Balanced Interval Coloring

Antonios Antoniadis, Falk Hüffner, Pascal Lenzner, Carsten Moldenhauer, and Alexander Souza

Institut für Informatik, Humboldt-Universität zu Berlin, D-10099 Berlin, Germany
{antoniad,hueffner,lenzner,moldenha,souza}@informatik.hu-berlin.de

Abstract

We consider the discrepancy problem of coloring n intervals with k colors such that at each point on the line, the maximal difference between the number of intervals of any two colors is minimal. Somewhat surprisingly, a coloring with maximal difference at most one always exists. Furthermore, we give an algorithm with running time $O(n \log n + kn \log k)$ for its construction. This is in particular interesting because many known results for discrepancy problems are non-constructive. This problem naturally models a load balancing scenario, where n tasks with given start- and endtimes have to be distributed among k servers. Our results imply that this can be done ideally balanced.

When generalizing to d-dimensional boxes (instead of intervals), a solution with difference at most one is not always possible. We show that for any $d \geq 2$ and any $k \geq 2$ it is NP-complete to decide if such a solution exists, which implies also NP-hardness of the respective minimization problem.

In an online scenario, where intervals arrive over time and the color has to be decided upon arrival, the maximal difference in the size of color classes can become arbitrarily high for any online algorithm.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems—Sequencing and scheduling

Keywords and phrases Load balancing, discrepancy theory, NP-hardness

Digital Object Identifier 10.4230/LIPIcs.STACS.2011.531

1 Introduction

In this paper, we consider the following load balancing problem: We are given a set $\mathcal{I} = \{I_1, \ldots, I_n\}$ of tasks, where each task is represented by an interval $I = [\ell, r] \in \mathcal{I}$ with starttime ℓ and endtime r. Furthermore, we are given k servers and have to assign the tasks to the servers as evenly as possible. That is, we want to minimize the maximal difference of the numbers of tasks processed by any two servers over all times.

We formalize this in terms of an interval coloring problem: We are given a set $\mathcal{I} = \{I_1, \ldots, I_n\}$ of n intervals on the real line and a set $K = \{1, \ldots, k\}$ of k colors. A *k-coloring* is a mapping $\chi : \mathcal{I} \to K$. For a fixed k-coloring χ and a point $x \in \mathbb{R}$, let $c_i(x)$ denote the number of intervals containing x that have color i in χ. Define the *imbalance* of χ at x by $\operatorname{imb}(x) = \max_{i,j \in K} |c_i(x) - c_j(x)|$. In words, this is the maximum difference in the size of color classes at point x. The *imbalance* of χ is given by $\operatorname{imb}(\chi) = \max_{x \in \mathbb{R}} \operatorname{imb}(x)$.

These definitions yield the following minimization problem:

MINIMUM IMBALANCE INTERVAL k-COLORING
Instance: A set of intervals $\mathcal{I}$.
Task: Find a k-coloring χ with minimal $\operatorname{imb}(\chi)$.

We call a k-coloring with imbalance at most one *balanced*. Observe that if the number of intervals intersecting at some point is not divisible by k, then imbalance at least one is unavoidable. On the other hand, if the number of intersecting intervals is divisible by k, then no coloring having imbalance one exists. Thus, if a balanced coloring exists, its imbalance is minimal.

28th Symposium on Theoretical Aspects of Computer Science (STACS'11).
Editors: Thomas Schwentick, Christoph Dürr; pp. 531–542
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
SYMPOSIUM ON THEORETICAL ASPECTS OF COMPUTER SCIENCE

As we will see shortly, it is always possible to find a balanced interval k-coloring. Hence, we will mostly be concerned with its construction. More specifically, the questions considered in this paper are outlined as follows:

(i) Is there always a balanced k-coloring?
(ii) If so, is it possible to construct a balanced k-coloring in polynomial time?
(iii) If we consider arcs of a circle (instead of intervals), do balanced k-colorings always exist?
(iv) How is the situation if intervals arrive online?
(v) If d-dimensional boxes (instead of intervals) are considered, can the existence of a balanced k-coloring be decided in polynomial time?

