. The claim follows since the algorithm uses at most $3L$ colors. ∎

Putting the parts together, in our algorithm we perform an online partition of requests into small, medium and large requests, and we color each type of requests separately using disjoint sets of colors.

THEOREM 4.3. *There exists an $O(\log b_{\max})$-competitive algorithm for the general case of the capacitated interval coloring problem.*

Proof. First, observe that the coloring produced by our algorithm is feasible. The colors for the small requests are feasible by Lemma 3.1. The colors for the medium requests are feasible because for each link the capacities set aside for different levels sum up to at most the total capacity of the link. For a color that is used by large requests of level i, note that we do not color intersecting requests using the same color, and thus we do not exceed the capacity of any link.

The algorithm uses up to $32 \cdot \mathrm{OPT}$ colors for small requests, $30 \cdot \mathrm{OPT}$ colors for medium requests (as the medium requests of all levels share colors), and $42 \cdot \mathrm{OPT}$ colors for large requests of each level. The number k of relevant levels of large requests 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. The total number of colors used by the algorithm is bounded by $62 \cdot \mathrm{OPT} + k \cdot 42 \cdot \mathrm{OPT} = O(\log b_{\max}) \cdot \mathrm{OPT}$. ∎

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 $O(\log \frac{b_{\max}}{c_{\min}})$.

4.2. 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 $\delta = \frac{\varepsilon}{3}$. Observe that $\varepsilon \leq \frac{1}{2}$ and $\delta \leq \frac{1}{6}$.

Define small requests, medium requests, and large requests in the same way as in §4.1. We perform an online partition of the requests into small requests, medium requests, and large requests. The small requests and medium requests are colored in the same way as in §4.1, using a total of at most $62 \cdot \mathrm{OPT}$ colors. We next describe the algorithm to obtain a coloring of the large requests. The main idea is to partition the large requests into finer levels (called δ-levels) and to use the slightly increased edge capacities to let requests in every k-th δ-level, for a suitable constant k depending on δ, share colors. The details are as follows.

Let $\tilde{c_i}$ denote c_i rounded up to the nearest integer power of $1+\delta$. We define the *δ-level* of a request $[s_j, t_j]$ to be the logarithm with respect to the base $1+\delta$ of the minimum rounded capacity of a link along this request, i.e., $\log_{1+\delta} \min_{s_j \le i \le t_j} \tilde{c_i}$. Note that the δ-level of a request is different from its level; the level of request j with interval I_j is still defined as $\lfloor \log_2 \min_{i \in I_j} c_i \rfloor$.

For each δ-level i of requests, we use algorithm KT (ignoring request bandwidths and capacities) to compute a packing of its large requests into colors. Note that no two intersecting intervals are assigned the same color, so we use a capacity of at most $(1+\delta)^i$ on each link. We can adapt the idea of Lemma 4.2 to obtain the following lemma.

LEMMA 4.4. *For each $i \ge 0$, the algorithm uses at most $54 \cdot \mathrm{OPT}$ colors to color all the large requests of δ-level i.*

Proof. Consider the maximum number L of large requests of δ-level i that share a common link and denote this link by p. As in the proof of Lemma 4.2, it follows that the algorithm uses at most $3L$ colors on the large requests of δ-level i.

We next show a lower bound on OPT. By a straightforward adaptation of the proof of Lemma 3.5, it follows that there is a critical link q of δ-level i (i.e., q has rounded capacity $\tilde{c_q} = (1+\delta)^i$) that is contained in at least $\frac{L}{2}$ large requests of δ-level i. The original capacity c_q of q satisfies $c_q \le (1+\delta)^i$. For each request j of δ-level i, we have that $\min_{k \in I_j} c_j > (1+\delta)^{i-1}$. This implies that the level of j is at least $\ell = \lfloor \log_2 (1+\delta)^{i-1} \rfloor$. As j is a large request of level ℓ, its bandwidth is at least $2^{\ell-2} \ge (1+\delta)^{i-1}/8$. It follows that at most $\lfloor 8(1+\delta) \rfloor = 9$ large requests of δ-level i that contain q can share a color, so the optimum coloring needs at least $L/18$ colors. As our algorithm uses at most $3L$ colors, the claim follows. ∎

For each $i \ge 0$, we define the *type* of δ-level i to be $i \bmod \frac{1}{\delta^2}$. Therefore, there are exactly $\frac{1}{\delta^2}$ types. For the large requests of all δ-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 $\left(62 + 54 \cdot \frac{1}{\delta^2}\right)\mathrm{OPT}$, and this provides a constant competitive ratio for all constant values of δ. It remains to show that we exceed the capacity of a link by a multiplicative factor of at most $1+\varepsilon$.

