Complexity and Online Algorithms for Minimum Skyline Coloring of Intervals

Thomas Erlebach[1⋆], Fu-Hong Liu[2], Hsiang-Hsuan Liu[3⋆⋆], Mordechai Shalom[4], Prudence W.H. Wong[5], and Shmuel Zaks[6]

[1] Department of Informatics, University of Leicester, UK.
t.erlebach@leicester.ac.uk
[2] Department of Computer Science, National Tsing Hua University, Taiwan
fhliu@cs.nthu.edu.tw
[3] Institute of Computer Science, University of Wroclaw, Poland
[4] TelHai College, Upper Galilee, 12210, Israel cmshalom@telhai.ac.il
[5] Department of Computer Science, University of Liverpool, UK
pwong@liverpool.ac.uk
[6] Department of Computer Science, Technion, Haifa, Israel zaks@cs.technion.ac.il

Abstract. Graph coloring has been studied extensively in the literature. The classical problem concerns the number of colors used. In this paper, we focus on coloring intervals where the input is a set of intervals and two overlapping intervals cannot be assigned the same color. In particular, we are interested in the setting where there is an increasing cost associated with using a higher color index. Given a set of intervals (on a line) and a coloring, the cost of the coloring at any point is the cost of the maximum color index used at that point and the cost of the overall coloring is the integral of the cost over all points on the line. The objective is to assign a valid color to each interval and minimize the total cost of the coloring. Intuitively, the maximum color index used at each point forms a skyline and so the objective is to obtain a minimum skyline coloring. The problem arises in various applications including optical networks and job scheduling.

Alicherry and Bhatia defined in 2003 a more general problem in which the colors are partitioned into classes and the cost of a color depends solely on its class. This problem is NP-hard and the reduction relies on the fact that some color class has more than one color. In this paper we show that when each color class only contains one color, this simpler setting remains NP-hard via a reduction from the arc coloring problem. In addition, we initiate the study of the online setting and present an asymptotically optimal online algorithm. We further study a variant of the problem in which the intervals are already partitioned into sets and the objective is to assign a color to each set such that the total cost is minimum. We show that this seemingly easier problem remains NP-hard by a reduction from the optimal linear arrangement problem.

⋆ Supported by a study leave granted by University of Leicester.
⋆⋆ Partially supported by Polish National Science Centre grant 2016/22/E/ST6/00499 and partially supported by a Dual PhD studentship when the author was with University of Liverpool and National Tsing Hua University.

1 Introduction

Graph coloring has been studied extensively in the literature [16]. In the basic problem, given a graph we have to color its vertices such that no two adjacent vertices are assigned the same color. The classical version of the problem concerns the number of colors used. Many different variants of the problem have been studied, e.g., coloring edges instead of vertices, focusing on different graph classes, and concerning different objective functions [7, 11, 12, 15, 16, 19, 20, 23].

In this paper, we focus on coloring of intervals [14] in which the input is a set of intervals on a line and two overlapping intervals cannot be assigned the same color. This corresponds to a coloring of an interval graph in the classical sense, however our cost measure is different, as follows. We are interested in the setting where there is an increasing cost associated with using a higher color index. Given a set of intervals (represented on a line) and a coloring, the cost of the coloring at any point is the cost of the maximum color index used at that point and the overall cost of the coloring is the integral of the cost over all points on the line. Intuitively, the maximum color index used at each point forms a "skyline" and so the objective is to obtain a minimum skyline coloring. A more formal definition of skyline will be given in Section 2.

The problem arises in various applications. In communication networks like optical networks in the line topology, a network needs to be equipped with optical amplifier devices for transmitting data through the optical fiber. The devices are increasingly more complicated when we need a higher wavelength (cf. color), and hence require a higher cost to operate; and each type of amplifier device is capable of amplifying all the wavelengths up to a certain maximum. Therefore, the cost of operation depends on the maximum wavelength which is reflected in the cost of the maximum color index defined in our problem. See [2] for a more detailed discussion.

Another application is from job scheduling, where each job has a required execution interval and has to be assigned to a machine. The machines are in an ordered list and one must at any time hire a set of machines that is a prefix of that ordered list. This means that if the machine of the largest index that one currently uses is machine k, one must pay the rental cost for the first k machines.

Related work. Interval scheduling was first studied with the objective of minimizing the number of colors used [7, 20]. Generalizations considered include minimizing the sum of the colors assigned to the vertices [11,12,15,19]; incorporating a bandwidth requirement for each interval and allowing overlapping intervals to be assigned the same color as long as their total bandwidth requirement does not exceed the capacity [1, 3]. The work most relevant to this paper includes generalized coloring problems studied in [2, 17, 23] and the busy time scheduling problems [5, 6, 8, 13, 18, 22].

In [2], a more general interval coloring problem is defined in which the set of colors is divided into color classes and each color class C_i has a cost of i. At any point on the line, if a color in C_i is the largest color assigned to some interval containing the point, then the cost at this point is i. The authors prove that this problem is NP-hard via a reduction from Numerical Three Dimensional Matching.

This reduction requires that some color class has more than one color in the class. A 2-approximation algorithm is also proposed in the paper. In the busy time scheduling problem [5, 6, 8, 13, 18, 22], a machine (cf. color) can be shared by a certain number of jobs (cf. intervals) and the usage of a machine costs the same no matter how many jobs are sharing the machine. The busy time problem can also be presented as other equivalent problems, e.g., in the context of optical line network wavelength assignment [17, 23] and dynamic bin packing with minimum server usage time [21].

Our contribution. The problem we study in this paper is a special case of the problem in [2] in which each color class consists of one color. Yet we prove a stronger NP-hardness result revealing that the problem remains NP-hard (Section 3). The proof is via a reduction from the ArcColoring problem [10]. We then initiate the study of the online setting for the problem (Section 4) and present an $O(1 + \log \frac{\ell_{\max}}{\ell_{\min}})$-competitive algorithm where $\ell_{\max}$ and $\ell_{\min}$ are the maximum and minimum length of the intervals. The algorithm assumes the knowledge of $\frac{\ell_{\max}}{\ell_{\min}}$ in advance. We also show a lower bound of $\frac{1}{2} \log \frac{\ell_{\max}}{\ell_{\min}}$ on the competitive ratio for any deterministic online algorithm even when the algorithm knows $\frac{\ell_{\max}}{\ell_{\min}}$ in advance. This implies that our online algorithm is asymptotically optimal. In addition, we extend our results to the case when each color has a positive capacity κ and can be assigned to a set of intervals with load at most κ (Section 5.1) showing that the online algorithm applies with only a constant factor increase in the competitive ratio. On the other hand, if the cost function is an arbitrary increasing function instead of linear in the class index, then any deterministic online algorithm can perform very badly (Section 5.2). We also note that our online algorithm applies when the underlying graph is a circular graph instead of a line (Section 5.3).