The problem has close connections to discrepancy theory; see Doerr [9] and Matoušek [17] for introductions to the field. Let $H = (X, U)$ be a hypergraph consisting of a set X of vertices and a set $U \subseteq 2^X$ of hyperedges. Analogous to the previous definitions, a k-coloring is a mapping $\chi : X \to K$, and the imbalance $\mathrm{imb}(\chi)$ is the largest difference in size between two color classes over all hyperedges. The *discrepancy* problem is to determine the smallest possible imbalance, i. e., $\mathrm{disc}(H) = \min_{\chi:X\to K} \mathrm{imb}(\chi)$.

Hence our problem is to find the discrepancy of the hypergraph $H = (I, U)$, where U is the family of all maximal subsets of intervals intersecting at some point. It turns out that this hypergraph has totally unimodular incidence matrix, which is useful because de Werra [22] proved that balanced k-colorings exist for hypergraphs with totally unimodular incidence matrix. However, the proof in [22] is only partially constructive: A balanced k-coloring is constructed by iteratively solving the problem of balanced 2-coloring on hypergraphs with totally unimodular incidence matrix, for which no algorithm was given in [22].

Further related work in discrepancy theory mostly considers hypergraph coloring with two colors and often from existential, rather than algorithmic perspective. For an arbitrary hypergraph H with n vertices and m hyperedges, the bound $\mathrm{disc}(H) \leq \sqrt{2n \ln(2m)}$ for 2-coloring follows with the probabilistic method; see also [9]. For $m \geq n$, Spencer [20] proved the stronger result $\mathrm{disc}(H) = O(\sqrt{n \log(m/n)})$, which is in particular interesting for $m = O(n)$. If each vertex is contained in at most t edges, the 2-coloring bound $\mathrm{disc}(H) = O(\sqrt{t} \log n)$ was shown by Srinivasan [21] and the bound $\mathrm{disc}(H) \leq 2t - 1$ by Beck and Fiala [5]. Biedl et al. [6] improved the bound to $\mathrm{disc}(H) \leq \max\{2t - 3, 2\}$ for 2-colorings and established $\mathrm{disc}(H) \leq 4t - 3$ for general k-colorings. They also showed that it is NP-complete to decide the existence of balanced k-colorings for hypergraphs with $t \geq \max\{3, k - 1\}$ and $k \geq 2$.

Bansal [4] recently gave efficient algorithms that achieve 2-color imbalances similar to [20, 21] up to constant factors. In particular, an algorithm yields $\mathrm{disc}(H) = O(\sqrt{n} \log(2m/n))$ matching the result of Spencer [20] if $m = O(n)$. Furthermore, $\mathrm{disc}(H) = O(\sqrt{t} \log n)$ complies with the non-constructive result of Srinivasan [21]. For general $k > 2$, Doerr and Srivastav [10] gave a recursive method constructing k-colorings from (approximative) 2-colorings.

Unfortunately, these results on general discrepancy theory do not answer any of the problems considered here, because t is only bounded by the number of vertices.

Our Contributions. We contribute the following answers to the above questions:

(i) Balanced k-colorings exist for any set $\mathcal{I}$ of intervals, i. e., question (i) can *always* be answered in the affirmative. We establish this by showing that our hypergraph H has totally unimodular incidence matrix and then applying a result of de Werra [22]. This also follows independently from our algorithmic results below.

(ii) We present an $O(n \log n)$ time algorithm for finding a balanced 2-coloring, thereby establishing the first constructive result for intervals. Furthermore, we give an $O(n \log n + kn \log k)$ algorithm for finding a balanced k-coloring. This is an improvement in time complexity, since the construction of de Werra [22] combined with our algorithm for 2-coloring only yields $O(n \log n + k^2 n)$. We also note that our algorithm works for any hypergraph with incidence matrix having the consecutive-ones property.

(iii) If we consider arcs of a circle instead of intervals, balanced k-colorings do not exist in general. However, we give an algorithm achieving imbalance at most two with the same time complexity as in the interval case.

(iv) In an online scenario, in which we learn intervals over time, the imbalance of *any* online algorithm can be made arbitrarily high.

(v) For d-dimensional boxes, it is NP-complete to decide if a balanced k-coloring exists for any $d \geq 2$ and any $k \geq 2$. Our reduction is from NOT-ALL-EQUAL 3SAT. This result clearly implies NP-hardness of the respective minimization problem.

2 Interval Colorings

In this section, we consider MINIMUM IMBALANCE INTERVAL k-COLORING, establish the existence of balanced k-colorings, and give algorithms for 2 and k colors, respectively. Later, we consider arcs of a circle and an online version.