LEMMA 4.5. *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 $(1+3\delta)\, c_j = (1+\varepsilon)c_j$.*

Proof. In order to determine the largest δ-level of type i that can have requests which contain link j, denote by k the maximum integer value so that $(1+\delta)^{i+\frac{k}{\delta^2}} \le \tilde{c_j}$. Each δ-level that contributes to the load of link j in color c is of type i. Each δ-level i' of type i such that $i' \le i + \frac{k}{\delta^2}$ adds to the total load of link j at most $(1+\delta)^{i'}$ and a δ-level i' of type i such that $i' > i + \frac{k}{\delta^2}$ adds nothing to the load of link j (such a request would be invalid in the given network). Therefore, the total load of link j is at most

$$\sum_{\ell=0}^{k} (1+\delta)^{i+\frac{\ell}{\delta^2}} \;=\; (1+\delta)^{i+\frac{k}{\delta^2}} \cdot \sum_{\ell=0}^{k} \left(\frac{1}{1+\delta}\right)^{\frac{\ell}{\delta^2}} \le_{(1)} (1+\delta)^{i+\frac{k}{\delta^2}} \cdot \sum_{\ell=0}^{\infty} \left(\frac{1}{1+\delta}\right)^{\frac{\ell}{\delta^2}}$$

$$\leq_{(2)} c_j \cdot (1+\delta) \cdot \sum_{\ell=0}^{\infty} \left(\frac{1}{1+\delta}\right)^{\frac{\ell}{\delta^2}} \leq_{(3)} \; c_j \cdot (1+\delta) \cdot \sum_{\ell=0}^{\infty} \left(1 - \frac{1}{1+\delta}\right)^{\ell}$$
$$= \; c_j \cdot (1+\delta)^2 \leq_{(4)} \; c_j \cdot (1+3\delta) = (1+\varepsilon)c_j$$

where (1) holds because $1+\delta > 0$, (2) holds because $(1+\delta)^{i+\frac{k}{\delta^2}} \leq \tilde{c_j} \leq (1+\delta)c_j$, (3) holds because $\left(\frac{1}{1+\delta}\right)^{\frac{1}{\delta^2}} \leq 1 - \frac{1}{1+\delta}$ (see below), and (4) holds as $\delta < \varepsilon \leq 1$.

We outline a proof that $\left(\frac{1}{1+\delta}\right)^{\frac{1}{\delta^2}} \leq 1 - \frac{1}{1+\delta}$ holds for all $\delta \leq 1$. By substituting $x = 1/\delta$ and taking the inverse on both sides of the inequality, we arrive at the following equivalent inequality:

$$(1+1/x)^{x^2} \geq x+1\,.$$

As $(1+1/x)^x \approx \mathrm{e}$ for large x, we see that the left-hand side of the inequality grows as e^x and thus the inequality clearly holds for large enough x. Furthermore, it is not difficult to verify that the inequality actually holds for all $x \geq 1$. A formal proof can be obtained by observing that the inequality holds with equality for $x = 1$, and the derivative of the left-hand side is greater than 1 for all $x \geq 1$. ∎

We obtain the following theorem.

THEOREM 4.6. *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 $1+\varepsilon$.*

4.3. Competitive lower bound. We give 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 $j \geq 1$. The number of links of the line will be determined later.

First, we give an informal outline of the ideas underlying the construction. As a starting point, consider the known lower bound for online coloring of trees [12]. If one wants to force a graph coloring algorithm to use color k on a tree, one can recursively present $k-1$ smaller subtrees that force the algorithm to use colors 1 to $k-1$ in different subtrees, and then present an extra node that is adjacent to $k-1$ nodes from the different subtrees that have been assigned colors 1 to $k-1$. When translating this idea to intervals with bandwidth, the main difficulty is that arbitrary trees cannot be represented as interval graphs. Instead of subtrees, we use "components," which are sequences of intervals that force an algorithm to use a "new" color on the last interval of the component. Furthermore, that last interval I (and potentially some other overlapping intervals, which do not affect the construction due to their small bandwidth) extends far to the right compared to the rest of the component. We can then place further components (to force the algorithm to use other new colors) inside that last interval I, because the exponentially increasing capacities of the links of the line ensure that the bandwidth of I is negligible compared to the bandwidth of intervals placed in these further components. Thus we can let a number of components overlap in such a way that their last intervals all intersect and extend further to the right than the rest of the components. Then an extra interval (corresponding to the extra node mentioned in the discussion of the tree case above) can be placed to the right in such a way that it intersects (and is in conflict with) all the last intervals of the components, thus forcing the algorithm to use a new color. Hence, the algorithm can be forced to use an arbitrary number of colors, while the optimal number of colors remains 2 as can be shown by a constructive offline coloring procedure for the components.