The coloring problem essentially consists of two components: partitioning the intervals into disjoint subsets such that in each subset no two intervals overlap; and assigning a color to each subset. We consider a variant of this problem in which the subsets are given and the only decision is to assign a different color to each subset, i.e., find a permutation of the subsets to map to color 1, 2, $\cdots$. At first glance this permutation problem may sound easier. Nevertheless, we show that the permutation problem is NP-hard (Section 6) by presenting a reduction from the optimal linear arrangement problem [9].

2 Definitions and preliminaries

Problem definition. We are given a set of n intervals $\mathcal{I} = \{I_1, I_2, \cdots, I_n\}$. Each I_j is a half open interval $[s_j, e_j)$, where s_j and e_j denote real numbers that are the start and end point of the interval I_j, respectively. Two intervals I_i and I_j are *overlapping* if $I_i \cap I_j \neq \emptyset$. Interval I_j contains point t if $t \in I_j$, i.e., $s_j \leq t < e_j$. We denote by $\mathcal{I}_t$ the set of intervals of $\mathcal{I}$ that contain point t, i.e., $\mathcal{I}_t = \{I_j \in \mathcal{I} | I_j \ni t\}$, and by $load_{\mathcal{I}}(t)$ the number $|\mathcal{I}_t|$ of these intervals which is termed the *load* induced by $\mathcal{I}$ at point t. When there is no ambiguity, we omit the subscript $\mathcal{I}$ and simply write $load(t)$. The *length* of I_j, denoted by $\ell(I_j)$, is

defined as $e_j - s_j$. The maximum and minimum lengths over all intervals in $\mathcal{I}$ are denoted by $\ell_{\max}$ and $\ell_{\min}$, respectively. The length $\ell(\mathcal{S})$ of a set $\mathcal{S}$ of intervals is the sum of the lengths of all intervals in $\mathcal{S}$, i.e., $\ell(\mathcal{S}) = \sum_{I \in \mathcal{S}} \ell(I)$.

We are also given an infinite set of colors $\Lambda = \{1, 2, 3, \cdots\}$, and every color i has an associated cost $\lambda(i) \geq 1$, where λ is a non-decreasing function of i. A coloring $\omega : \mathcal{I} \to \Lambda$ is *valid* if for any pair of distinct overlapping intervals I_i and I_j, we have $\omega(I_i) \neq \omega(I_j)$. We refer to the coloring of the intervals in $\mathcal{I}$ as $\omega(\mathcal{I})$. For any subset $\mathcal{I}' \subseteq \mathcal{I}$, we denote by $\omega(\mathcal{I}')$ the coloring obtained by restricting ω to the intervals $\mathcal{I}'$. The instantaneous cost of ω at point t, denoted by $cost(\omega, t)$, is the maximum cost of the colors of all intervals containing t, i.e., $cost(\omega, t) = \max_{I \in \mathcal{I}_t} \lambda(\omega(I))$ if $\mathcal{I}_t \neq \emptyset$ and zero otherwise. Note that $cost(\omega, t) = 0$ when $load(t) = 0$. Since λ is a non-decreasing function, we have $cost(\omega, t) = \lambda(\max_{I \in \mathcal{I}_t} \omega(I))$. We term this color (i.e., $\max_{I \in \mathcal{I}_t} \omega(I)$), as the *skyline* of ω at t, and the unique interval of $\mathcal{I}_t$ colored with this color, as the *contributing* interval of ω at t. We denote the set of all contributing intervals by $\mathcal{I}_{\mathrm{s}}$, i.e., an interval I is in $\mathcal{I}_{\mathrm{s}}$ if there exists $t \in I$ such that $cost(\omega, t) = \lambda(\omega(I))$, or equivalently, $I = \arg\max_{I \in \mathcal{I}_t} \omega(I)$.

The total cost of ω, denoted as $cost(\omega)$, is the integral of all the instantaneous costs, i.e., $cost(\omega) = \int_{-\infty}^{\infty} cost(\omega, t) \mathrm{d}t$. From our definitions it follows that $cost(\omega(\mathcal{I})) = cost(\omega(\mathcal{I}_{\mathrm{s}}))$. Moreover, when ω is a valid coloring we have $\max_{I \in \mathcal{I}_t} \omega(I) \geq load(t)$, since the intervals of $\mathcal{I}_t$ are colored with distinct colors. Therefore, $cost(\omega, t) \geq \lambda(load(t))$, and consequently, $cost(\omega) \geq \int_{-\infty}^{\infty} \lambda(load(t)) \mathrm{d}t$. A valid coloring for which the last inequality is tight is clearly optimal. We term such colorings as *load-optimal*. From the definitions it follows:

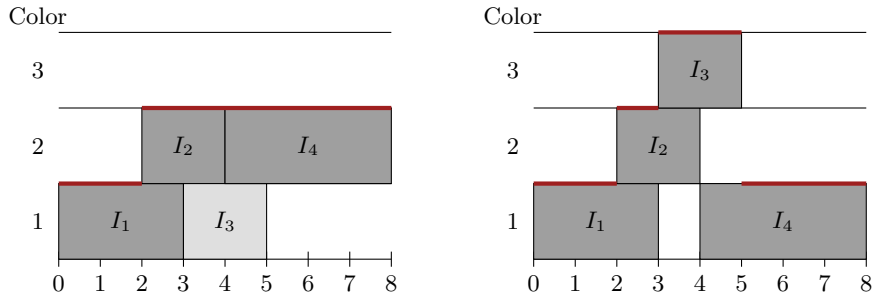
Observation 1 *A valid coloring ω of an input set of intervals $\mathcal{I}$ is load-optimal if and only if for every point t, the set of colors used by ω for intervals in $\mathcal{I}_t$ is $\{1, \ldots, load(t)\}$.*

In this work, unless otherwise specified we assume $\lambda(i) = i$. Whenever this is the case we have $\int_{-\infty}^{\infty} \lambda(load(t)) \mathrm{d}t = \int_{-\infty}^{\infty} load(t) \mathrm{d}t = \ell(\mathcal{I})$. The last equality is due to the fact that every infinitesimal subinterval of an interval in $\mathcal{I}$ contributes the same value (namely, its length) to both sides. This implies:

Observation 2 *For every valid coloring ω of an input set of intervals $\mathcal{I}$, we have $cost(\omega) \geq \ell(\mathcal{I})$ when $\lambda(i) = i$ for all i.*

The objective of the Skyline problem is to find a valid coloring ω such that $cost(\omega)$ is minimized. Without loss of generality, we can assume that the union $\cup\mathcal{I}$ of the intervals in $\mathcal{I}$ is an interval that we term *the horizon*. Otherwise, the coloring of each maximal interval of $\cup\mathcal{I}$ is independent of the others. Figure 1 illustrates various notions used in the problem definition.

Online algorithms. In this paper we focus on the online setting where intervals arrive one at a time in an arbitrary order. An online algorithm has to decide on the color of an interval upon its arrival, and this decision cannot be modified later. Such an algorithm is *c-competitive* if for every input the cost of the solution of the algorithm is no more than c times that of an optimal (offline)



(a) Assigning the lowest available color. The cost is $2 \times 1 + 6 \times 2 = 14$.