2.1 Existence of Balanced k-Colorings

We begin by observing the existence of balanced k-colorings. In the proof below, we use a theorem of de Werra [22], but the existence of balanced k-colorings also follows from our algorithmic results.

▶ **Theorem 1.** *For any set $\mathcal{I}$ of intervals and any $k \in \mathbb{N}$, there is a balanced k-coloring.*

Proof. Let $\mathcal{I}$ be the set of given intervals. Define a hypergraph $H = (I, U)$, where U is the family of all maximal subsets of intervals intersecting at some point. For H with $I = \{I_1, \ldots, I_n\}$ and $U = \{U_1, \ldots, U_m\}$, the incidence matrix is defined by $A = (a_{i,j})$ with $a_{i,j} = 1$ if $I_i \in U_j$ and $a_{i,j} = 0$ otherwise.

De Werra [22] showed that any hypergraph with totally unimodular incidence matrix admits a balanced k-coloring. It is well-known [19] that a 0–1-matrix is totally unimodular if it has the consecutive-ones property, i. e., if there is a permutation of its columns such that all 1-entries appear consecutively in every row. The incidence matrix A of H has this property: If we order the U_j in increasing order of intersection points, then the entries $a_{i,j} = 1$ appear consecutively in each row. ◀

2.2 Algorithm for Two Colors

In this section, we present an algorithm that constructs a balanced 2-coloring in polynomial time. Since the algorithm produces a valid solution for every possible instance, Theorem 1 for $k = 2$ also follows from this algorithmic result. We note in passing that a polynomial-time algorithm can also be obtained by solving a simple Integer Linear Program (ILP) with a totally unimodular constraint matrix. However, this gives a much worse running time bound of $O(n^5 / \log n)$ [3].

The main idea of our algorithm is to simplify the structure of the instance such that the remaining intervals have start- and endpoints occurring pairwise. We then build a constraint graph that has the intervals as vertices. Finally, a proper 2-coloring of the constraint graph induces a solution to the problem.

The start- and endpoints of the intervals are called *events*. A *region* is an interval spanned by two consecutive events and is called even (odd) if it is contained in an even (odd) number of input intervals.

▶ **Theorem 2.** *For any set $\mathcal{I}$ of n intervals, there is a balanced 2-coloring that can be constructed in $O(n \log n)$ time.*

Proof. W. l. o. g. we can assume that the start- and endpoints of the input intervals are pairwise disjoint. If not, a new instance can be obtained by repeatedly increasing one of the coinciding start- or endpoints by $\epsilon/2$, where ϵ is the minimum size of a region. Since the new instance includes a corresponding region for every region of the original instance, a balanced coloring for the new instance is a balanced coloring for the old instance (the converse is not true).

Observe that a coloring that is balanced on all even regions is also balanced on all odd regions. This is because odd regions only differ by one interval from a neighboring even region. Thus, the task reduces to constructing a balanced coloring of the even regions. Since between two consecutive even regions, exactly two events occur, it suffices to consider only pairs of consecutive events enclosing odd regions.

If a pair of events consists of the start- and endpoint of the same interval, this interval is assigned any color and is removed from the instance. If a pair consists of start- and endpoint of different intervals, these intervals are removed from the instance and substituted by a new (minimal) interval that covers their union. In a final step of the algorithm, both intervals will be assigned the color of their substitution. The remaining instance consists solely of pairs of events where two intervals start or two intervals end. Clearly, a balanced coloring has to assign opposite colors to the corresponding two intervals of such a pair, and any such assignment yields a balanced coloring.

The remaining pairs of events induce a *constraint graph*. Every vertex corresponds to an interval, and an edge is added between two vertices if there is a pair of events containing both startpoints or both endpoints. Finding a proper vertex two-coloring of this graph gives a balanced 2-coloring. The constraint graph is bipartite: Each edge can be labeled by "$\vdash$" or "$\dashv$" if it corresponds to two start- or endpoints, respectively. Since each interval is incident to exactly two edges, any path must traverse $\vdash$- and $\dashv$-edges alternatingly. Therefore, every cycle must be of even length and hence the graph is bipartite. Thus, a proper vertex two-coloring of the constraint graph can be found in linear time by depth-first search.

Sorting events takes $O(n \log n)$ time. Creation of the constraint graph and coloring it takes linear time. ◀

Note that if intervals are given already sorted, or interval endpoints are described by small integers, then the above algorithm can even find a balanced 2-coloring in linear time.