The formal proof follows. 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.

The adversary construction has the following properties:

(i) Each newly presented interval has its left endpoint strictly to the right of all left endpoints of previously presented intervals.

(ii) Each newly presented interval has a strictly larger bandwidth than all previously presented intervals. In fact, an interval with leftmost link L has bandwidth at least $3^L - 3^{L-1} > 3^{L-1}$.

(iii) 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 whose leftmost link is some 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 (to be determined later). Note that $g(\ell, |F|) \geq \ell$ always holds, 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 is about to present 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 $C_F(\ell)$.

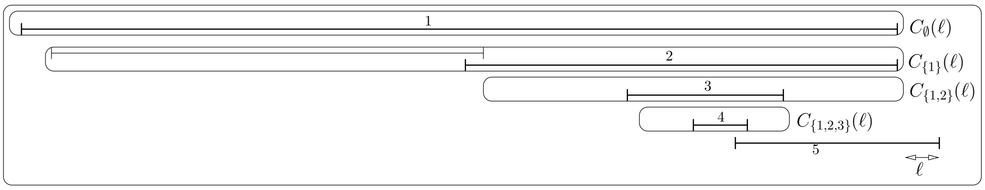

FIG. 4.1. *Example illustrating the construction of the component $C_F(\ell)$ that is used in the lower bound, here for $F = \{1, 2, 3, 4\}$. The boxes with rounded corners inside the larger box represent the smaller components used in the construction of $C_{\{1,2,3,4\}}(\ell)$. The last interval in each component is drawn in bold and labelled with the color assigned to it by the algorithm in this example. The last interval of the construction is in conflict with the four last intervals of the smaller components and is assigned color 5 by the algorithm. For $C_{\{1,2\}}(\ell)$ and $C_{\{1,2,3\}}(\ell)$, only the last interval of the component is shown. Note that each component is placed inside the last interval of the previous component, no matter whether that interval extends until the right end of that component or not.*

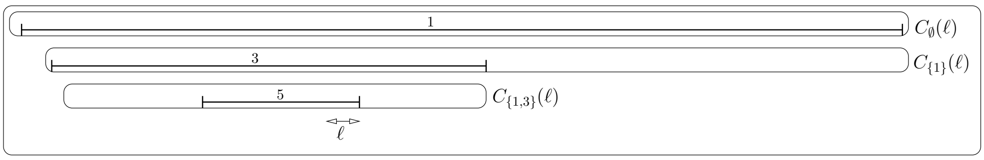

FIG. 4.2. *Different example for the construction of $C_{\{1,2,3,4\}}(\ell)$. Here, the algorithm uses color 3 on the last (and only) interval of $C_{\{1\}}(\ell)$, so that the adversary next presents a $C_{\{1,3\}}(\ell)$. The algorithm uses color 5 already for one of the intervals of $C_{\{1,3\}}(\ell)$, so that the construction finishes early.*

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 (numbered from 0 to $k-1$), and after the last round one additional final interval may be presented. Illustrations are given in Figs. 4.1 and 4.2.

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. Thus ℓ_0 must be chosen large enough. We will determine the exact size of ℓ_0 later.

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 and 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 $C_{F_{j+1}}(\ell_{j+1})$ 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 (Fig. 4.2 shows an example), 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 (Fig. 4.1 shows an example). 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.