(b) Optimal coloring. The cost is $5 \times 1 + 1 \times 2 + 2 \times 3 = 13$.

Fig. 1: Two different colorings of four intervals. An optimal solution does not necessarily minimize the number of colors used. The darker intervals contribute to the cost of the coloring but the lighter interval does not. The bolded line indicates the skyline.

solution [4]. We also denote by $\mathcal{A}$ the coloring returned by an algorithm $\mathcal{A}$, and the cost of this solution by $cost(\mathcal{A})$. We denote by $\mathcal{O}$ an optimal solution.

3 NP-hardness of Skyline

Theorem 3. *It is NP-complete to decide whether a given instance of* Skyline *has a load-optimal coloring.*

Proof. It is easy to verify whether a given coloring is load-optimal. The NP-hardness is proved by a reduction from ArcColoring. An instance of ArcColoring is given by a family $\mathcal{F} = \{A_1, \ldots, A_n\}$ of circular arcs and a positive integer K. Each arc $A_i \in \mathcal{F}$ is given by a pair (a_i, b_i) with $a_i \neq b_i$ and $a_i, b_i \in \{1, \ldots, m\}$ for some $m \leq 2n$. Intuitively, the set $\{1, \ldots, m\}$ represents points that are located around a circle. The *span* of arc A_i is the set $\{a_i, a_i + 1, \ldots, b_i - 1\}$ if $a_i < b_i$ and $\{a_i, a_i + 1, \ldots, m\} \cup \{1, \ldots, b_i - 1\}$ if $b_i < a_i$. We say that two arcs *intersect* if their spans have a non-empty intersection. It is NP-hard to decide whether the arcs in $\mathcal{F}$ can be colored with at most K colors in such a way that arcs with the same color do not intersect [10]. Let an instance $(\mathcal{F}, K)$ of ArcColoring be given. We say that a point $p \in \{1, \ldots, m\}$ is contained in an arc if it is contained in the span of the arc. Without loss of generality, we can assume that every point is contained in exactly K arcs: If a point is contained in more than K arcs, the instance is trivially a no-instance. If a point p is contained in fewer than K arcs, we can add arcs of the form $(p, p + 1)$ until p is contained in K arcs, without changing the K-colorability of the instance.

We construct an instance $\mathcal{I}$ of Skyline from $(\mathcal{F}, K)$ as follows. Intuitively, we "cut" the ring at the point 1 to turn the set of arcs into a set of intervals. The intervals resulting from arcs that were cut are then extended (into a "left

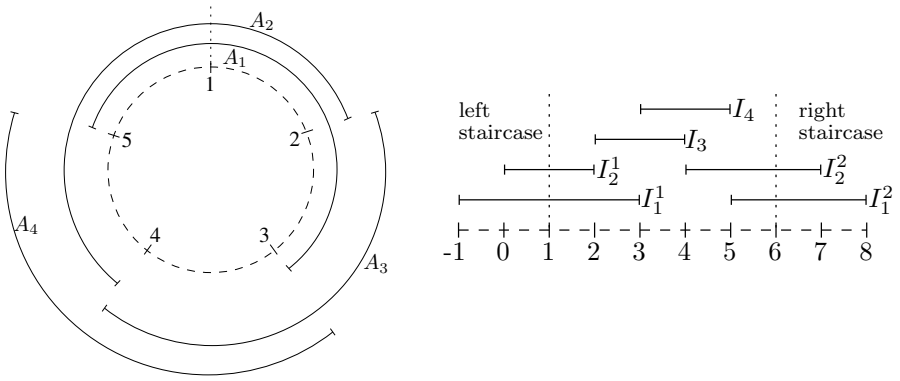


Fig. 2: Instance of ArcColoring (left), constructed intervals (right)

staircase" and a "right staircase") in such a way that the two intervals resulting from the same arc must receive the same color in any load-optimal coloring. Formally, we create intervals from the arcs in $\mathcal{F}$ as follows: Any arc $A_i = (a_i, b_i)$ that does not contain the point 1 produces the interval $I_i = [a_i, b_i)$ if $b_i > a_i$, or the interval $I_i = [a_i, m+1)$ if $b_i = 1$. Let $A_{j_1}, \ldots, A_{j_K}$ be the K arcs that contain point 1. For $1 \leq i \leq K$, the arc $A_{j_i} = (a_{j_i}, b_{j_i})$ produces two intervals $I^1_{j_i}$ and $I^2_{j_i}$: If $a_{j_i} > b_{j_i}$, the two intervals are $I^1_{j_i} = [-K+i, b_{i_j})$ and $I^2_{j_i} = [a_{i_j}, m+2+K-i)$. If $a_{j_i} = 1 < b_{j_i}$, the two intervals are $I^1_{j_i} = [-K+i, b_{i_j})$ and $I^2_{j_i} = [m+1, m+2+K-i)$. An example of the construction is shown in Figure 2. The arcs in the example are $A_1 = (5,3)$, $A_2 = (4,2)$, $A_3 = (2,4)$ and $A_4 = (3,5)$, and $K = 2$. A_1 and A_2 contain the point 1 and thus produce two intervals each, while A_3 and A_4 produce only one interval. In this example, a load-optimal coloring exists: Color I^1_1 and I^2_1 with 1, I^1_2 and I^2_2 with 2, I_3 with 2, and I_4 with 1.

We claim that $\mathcal{I}$ has a load-optimal coloring if and only if $(\mathcal{F}, K)$ is a yes-instance of ArcColoring. For the "if" direction, let $\omega : \mathcal{F} \to \{1, \ldots, K\}$ be a K-coloring of $\mathcal{F}$. We can rename the colors so that $\omega(A_{j_i}) = i$ for $1 \leq i \leq K$. Let $\omega' : \mathcal{I} \to \{1, \ldots, K\}$ map each interval in $\mathcal{I}$ to the color assigned by ω to the arc from which the interval was produced. First, note that ω' is a feasible coloring of $\mathcal{I}$ since any two intervals that intersect are produced from arcs that intersect and hence their colors are different. We claim that ω' is a load-optimal coloring of $\mathcal{I}$. For $t \in [-K+r, -K+r+1)$ for some $r \in \{1, 2, \ldots, K\}$, the only intervals containing t are the r intervals $I^1_{j_i}$ for $1 \leq j \leq r$, and these intervals have colors $1, \ldots, r$. Similarly, for $t \in [m+1+K-r, m+2+K-r)$ for some $r \in \{1, 2, \ldots, K\}$, the only intervals containing t are the r intervals $I^2_{j_i}$ for $1 \leq j \leq r$, and these intervals have colors $1, \ldots, r$. All points $t \in [1, m+1)$ are contained in exactly K intervals that receive colors $1, \ldots, K$. Hence, ω' is indeed load-optimal.

For the "only if" direction, let ω' be a load-optimal coloring of $\mathcal{I}$. The points in $[-K+1, -K+2)$ and $[m+K, m+K+1)$ are contained only in the intervals $I^1_{j_1}$ and $I^2_{j_1}$, respectively, and hence these two intervals must both receive color 1.

Similarly, as these intervals form staircase patterns, it follows that $I^1_{j_i}$ and $I^2_{j_i}$ must both receive color i, for $2 \leq i \leq K$. All other intervals must receive colors in $\{1, \ldots, K\}$ as the load at any point is at most K. Define a coloring $\omega : \mathcal{F} \to \{1, \ldots, K\}$ by assigning to each arc in $\mathcal{F}$ the color of the interval(s) it has produced (for arcs that have produced two intervals, this is still well-defined as both intervals must have the same color, as argued above). It follows that ω is a feasible K-coloring of $\mathcal{F}$. □

Combining with Observation 1 we have:

Corollary 1. SKYLINE *is NP-hard for any strictly increasing color cost function* λ.

4 Online algorithms for SKYLINE when $\lambda(i) = i$

In this section, we present online algorithms for the SKYLINE problem for the case where the cost of a color is equal to its index, i.e., $\lambda(i) = i$ for all i. We first focus in Section 4.1 on bounded length intervals and present an $O(1)$-competitive greedy algorithm. In Section 4.2 we adapt the greedy algorithm to the case where the lengths of intervals are arbitrary.

4.1 Bounded length intervals

In this section we consider bounded length intervals, i.e., we assume there is a constant k such that for any interval I in the input, we have $\ell(I) \in [\ell_{\min}, k \cdot \ell_{\min})$. This section is dedicated to the analysis of the following greedy algorithm.

The algorithm $\mathcal{G}$ and some basic properties When an interval $I_j \in \mathcal{I}$ arrives, assign the minimum color that is valid for it, i.e., the minimum color i such that for all $j' < j$ and $I_{j'} \cap I_j \neq \emptyset$, we have $\mathcal{G}(I_{j'}) \neq i$.

Roughly speaking, in the analysis, we select a subset of intervals on the skyline of $\mathcal{G}$ (i.e., from $\mathcal{I}_{\mathrm{s}}$), partition the horizon into segments based on this subset, and show that we can "charge" the cost of $\mathcal{G}$ and $\mathcal{O}$ to this subset, thus allowing us to relate the two costs. The partition of the horizon is based on the notion of extended interval. For any interval I_j, we define its *hat* interval as $I^{\mathrm{h}}_j = [s_j - k\ell_{\min}, e_j + k\ell_{\min})$ and *extended hat* interval as $I^{\mathrm{e}}_j = [s_j - 3k\ell_{\min}, e_j + 3k\ell_{\min})$. Clearly, $\ell(I^{\mathrm{e}}_j) = 6k\ell_{\min} + \ell(I_j) \leq 7k\ell_{\min}$.

We first observe a property about $\mathcal{G}$. Intuitively, when $\mathcal{G}$ assigns an interval I a certain color c, there are a substantial number of intervals overlapping with I in the input. Precisely,

Lemma 1. *Consider an interval I_j with $\mathcal{G}(I_j) = c$.* (i) *There are at least $c - 1$ intervals that overlap with I_j and are contained in I^{h}_j;* (ii) *the total length of these $c - 1$ intervals and I_j is at least $c\ell_{\min}$.*

Proof. (i) Since $\mathcal{G}$ assigns the smallest possible color to any interval, I_j gets color c only if there are already $c-1$ intervals colored by $1, 2, \cdots, c-1$ and all overlap with I_j. Since the length of any interval is bounded by $k\ell_{\min}$, for each of these intervals, its start point is at least $s_j - k\ell_{\min}$ and its end point at most $e_j + k\ell_{\min}$, meaning that they are all inside I_j^{h}.

(ii) Follows from (i) and the fact the length of any interval is at least $\ell_{\min}$. □

Analysis of $\mathcal{G}$ Overview. The analysis is based on choosing a subset of intervals $\mathcal{I}_{\mathrm{s}}^*$ of $\mathcal{I}_{\mathrm{s}}$. We first give an overview of the role of $\mathcal{I}_{\mathrm{s}}^*$ and then show how to construct $\mathcal{I}_{\mathrm{s}}^*$. The aim is to obtain the following properties: (i) the hat intervals of any two intervals of $\mathcal{I}_{\mathrm{s}}^*$ do not overlap, (ii) the union of the extended hat intervals of $\mathcal{I}_{\mathrm{s}}^*$ form a contiguous interval that contains the horizon. The first property means that we can lower bound $cost(\mathcal{O})$ by considering these hat intervals since these hat intervals are disjoint. The second property means that we can map each interval to some extended hat interval. As to be shown, the procedure of selecting $\mathcal{I}_{\mathrm{s}}^*$ further ensures that the mapping allows bounding the cost of $\mathcal{G}$.

Choosing $\mathcal{I}_{\mathrm{s}}^*$. We choose the elements of $\mathcal{I}_{\mathrm{s}}^*$ according to the following procedure. Initially $\mathcal{I}_{\mathrm{s}}^*$ is empty. We consider the intervals of $\mathcal{I}_{\mathrm{s}}$ in decreasing order of their colors, and within each color, in the order of their start points. We add the interval I under consideration to $\mathcal{I}_{\mathrm{s}}^*$ if it is not completely contained in the extended hat of an interval of $\mathcal{I}_{\mathrm{s}}^*$.

Competitiveness of $\mathcal{G}$. We now analyze the properties of $\mathcal{I}_{\mathrm{s}}^*$. We first prove the following lemma.

Lemma 2. (i) *For every interval $I_j \in \mathcal{I}_{\mathrm{s}}$, there exists an interval $I_{j'} \in \mathcal{I}_{\mathrm{s}}^*$ such that $I_j \subseteq I_{j'}^{\mathrm{e}}$ and $\mathcal{G}(I_j) \leq \mathcal{G}(I_{j'})$.* (ii) *The hat intervals of the intervals of $\mathcal{I}_{\mathrm{s}}^*$ are pairwise disjoint.*

Proof. (i) Follows from the way $\mathcal{I}_{\mathrm{s}}^*$ is chosen. Consider an interval $I_j \in \mathcal{I}_{\mathrm{s}}$. If $I_j \in \mathcal{I}_{\mathrm{s}}^*$ the claim follows. Otherwise, there is an interval $I_{j'} \in \mathcal{I}_{\mathrm{s}}^*$ such that $I_j \subseteq I_{j'}^{\mathrm{e}}$ and $I_{j'}$ is considered before I_j in the selection process. Therefore, $\mathcal{G}(I_j) \leq \mathcal{G}(I_{j'})$.