It is clear that the component $C_F(\ell)$ forces the algorithm to use a color outside the set F. Furthermore, the construction of $C_F(\ell)$ also ensures that Invariant 2 holds for it, i.e., that the total bandwidth of intervals from the $C_F(\ell)$ that overlap the rightmost ℓ links of its last interval is at most $3^{R'-1} - 3^{L-1}$, where R' is the leftmost among the rightmost ℓ links of the last interval. If the last interval is presented after round $k-1$, this follows as discussed above. If the construction ends early, the last interval of the $C_F(\ell)$ is also the last interval of a component $C_{F_j}(\ell_j)$ placed at some link L_j in the construction. By induction, we know that the intervals of the $C_{F_j}(\ell_j)$ that overlap the rightmost ℓ_j links of its last interval have total bandwidth at most $3^{R'_j-1} - 3^{L_j-1}$, where R'_j is the leftmost among the rightmost ℓ_j links of the last interval. Furthermore, the total bandwidth of intervals from the $C_{F_0}(\ell_0)$, $C_{F_1}(\ell_1)$, $\ldots$, $C_{F_{j-1}}(\ell_{j-1})$ placed to the left of the $C_{F_j}(\ell_j)$ that overlap R'_j is bounded by $3^{L_j-1} - 3^{L-1}$ (since Invariant 2 holds for all these components). Therefore, the total bandwidth of intervals from the $C_F(\ell)$ that overlap the rightmost ℓ_j links of the last interval is at most $3^{L_j-1} - 3^{L-1} + 3^{R'_j-1} - 3^{L_j-1} = 3^{R'_j-1} - 3^{L-1}$, establishing that Invariant 2 holds for the $C_F(\ell)$.

In the following, we first calculate the lengths ℓ_j for $j = 0, \ldots, k$ that are needed in the construction of a $C_F(\ell)$; from this we derive that the length $g(\ell, k)$ of the part of the line that is needed to place a $C_F(\ell)$, for $\ell > 0$, with $|F| = k$ is $g(\ell, k) = \ell + 1$ for $k = 0$ and $g(\ell, k) = a_k(\ell + 1) - 1$ for $k > 0$, where the sequence a_n for $n \geq 0$ is defined by $a_0 = 1$ and $a_{n+1} = 1 + \prod_{i=0}^{n} a_i$. Afterwards, we will show that the set of intervals of any $C_F(\ell)$ can be colored with 2 colors by an optimal offline algorithm. Finally, we put everything together to obtain the lower bound on the competitive ratio.

Before we begin the technical analysis, we give a concrete example for the execution of the adversary strategy by showing the construction of a $C_F(1)$ for $F = \{1, 2, 3, 4\}$ against the First-Fit coloring algorithm. For this construction, a line with $g(1, 4) = 2a_4 - 1 = 85$ links is needed. The list of intervals is as follows. For each interval we specify the indices of the first and last link, its bandwidth requirement, its color (1 or 2) in an optimal coloring, and the color that First-Fit assigns to it:

1. $[1, 84]$, $3^1 - 3^0$, 2, 1.
2. $[2, 43]$, $3^2 - 3^1$, 1 ,1.
3. $[43, 84]$, $3^{43} - 3^1$, 2, 2.
4. $[44, 71]$, $3^{44} - 3^{43}$, 1, 1.
5. $[45, 58]$, $3^{45} - 3^{44}$, 2, 1.
6. $[58, 71]$, $3^{58} - 3^{44}$, 1, 2.
7. $[71, 84]$, $3^{71} - 3^{43}$, 2, 3.
8. $[72, 83]$, $3^{72} - 3^{71}$, 1, 1.
9. $[73, 78]$, $3^{73} - 3^{72}$, 2, 1.
10. $[78, 83]$, $3^{78} - 3^{72}$, 1, 2.
11. $[79, 82]$, $3^{79} - 3^{78}$, 2, 1.
12. $[80, 81]$, $3^{80} - 3^{79}$, 1, 1.
13. $[81, 82]$, $3^{81} - 3^{79}$, 2, 2.
14. $[82, 83]$, $3^{82} - 3^{78}$, 1, 3.
15. $[83, 84]$, $3^{83} - 3^{71}$, 2, 4.
16. $[84, 85]$, $3^{84} - 3^0$, 1, 5.

4.3.1. Calculating the lengths. Recall that $g(\ell, k)$ denotes the length of the part of the line that is needed to place a $C_F(\ell)$ for $|F| = k$. We have seen that $g(\ell, 0) = \ell + 1$ and $g(\ell, 1) = 2\ell + 1$.

During the construction of a $C_F(\ell)$, we need that $\ell_j \geq \ell$ for $0 \leq j \leq k - 1$, since the last interval of each component $C_{F_j}(\ell_j)$ could potentially be the last interval of the $C_F(\ell)$. Furthermore, we need each ℓ_j to be large enough so that all the components $C_{F_{j'}}(\ell_{j'})$ for $j' > j$ fit inside a range of length ℓ_j.