(ii) Consider any two intervals I_j and $I_{j'}$ in $\mathcal{I}_{\mathrm{s}}^*$. Assume without loss of generality that I_j is chosen before $I_{j'}$. When I_j is chosen, any intervals that are entirely contained in I_j^{e} are removed. Since $I_{j'}$ is not removed, at least one of the following conditions holds. (1) $e_{j'} > e_j + 3k\ell_{\min}$, (2) $s_{j'} < s_j - 3k\ell_{\min}$. We analyze only the case where (1) holds, the other case being symmetric. If (1) holds we have that $s_{j'} \geq e_{j'} - k\ell_{\min} > e_j + 2k\ell_{\min}$ and the left point of $I_{j'}^{\mathrm{h}}$ is $s_{j'} - k\ell_{\min} > e_j + k\ell_{\min}$. Therefore, I_j^{h} and $I_{j'}^{\mathrm{h}}$ are disjoint. □

Using Lemma 2, we can relate the cost of the greedy algorithm to the optimum.

Lemma 3. (i) $cost(\mathcal{G}) \leq 7k\ell_{\min} \cdot \sum_{I \in \mathcal{I}_{\mathrm{s}}^*} \mathcal{G}(I)$; (ii) $cost(\mathcal{O}) \geq \ell_{\min} \cdot \sum_{I \in \mathcal{I}_{\mathrm{s}}^*} \mathcal{G}(I)$; *and* (iii) $cost(\mathcal{G}) \leq 7k\ell(\mathcal{I})$.

Proof. (i) Let $\mathcal{I}_{\mathrm{s}}^{\mathrm{e}}$ be the set of extended hat intervals of $\mathcal{I}_{\mathrm{s}}^*$, i.e., $\mathcal{I}_{\mathrm{s}}^{\mathrm{e}} = \{I^{\mathrm{e}} \mid I \in \mathcal{I}_{\mathrm{s}}^*\}$, and $\mathcal{G}(\mathcal{I}_{\mathrm{s}}^{\mathrm{e}})$ be the coloring of $I^{\mathrm{e}} \in \mathcal{I}_{\mathrm{s}}^{\mathrm{e}}$ using the color of the corresponding interval

I, i.e., $\mathcal{G}(I^{\mathrm{e}}) = \mathcal{G}(I)$. Note that $\mathcal{G}$ is not necessarily a valid coloring for $\mathcal{I}_{\mathrm{s}}^{\mathrm{e}}$, but its cost is yet well defined.

By Lemma 2, for every interval $I_j \in \mathcal{I}$, there is an interval $I_{j'} \in \mathcal{I}_{\mathrm{s}}^*$ such that $\mathcal{G}(I_j) \leq \mathcal{G}(I_{j'})$. If we raise the color of I_j from $\mathcal{G}(I_j)$ to $\mathcal{G}(I_{j'})$, then the resulting skyline is of the same height or higher at every point t, in other words, $cost(\mathcal{G}(\mathcal{I}_{\mathrm{s}}), t) \leq cost(\mathcal{G}(\mathcal{I}_{\mathrm{s}}^{\mathrm{e}}), t)$ at every point t. Therefore, $cost(\mathcal{G}) = cost(\mathcal{G}(\mathcal{I}_{\mathrm{s}})) \leq cost(\mathcal{G}(\mathcal{I}_{\mathrm{s}}^{\mathrm{e}}))$. We also have $cost(\mathcal{G}(I^{\mathrm{e}})) = \ell(I^{\mathrm{e}})\mathcal{G}(I) \leq 7k\ell_{\min}\mathcal{G}(I)$ for every interval I. Therefore, $cost(\mathcal{G}(\mathcal{I}_{\mathrm{s}}^{\mathrm{e}})) \leq 7k\ell_{\min}\sum_{I\in\mathcal{I}_{\mathrm{s}}^*}\mathcal{G}(I)$.

(ii) By Lemma 1, for every interval $I \in \mathcal{I}_{\mathrm{s}}^*$, there is a set of $\mathcal{G}(I)$ intervals with total length of $\ell_{\min}\mathcal{G}(I)$ each of which is contained in I^{h}. By Lemma 2, the hat intervals of $I \in \mathcal{I}_{\mathrm{s}}^*$ are pairwise disjoint. This means the total length of all intervals is at least $\sum_{I\in\mathcal{I}_{\mathrm{s}}^*}\ell_{\min}\mathcal{G}(I)$. The statement then follows from Observation 2.

(iii) The proof of (ii) states that $\ell(\mathcal{I}) \geq \sum_{I\in\mathcal{I}_{\mathrm{s}}^*}\ell_{\min}\mathcal{G}(I)$. Then Statement (i) implies that $cost(\mathcal{G}) \leq 7k\ell(\mathcal{I})$. □

Theorem 4. *When $\lambda(i) = i$, the greedy algorithm $\mathcal{G}$ is $7k$-competitive where $k = \ell_{\max}/\ell_{\min}$.*[7]

4.2 Arbitrary length intervals

In this section we consider intervals with arbitrary lengths. We first observe in the following lemma that the greedy algorithm $\mathcal{G}$ performs badly for such instances since $\frac{\ell_{\max}}{\ell_{\min}}$ can be large.

Lemma 4. *The greedy algorithm $\mathcal{G}$ is $\Omega(\frac{\ell_{\max}}{\ell_{\min}})$-competitive.*

Proof. Consider the following instance consisting of n intervals, $I_j = [0, 1)$ for $j \in [1, n-1]$, and $I_n = [0, \ell)$. Consider the coloring ω such that $\omega(I_n) = 1$ and $\omega(I_j) = j + 1$ for every $j \in [1, n-1]$. The cost is $cost(\omega) = (\ell - 1) + n$. On the other hand, the greedy algorithm gives the following coloring: $\mathcal{G}(I_j) = j$ for $j \in [1, n-1]$ and $\mathcal{G}(I_n) = n$ and $cost(\mathcal{G}) = n\ell$. We note that the ratio $\frac{cost(\mathcal{G})}{cost(\omega)} = \frac{\ell n}{\ell - 1 + n}$ can be made arbitrarily close to $\ell = \frac{\ell_{\max}}{\ell_{\min}}$. □

The greedy algorithm performs badly against the adversary in Lemma 4 because it uses up the small colors for short intervals and then has to use a large color for the long interval. To address this issue, we would like to design an algorithm that distributes the colors among intervals of different lengths in a "fair" way.

In order to obtain a better competitive ratio, we propose the algorithm *Classify-greedy* which we denote by $\mathcal{C}$. For ease of presentation, we first assume that $\mathcal{C}$ knows in advance $\ell_{\max}$ and $\ell_{\min}$. Let $L = 1 + \lceil \log \frac{\ell_{\max}}{\ell_{\min}} \rceil$. We partition $\mathcal{I}$

[7] As was pointed out by an anonymous reviewer of a previous version of this paper, the competitive ratios can be improved to 4 when $k = 1$ and 9 when $k = 2$ by using a different algorithm, while the ratio becomes $(k + 1)^2$ for larger k. This improvement does not affect the order of the competitive ratio for the general case in Theorem 6.

into L classes $C_1, C_2, \cdots, C_L$ such that C_i contains all intervals I with $\ell(I) \in [\ell_{\min} \cdot 2^{i-1}, \ell_{\min} \cdot 2^i)$. Furthermore, we also partition the set of colors Λ into L disjoint sets, where $\Lambda_i = \{i, i+L, i+2L, i+3L, \cdots\}$, for $i \in [L]$.

Classify-greedy $\mathcal{C}$ runs L copies $\mathcal{G}_1, \ldots \mathcal{G}_L$ of $\mathcal{G}$ where $\mathcal{G}_i$ uses the set of colors Λ_i. When $I \in C_i$ arrives, it is processed by $\mathcal{G}_i$ which colors it with the smallest color in Λ_i that is valid for I.

We denote an optimal coloring of $\mathcal{I} \cap C_i$ by $\mathcal{O}_i$. The following observation is due to Lemma 3(iii) (for $k = 2$) and the fact that $\mathcal{G}_i$ uses Λ_i that contains one color per every interval of L colors.

Observation 5 *$cost(\mathcal{G}_i) \leq 14L \cdot \ell(C_i)$.*

Theorem 6. *Algorithm $\mathcal{C}$ is $O(1 + \lceil \log \frac{\ell_{\max}}{\ell_{\min}} \rceil)$-competitive.*

Proof. The cost of $\mathcal{C}$ is the integral over the horizon of the maximum color used by all copies of $\mathcal{G}$ at every point t, i.e., $cost(\mathcal{C}) = \int_{-\infty}^{\infty} \max_{i \in [L]} cost(\mathcal{G}_i, t) \mathrm{d}t \leq \int_{-\infty}^{\infty} \sum_{i=1}^{L} cost(\mathcal{G}_i, t) \mathrm{d}t = \sum_{i=1}^{L} \int_{-\infty}^{\infty} cost(\mathcal{G}_i, t) \mathrm{d}t$
$= \sum_{i=1}^{L} cost(\mathcal{G}_i) \leq \sum_{i=1}^{L} 14L \cdot \ell(C_i) = 14L \cdot \ell(\mathcal{I}) \leq 14L \cdot \mathcal{O}$. □

Knowing the ratio $\frac{\ell_{\max}}{\ell_{\min}}$ only. We now describe how the algorithm can be adapted to the setting where only the ratio $\frac{\ell_{\max}}{\ell_{\min}}$ is known instead of knowing $\ell_{\max}$ and $\ell_{\min}$. An interval of length ℓ is assigned to the class $\lceil \log_2 \ell \rceil$. In this way, the intervals are assigned to at most $L + 1$ length classes with consecutive indices though the indices may not be from 1 to $L + 1$. The set of colors Λ is now divided into $L + 1$ disjoint sets $\Lambda_1, \Lambda_2, \cdots, \Lambda_{L+1}$ (note that set $\Lambda_i = \{i, i + L + 1, i + 2(L + 1), i + 3(L + 1), \cdots\}$). When an interval in a new length class is released, we map this length class to the next color set. We note that Observation 5 remains correct with $L = 2 + \lceil \log \frac{\ell_{\max}}{\ell_{\min}} \rceil$ and Theorem 6 follows with competitive ratio $14(2 + \lceil \log \frac{\ell_{\max}}{\ell_{\min}} \rceil)$, which is still $O(1 + \lceil \log \frac{\ell_{\max}}{\ell_{\min}} \rceil)$.

4.3 Lower bound

In this section, we present an adversary to show a lower bound for any deterministic online algorithm that asymptotically matches the upper bound shown in Section 4.2 for Classify-greedy.

Theorem 7. *No deterministic online algorithm can achieve competitive ratio better than $\frac{1}{2} \log \frac{\ell_{\max}}{\ell_{\min}}$ even if it knows $\frac{\ell_{\max}}{\ell_{\min}}$ in advance. This holds even when the intervals are released from left to right and even for special instances including proper instances and laminar instances.* [8]

Proof. Let $\mathcal{A}$ be an online algorithm. Let L be an arbitrarily large positive integer. The adversary creates an instance $\mathcal{I}$ with $\frac{\ell_{\max}}{\ell_{\min}} = 2^L$, or equivalently $\log \frac{\ell_{\max}}{\ell_{\min}} = L$,

[8] An instance is a proper instance if for any two intervals I_1 and I_2, $s_1 \leq s_2$ implies $e_1 \leq e_2$. An instance is a laminar instance if any two intervals are either disjoint or one is completely contained in another.

as follows. The instance will be such that it is easy to see that a load-optimal coloring exists.

The adversary releases a sequence of up to L intervals $I_j = [0, 2^j)$ for $j = 1, 2, \ldots, L$. If the algorithm uses color $L+1$ for one of them, say for interval I_k, the adversary stops the sequence and presents only one more final interval $I_f = [2^k, 2^k + \frac{1}{2^{L-k}})$. Note that $\ell_{\max} = 2^k$ and $\ell_{\min} = 2^{-(L-k)}$, so $\frac{\ell_{\max}}{\ell_{\min}} = 2^L$. We have $cost(\mathcal{A}) \geq (L+1)2^k$ and $cost(\mathcal{O}) = \ell(\mathcal{I}) < 2^{k+1}$, so $\frac{cost(\mathcal{A})}{cost(\mathcal{O})} > \frac{L+1}{2} > \frac{L}{2}$.

If the algorithm does not use color $L+1$ on the L intervals of the sequence, it must use colors $1, \ldots, L$ on these intervals as they all overlap. The adversary then presents one more interval $I_{L+1} = [0, 2^{L+1})$, which must receive color at least $L+1$. Note that $\ell_{\max} = 2^{L+1}$ and $\ell_{\min} = 2$, so $\frac{\ell_{\max}}{\ell_{\min}} = 2^L$. We have $cost(\mathcal{A}) \geq (L+1)2^{L+1}$ and $cost(\mathcal{O}) = \ell(\mathcal{I}) \leq 2^{L+2}$, so $\frac{cost(\mathcal{A})}{cost(\mathcal{O})} \geq \frac{L+1}{2} > \frac{L}{2}$.

The above instance is a laminar instance. We can make a proper instance by slight modification: let ϵ be a very small positive value; then I_j is set to $[(j-1)\epsilon, 2^j + (j-1)\epsilon)$ and I_f is set to $[2^k + (k-1)\epsilon, 2^k + \frac{1}{2^{L-k}} + (k-1)\epsilon)$. In both cases, intervals are released from left to right. □

5 Extensions

5.1 Uniform color capacity

We consider the extension where each color has a "capacity" κ: it is allowed to have κ overlapping intervals sharing the same color at the same point. A coloring $\omega : \mathcal{I} \to \Lambda$ is *valid* if for any $c \in \Lambda$ and at any point t, there are at most κ intervals $I \in \mathcal{I}_t$ with $\omega(I) = c$. In this case, we show that we can adapt the algorithms in Section 4 with a constant factor increase in the competitive ratio.