We get the following conditions:

(i) $\ell_{k-1} \geq \ell$. This must hold as the last interval of the $C_{F_{k-1}}(\ell_{k-1})$ could be the last interval of $C_F(\ell)$. We have no other condition on ℓ_{k-1}, so we can choose $\ell_{k-1} = \ell$.

The space on the line that is required by the $C_{F_{k-1}}(\ell_{k-1})$ is then $g(\ell_{k-1}, k-1)$.

(ii) $\ell_{k-2} \geq g(\ell_{k-1}, k-1)$, since a $C_{F_{k-1}}(\ell_{k-1})$ needs to be placed inside a part of length ℓ_{k-2}. We also need $\ell_{k-2} \geq \ell$, but this is redundant since $g(\ell_{k-1}, k-1) \geq \ell$. Thus, we can choose $\ell_{k-2} = g(\ell_{k-1}, k-1)$, and the space for $C_{F_{k-2}}(\ell_{k-2})$ is $g(\ell_{k-2}, k-2)$.

(iii) For any j, we can choose $\ell_j = g(\ell_{j+1}, j+1)$ and get that the space for $C_{F_j}(\ell_j)$ is $g(\ell_j, j)$.

(iv) The total space for the $C_F(\ell)$ is then $g(\ell_0, 0) + \ell$, because space $g(\ell_0, 0)$ suffices for the component $C_{F_0}(\ell_0)$, all other components $C_{F_j}(\ell_j)$ for $1 \leq j \leq k-1$ are placed inside it, and the final interval that is potentially presented after round $k-1$ extends by ℓ links further to the right. So we obtain $g(\ell, k) = g(\ell_0, 0) + \ell$, where ℓ_0 can be calculated as outlined above.

The above description represents an inductive definition of the function g. To summarize, we have $g(\ell, 0) = \ell + 1$. For any $k \geq 1$, if we assume that $g(\ell', k')$ is already defined for all positive ℓ' and $0 \leq k' < k$, the following equations allow us to determine $g(\ell, k)$:

$$
\begin{aligned}
\ell_{k-1} &= \ell \\
\ell_j &= g(\ell_{j+1}, j+1) \text{ for } 0 \leq j \leq k-2 \\
g(\ell, k) &= g(\ell_0, 0) + \ell
\end{aligned}
$$

To get an explicit representation of $g(\ell, k)$, we proceed as follows. Define the sequence a_n for $n \geq 0$ 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 [20], sequences A000058 and A007018).

Lemma 4.7. *For $\ell > 0$ we have $g(\ell, k) = \ell + 1$ for $k = 0$ and $g(\ell, k) = a_k(\ell + 1) - 1$ for $k > 0$.*

Proof. The claim for $k = 0$ is clear. Let $k > 0$. Assume that the claim has been shown for $g(\ell, k')$ for all $k' < k$. Consider the calculation of the values ℓ_j that determine $g(\ell, k)$. We have $\ell_{k-1} = \ell$. We claim that

$$\ell_j = a_{j+1} a_{j+2} \cdot \ldots \cdot a_{k-1}(\ell + 1) - 1$$

for $j = 0, \ldots, k-1$. For $j = k-1$ the claim is true (considering the product $a_{j+1} \cdot \ldots \cdot a_{k-1}$ to be empty and equal to 1). Consider some $j < k-1$. We know that $\ell_j = g(\ell_{j+1}, j+1)$. With $\ell_{j+1} = a_{j+2} a_{j+3} \cdot \ldots \cdot a_{k-1}(\ell + 1) - 1$ and $g(\ell_{j+1}, j+1) = a_{j+1}(\ell_{j+1} + 1) - 1$, we obtain

$$
\begin{aligned}
\ell_j &= a_{j+1}((a_{j+2} a_{j+3} \cdot \ldots \cdot a_{k-1}(\ell + 1) - 1) + 1) - 1 \\
&= a_{j+1} a_{j+2} a_{j+3} \cdot \ldots \cdot a_{k-1}(\ell + 1) - 1,
\end{aligned}
$$

showing the claim. So we get that $\ell_0 = a_1 a_2 a_3 \cdot \ldots \cdot a_{k-1}(\ell + 1) - 1$ and

$$
\begin{aligned}
g(\ell, k) &= g(\ell_0, 0) + \ell \\
&= \ell_0 + 1 + \ell \\
&= (a_1 a_2 a_3 \cdot \ldots \cdot a_{k-1})(\ell + 1) - 1 + 1 + \ell \\
&= (a_1 a_2 a_3 \cdot \ldots \cdot a_{k-1} + 1)(\ell + 1) - 1 \\
&= a_k(\ell + 1) - 1
\end{aligned}
$$