Adapted algorithms. First, we observe that Observation 2 can be adapted to $cost(\omega) \geq \frac{\ell(\mathcal{I})}{\kappa}$ because there can be at most κ intervals sharing a color at any point t, i.e., $cost(\omega, t) \geq \lceil \frac{load(t)}{\kappa} \rceil$, and $\int_{-\infty}^{\infty} load(t)\mathrm{d}t = \ell(\mathcal{I})$. The color assignment of the greedy algorithm $\mathcal{G}$ remains the same except that the condition of valid coloring is now adapted as above to allow κ intervals sharing a color. Then the algorithm Classify-greedy $\mathcal{C}$ is exactly the same as before, but using the adapted $\mathcal{G}$. The analysis is also similar but more involved. We give here a high level description of the adapted analysis and we elaborate on the details in the full paper.

Adapted analysis. In the analysis of $\mathcal{G}$, we rely on the fact that when $\mathcal{G}$ assigns color c to an interval I_j, there are a substantial number of intervals overlapping with I_j in the input (see Lemma 1). With the capacity, we show a variant of Lemma 1: the length of I_j plus the total length of the $(c-1)\kappa$ intervals that overlap with I_j and are contained in I_j^{h} is at least $((c-1)\kappa + 1) \cdot \ell_{\min}$. The main difference of this property is that we are no longer able to show that the total length of overlapping intervals is $c\kappa\ell_{\min}$, i.e., we have $(c-1)\kappa\ell_{\min}$ instead of the desired $c\kappa\ell_{\min}$. Recall that the analysis uses a charging scheme that maps intervals on the skyline to certain intervals in the optimal coloring. Most of the

analysis still carries forward except for intervals that are colored with color 1 because in such case the new bound only guarantees that a total length of $\ell_{\min}$ intervals overlap with such an interval (instead of the desired $\kappa\ell_{\min}$). Yet for intervals that are colored with color 1, this means that the optimal algorithm also needs to use at least color 1 because there is indeed an interval.

Roughly speaking, we divide the analysis into two parts: (i) $\mathcal{I}_s^*$ is selected from highest color until color 2 (instead of color 1) with the same analysis as before and (ii) remaining skyline intervals colored with color 1 are compared directly to the optimal coloring. We prove

Theorem 8. (Adapted Theorem 4) *When $\lambda(i) = i$, the greedy algorithm $\mathcal{G}$ is $(14\ell_{\max}/\ell_{\min} + 1)$-competitive for the uniform color capacity setting.*

Similarly, the analysis of $\mathcal{C}$ also takes the approach of dividing the horizon into two parts: those with intervals colored higher than color L and those with intervals colored L or lower; the latter corresponds to color 1 for each length class. Precise definitions of the partition and the detailed analysis are given in the full paper where we prove

Theorem 9. (Adapted Theorem 6) *Algorithm $\mathcal{C}$ is $O(1+\lceil \log \frac{\ell_{\max}}{\ell_{\min}} \rceil)$-competitive for the uniform color capacity setting.*

5.2 Generalized color cost function

A more general problem of Skyline is to generalize the cost function of colors. In the original Skyline problem, we assume that $\lambda(i) = i$ for all colors $i \in \Lambda$. We relax this constraint by considering a bounded relative cost of the neighboring colors, i.e., $1 \leq \frac{\lambda(i+1)}{\lambda(i)} \leq \delta$. Note that for the original setting we have $\frac{\lambda(i+1)}{\lambda(i)} \leq 2$. The Skyline problem under this new setting, however, is much harder in the sense of competitive ratio. In fact, we show in Theorem 10 that the lower bound on the competitive ratio of this problem is exponential in n (proof in the full paper). On the other hand, the competitive ratio in terms of n for the original setting can be shown to be $\Theta(n)$ as follows. The greedy algorithm is $O(n)$-competitive because it colors any interval by a color at most n and so the cost of the greedy algorithm is at most $n \cdot h$ and the optimal cost is at least h, where h is the length of the horizon. A lower bound of $\Omega(n)$ follows from the construction in the proof of Theorem 7 since the number of intervals in the construction is at most $L + 1$.

Theorem 10. *Consider* Skyline. *There exists a cost function λ and some $\delta > 1$ such that $\frac{\lambda(i+1)}{\lambda(i)} \leq \delta$ and $\frac{cost(\mathcal{A})}{cost(\mathcal{O})} \geq \delta^{\Omega(n)}$.*

5.3 Circular graphs

The upper and lower bound results in Section 4 apply to circular graphs as well. For the lower bound this is obvious. For the upper bound, suppose we have a circle from label 0 running clockwise until label $\mathcal{T}$ (0 coincides with $\mathcal{T}$). An input

interval consists of a start point and an end point. If the start point has label larger than the end point, this means the interval runs across point 0. Without loss of generality, we assume that the union of the input intervals cover the entire circle, otherwise, the input can be treated as an input on a line. The algorithm $\mathcal{G}$ works the same way on a circle. The analysis needs modification for intervals crossing the point 0. For an interval $[s, e)$, the hat and extended hat interval is now defined as $[(s_j - k\ell_{\min}) \mod \mathcal{T}, (e_j + k\ell_{\min}) \mod \mathcal{T})$ and $[(s_j - 3k\ell_{\min}) \mod \mathcal{T}, (e_j + 3k\ell_{\min}) \mod \mathcal{T})$, respectively. With this definition, we select $\mathcal{I}_{\mathrm{s}}^*$ the same way as before until the whole horizon is covered by the span of the extended hat intervals of selected intervals. In this way, Lemma 2 remains correct and the analysis follows.

6 The Permutation problem

In this section, we consider a variant of the Skyline problem. A solution of the Skyline problem can be obtained by first partitioning $\mathcal{I}$ into disjoint subsets such that the intervals of every subset are pairwise disjoint, and then assigning distinct colors to the subsets. The second stage is exactly the problem of finding a permutation of the subsets of intervals.

Precisely, we define the problem Permutation as follows. We are given $|\Lambda|$ disjoint sets of intervals $\mathcal{I}_1, \mathcal{I}_2, \cdots, \mathcal{I}_{|\Lambda|}$ such that the intervals in each set are pairwise disjoint, i.e., all the intervals of a set $\mathcal{I}_i$ can be assigned the same color. The goal is to find a permutation π of the colors such that $\mathcal{I}_i$ is assigned the color $\pi(i)$ and the total cost of the coloring induced by π is minimized. At first glance, the permutation problem may look simpler since the partition into sets is already given. Yet we show in this section that the permutation problem is NP-hard. Note that there is no requirement on whether the given partition is an optimal partition or not. Our NP-hardness proof does not tell the complexity of the problem when we are given an optimal partition into colors. The proof is by reduction from the optimal linear arrangement problem.

Optimal linear arrangement (Lina). The input is a graph $G = (V, E)$, and the goal is to find a one-to-one function $f : V \to \{1, 2, \ldots, |V|\}$ such that $\sum_{(u,v)\in E} |f(v) - f(u)|$ is minimized. The decision version of this problem is known to be NP-hard (see [9]).

We denote the degree of a vertex $v \in V$ by $d(v)$. We also denote the maximum degree of all vertices by Δ.

The reduction. Given a simple graph $G = (V, E)$ which is an instance of Lina, we construct an instance $\mathcal{I}$ of Permutation. For each vertex $v \in V$, we create a subset of intervals $\mathcal{I}_v \subseteq \mathcal{I}$ such that the intervals in $\mathcal{I}_v$ are pairwise disjoint. The details of construction are as follows. For each edge $e = uv \in E$, we create an edge gadget containing two identical intervals I_u^e and I_v^e of length 2. The intervals corresponding to distinct edges are disjoint. Then for every vertex $v \in V$, we create a dummy interval of length $\Delta - d(v)$. Each dummy interval is disjoint from any other interval in the construction. Overall, the set $\mathcal{I}_v$ consists

of all intervals I_v^e where v is an endpoint of edge e and its dummy interval. The input $\mathcal{I}$ is then $\{\mathcal{I}_v \mid v \in V\}$. We are going to prove the following theorem.

Theorem 11. *The* Permutation *problem is NP-hard.*

Proof. Consider a solution π of Permutation on instance $\mathcal{I}$. The cost of the two intervals associated with an edge $e = uv$ is

$$2\max\{\pi(u), \pi(v)\} = \pi(u) + \pi(v) + |\pi(u) - \pi(v)|.$$

The cost of a dummy edge associated with a vertex v is

$$(\Delta - d(v))\,\pi(v)$$

Summing up the first cost over all edges and the second over all vertices we get the following expression for the cost of π.

$\textsc{Permutation}(\mathcal{I}, \pi)$

$$\begin{aligned}
&= \sum_{e=uv\in E} (\pi(u) + \pi(v)) + \sum_{e=uv\in E} |\pi(u) - \pi(v)| + \sum_{v\in V} (\Delta - d(v))\pi(v) \\
&= \textsc{Lina}(G, \pi) + \sum_{e=uv\in E} (\pi(u) + \pi(v)) + \sum_{v\in V} (\Delta - d(v))\pi(v) \\
&= \textsc{Lina}(G, \pi) + \sum_{v\in V} \pi(v)d(v) + \sum_{v\in V} (\Delta - d(v))\pi(v) \\
&= \textsc{Lina}(G, \pi) + \sum_{v\in V} \Delta \cdot \pi(v) = \textsc{Lina}(G, \pi) + \Delta \sum_{v\in V} \pi(v) \\
&= \textsc{Lina}(G, \pi) + \frac{\Delta \cdot |V| \cdot (|V| + 1)}{2}.
\end{aligned}$$

Since the second term does not depend on π, minimizing $\textsc{Permutation}(\mathcal{I}, \pi)$ is equivalent to minimizing $\textsc{Lina}(G, \pi)$. □

7 Conclusion

We initiated the study of online algorithms for the coloring problem Skyline. An immediate research direction is to extend the online algorithms for two cases: (i) each color can have an arbitrary capacity; and (ii) the cost of a color class is given by an arbitrary increasing function, i.e., for arbitrary λ. Another direction is to find a better competitive ratio for bounded length intervals. The other directions include determining if there is PTAS for the problem or whether the problem is APX-hard. For the variant Permutation, it is desirable to obtain a stronger complexity result by determining whether the problem of permuting the color classes stays NP-hard if the given color classes correspond to an overall optimal solution. One can also study offline approximation algorithms and online algorithms, and other objective functions.

References

1. U. Adamy and T. Erlebach. Online coloring of intervals with bandwidth. In *WAOA*, pages 1–12, 2003.
2. M. Alicherry and R. Bhatia. Line system design and a generalized coloring problem. In *ESA*, pages 19–30, 2003.
3. Y. Azar, A. Fiat, M. Levy, and N. S. Narayanaswamy. An improved algorithm for online coloring of intervals with bandwidth. *TCS*, 363(1):18–27, 2006.
4. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.
5. J. Chang, H. N. Gabow, and S. Khuller. A model for minimizing active processor time. *Algorithmica*, 70(3):368–405, 2014.
6. J. Chang, S. Khuller, and K. Mukherjee. LP rounding and combinatorial algorithms for minimizing active and busy time. In *SPAA*, pages 118–127, 2014.
7. M. Chrobak and M. Slusarek. On some packing problem related to dynamic storage allocation. *ITA*, 22(4):487–499, 1988.
8. M. Flammini, G. Monaco, L. Moscardelli, H. Shachnai, M. Shalom, T. Tamir, and S. Zaks. Minimizing total busy time in parallel scheduling with application to optical networks. *Theor. Comput. Sci.*, 411(40-42):3553–3562, 2010.
9. M. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
10. M. Garey, D. S. Johnson, G. Miller, and C. H. Papadimitriou. The complexity of coloring circular arcs and chords. *SIAM J. Algebraic Discrete Methods*, 1(2):216–227, 1980.
11. M. M. Halldórsson, G. Kortsarz, and H. Shachnai. Minimizing average completion of dedicated tasks and interval graphs. In *APPROX*, pages 114–126, 2001.
12. K. Jansen. Approximation results for the optimum cost chromatic partition problem. *J. Algorithms*, 34(1):54–89, 2000.
13. R. Khandekar, B. Schieber, H. Shachnai, and T. Tamir. Minimizing busy time in multiple machine real-time scheduling. In *FSTTCS*, pages 169–180, 2010.
14. H. Kierstead and W. Trotter. An extremal problem in recursive combinatorics. *Congressus Numerantium*, 33:143–153, 1981.
15. L. G. Kroon, A. Sen, H. Deng, and A. Roy. The optimal cost chromatic partition problem for trees and interval graphs. In *WG*, pages 279–292, 1996.
16. M. Kubale, editor. *Graph Colorings*. American Mathematical Society, 2004.
17. V. Kumar and A. Rudra. Approximation algorithms for wavelength assignment. In *FSTTCS*, pages 152–163, 2005.
18. G. Mertzios, M. Shalom, A. Voloshin, P. Wong, and S. Zaks. Optimizing busy time on parallel machines. *TCS*, 562:524–541, 2015.
19. S. Nicoloso, M. Sarrafzadeh, and X. Song. On the sum coloring problem on interval graphs. *Algorithmica*, 23(2):109–126, 1999.
20. S. V. Pemmaraju, R. Raman, and K. R. Varadarajan. Buffer minimization using max-coloring. In *SODA*, pages 562–571, 2004.
21. R. Ren and X. Tang. Clairvoyant dynamic bin packing for job scheduling with minimum server usage time. In *SPAA*, pages 227–237, 2016.
22. M. Shalom, A. Voloshin, P. Wong, F. Yung, and S. Zaks. Online optimization of busy time on parallel machines. *TCS*, 560:190–206, 2014.
23. P. Winkler and L. Zhang. Wavelength assignment and generalized interval graph coloring. In *SODA*, pages 830–831, 2003.