This shows that the statement of the lemma is true for $g(\ell, k)$. $\square$

4.3.2. The optimal coloring. We need to show that the intervals in a $C_F(\ell)$ presented by the adversary can be colored with 2 colors by an optimal offline algorithm.

A brief outline of how such a coloring can be obtained is as follows. By leaving enough free capacity in both colors, we will ensure that the coloring of a $C_F(\ell)$ at L does not constrain the coloring of intervals starting to the left of L in any way. Now, consider a $C_F(\ell)$ at 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. Assuming that the inductive hypothesis holds for the components $C_{F'}(\ell')$ that were part of the construction of the $C_F(\ell)$, we can then obtain a coloring for $C_F(\ell)$ by coloring these components from right to left.

Now we give a detailed proof. Our coloring method ensures that the coloring of intervals in a $C_F(\ell)$ at L is essentially independent of the coloring of intervals with left endpoint smaller than L (which will be colored later). By construction, the total bandwidth of intervals with left endpoint smaller than L that contain L is bounded by $\beta_L = 3^{L-1}$. Furthermore, we will color the intervals with left endpoint L or larger in such a way that a free capacity of at least β_L is left for each of the two colors on every edge used by such intervals. Hence, once the set of intervals with left endpoint L or larger is colored, the remaining intervals (with left endpoint smaller than L) can be colored independently without any danger of conflict.

Consider any component $C_F(\ell)$ placed at some link L (i.e., with leftmost 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. Recall that the intervals from $C_F(\ell)$ that contain R' and are different from I are called the siblings of I. Let E denote the rightmost link in the part of the line assigned to $C_F(\ell)$, i.e. $E = L + g(\ell, |F|) - 1$. Note that all intervals presented after $C_F(\ell)$ will have their left endpoints on R' or to the right of it. The total bandwidth of intervals presented before $C_F(\ell)$ that contain L is at most 3^{L-1}, by Invariant 1. The total bandwidth of intervals from $C_F(\ell)$ and from intervals presented earlier that contain R' is then at most $3^{R'-1}$, by Invariant 2.

LEMMA 4.8. *Every $C_F(\ell)$ can be colored with 2 colors such 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 from L to E.*

Proof. We prove the claim by induction on $|F|$. Consider the case $|F| = 0$. $C_F(\ell)$ consists of a single interval of bandwidth $3^L - 3^{L-1}$ and obviously satisfies the properties.

Assume $|F| = k > 0$. Consider the last interval I of $C_F(\ell)$. If I was presented after round $k-1$ (i.e., if the algorithm has used only colors 1 to k in the components $C_{F_j}(\ell_j)$ for $j = 0, \ldots, k-1$), assign color 1 to I, and assign color 2 to all intervals from $C_F(\ell)$ that intersect I. Note that the latter are the last intervals of the components $C_{F_j}(\ell_j)$ that were used in the construction of $C_F(\ell)$, and their siblings. As I has bandwidth $3^{R_{k-1}} - 3^{L-1}$, where R_{k-1} is its leftmost link, we have that color 1 has a free capacity of 3^{L-1} on all links of I. Furthermore, the total bandwidth of the remaining intervals from the $C_F(\ell)$ that contain R_{k-1} is at most $3^{R_{k-1}-1} - 3^{L-1}$ (a consequence of Invariant 2), and hence we have a free capacity of at least 3^{L-1} also for color 2 on all links of I. We will argue about the free capacity in each of the two colors on the links to the left of I below.

If I was not presented after round $k-1$, the construction has ended early and I was actually the last interval of one of the components $C_{F_j}(\ell_j)$. In this case, assign color 2 to I and to all intervals of $C_F(\ell)$ overlapping any of the rightmost ℓ_j links of I. These intervals are the last intervals of the components $C_{F_j}(\ell_j)$ used in the construction of $C_F(\ell)$, and their siblings.

In both cases, we have assigned color 2 to the last intervals and their siblings in all the components $C_{F_j}(\ell_j)$ that have been used in the construction of $C_F(\ell)$. By the induction hypothesis, each of these $C_{F_j}(\ell_j)$ has a coloring with two colors in which the last interval and its siblings receive the same color. Hence, we can use that coloring (after exchanging colors 1 and 2 if necessary) to complete the coloring of $C_F(\ell)$.

We have to show that the coloring is feasible and leaves a free capacity of at least 3^{L-1} in each of the two colors. Let L_j denote the leftmost link of $C_{F_j}(\ell_j)$, i.e., the $C_{F_j}(\ell_j)$ is placed at L_j. By the induction hypothesis, we know that the coloring of $C_{F_j}(\ell_j)$ leaves a free capacity of 3^{L_j-1} in each of the two colors. Furthermore, the total bandwidth of intervals from all $C_{F_{j'}}(\ell_{j'})$ for $j' < j$ together that overlap L_j is bounded by $3^{L_j-1} - 3^{L-1}$ (because Invariant 2 holds for all these components). Therefore, each of the two colors will still have a free capacity of at least $3^{L_j-1} - (3^{L_j-1} - 3^{L-1}) = 3^{L-1}$ on all links of $C_{F_j}(\ell_j)$, as required. The argument applies to all $C_{F_j}(\ell_j)$ used in the construction of $C_F(\ell)$, and so we have that the coloring is feasible and satisfies the claimed properties. $\square$

4.3.3. Lower bound result. By Lemma 4.8, we know that any $C_F(\ell)$ can be colored optimally with at most two colors. For any $k \geq 1$, we can let $F = \{1, 2, \ldots, k-1\}$ and place a $C_F(1)$ starting at link 1. The on-line algorithm $\mathcal{A}$ uses at least k colors 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 4.9. *There is no deterministic on-line algorithm for capacitated interval coloring with non-uniform capacities that achieves a 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.

5. Concluding remarks. We have considered the problem of online capacitated interval coloring with bandwidth. For the case $b_{\max} \leq c_{\min}$, we have presented a 78-competitive algorithm. For the general case, we have given an $O(\log \frac{b_{\max}}{c_{\min}})$-competitive algorithm and, using resource augmentation by a factor of $1+\varepsilon$, a constant competitive algorithm. Note that it is not difficult to design an $O(n)$-competitive algorithm (without resource augmentation). 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.

We have also presented a lower bound showing that no deterministic online algorithm can have competitive ratio better than $O(\log\log n)$ or $O(\log\log\log \frac{c_{\max}}{c_{\min}})$ for the general case.

It is possible to consider special cases of the problem, for which the lower bound possibly does not hold. Candidates for such cases are variants which are easy variants of bin packing.

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, 2003, pp. 1–12.

[2] U. Adamy, T. Erlebach, D. Mitsche, I. Schurr, B. Speckmann, and E. Welzl, *Off-line admission control for advance reservations in star networks*, in Poceedings of the Second Workshop on Approximation and Online Algorithms (WAOA'04), LNCS 3351, 2005, pp. 221–224.

[3] 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), 1999, pp. 1–10.

[4] Y. Azar, A. Fiat, M. Levy, and N. Narayanaswamy, *An improved algorithm for online coloring of intervals with bandwidth*, Theoretical Computer Science, 363 (2006), pp. 18–27.

[5] 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), 2006, pp. 721–729.

[6] 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, 2002, pp. 51–66.

[7] 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, 2003, pp. 410–425.

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

[9] 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, 2005, pp. 602–613.

[10] ———, *Online interval coloring with packing constraints*, Theoretical Computer Science, 407 (2008), pp. 203–212.

[11] T. Erlebach, *Call admission control for advance reservation requests with alternatives*, in Proceedings of the 3rd Workshop on Approximation and Randomization Algorithms in Communication Networks (ARACNE'02), Carleton Scientific, 2002, pp. 51–64.

[12] A. Gyárfás and J. Lehel, *On-line and first-fit colorings of graphs*, Journal of Graph Theory, 12 (1988), pp. 217–227.

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

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

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

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

[17] 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), 2004, pp. 329–338.

[18] 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), 2004, pp. 562–571.

[19] 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), 1994, pp. 134–143.

[20] N. Sloane, *On-line encyclopedia of integer sequences*, 1996–2005. Available on-line at http://www.research.att.com/∼njas/sequences/Seis.html.