2.3 Algorithms for k Colors

In this section, we extend the results of the previous section to an arbitrary number of colors k and show that a balanced interval k-coloring can be found in polynomial time.

A first polynomial time algorithm can be obtained using a construction by de Werra [22]: Start with an arbitrary coloring and find two colors i and j for which $\max_x |c_i(x) - c_j(x)|$ is maximal. Use the algorithm from Section 2.2 to find a balanced 2-coloring of all intervals that currently have color i or j and recolor them accordingly. Repeat until the coloring is balanced. This algorithm has running time $O(n \log n + k^2 n)$, because sorting intervals is needed only once for the above algorithm for 2 colors, and there are at most $\binom{k}{2}$ recolorings necessary.

In the following, we present an alternative algorithm for k colors, which is faster than $O(n \log n + k^2 n)$. We will first give an overview of the argument, and then a more formal description.

As in Section 2.2, we assume w. l. o. g. that all start- and endpoints are pairwise disjoint. The idea is to scan the events in order, beginning with the smallest, and to capture dependencies in k-tuples of intervals that indicate pairwise different colors. That is, we reduce the MINIMUM IMBALANCE INTERVAL k-COLORING instance to an instance of STRONG HYPERGRAPH COLORING, formally defined as follows.

STRONG HYPERGRAPH COLORING
Instance: A ground set X, a family $\mathcal{S}$ of constraints $S_1, \ldots, S_n \subseteq X$, and an integer k.
Task: Find a k-coloring $\chi : X \to K$ with $\forall\, 1 \leq i \leq n : x, y \in S_i, x \neq y \Rightarrow \chi(x) \neq \chi(y)$.

For example, in the special case that each block of k consecutive events consists only of start- or only of endpoints, the constraints that the corresponding k intervals have to be differently colored will capture the whole solution. As we will see below, also different interval nesting structures can be captured by such constraints.

STRONG HYPERGRAPH COLORING is NP-hard in general [1]. However, each interval will occur in at most two constraints, corresponding to its start- and endpoint. Thus, we can further reduce the STRONG HYPERGRAPH COLORING instance to EDGE COLORING, where the goal is to color edges of a multigraph such that the edges incident to each vertex are all differently colored. More formally:

EDGE COLORING
Instance: A multigraph $G = (V, E)$ and an integer k.
Task: Find a k-coloring $\chi : E \to K$ with $\forall e, e' \in E, e \cap e' \neq \emptyset, e \neq e' \Rightarrow \chi(e) \neq \chi(e')$.

It is easy to see that any instance $(X, \mathcal{S}, k)$ of STRONG HYPERGRAPH COLORING where each element of X occurs in at most two constraints can be reduced to EDGE COLORING as $(G = (\mathcal{S}, E), k)$, where for each $x \in X$ that occurs in S_i and S_j with $i \neq j$, we add the edge $\{S_i, S_j\}$ to E. For our case, a constraint corresponds to a vertex, and an interval corresponds to an edge that connects the two constraints it occurs in. An edge coloring with k colors will thus provide a balanced interval k-coloring.

Clearly, an edge coloring of a multigraph with maximum degree Δ needs at least Δ colors. Finding an edge coloring of minimum size is NP-hard in general [14]. However, Kőnig [16] showed that for bipartite multigraphs, Δ colors in fact always suffice. Further, an edge coloring of a bipartite multigraph with m edges can be found in $O(m \log \Delta)$ time [8]. The multigraph we will construct has maximum degree k and is bipartite. Thus, a balanced interval k-coloring always exists and can be found in polynomial time.

We now describe the construction of the constraints and give a more rigorous description of the results. From a MINIMUM IMBALANCE INTERVAL k-COLORING instance $\mathcal{I}$, we construct a STRONG HYPERGRAPH COLORING instance $(\mathcal{I}, \mathcal{S}, k)$ over the ground set of the intervals. The algorithm scans the set of events in order, beginning with the smallest. It keeps a set of active events and adds constraints to $\mathcal{S}$. The set of active events will be cleared at each region where the number of intervals is 0 modulo k. Thus, the active events always describe the change from a situation where each color occurs the same number of times.

The construction of the constraints can be visualized with a decision tree, depicted for the example $k = 4$ in Figure 1. At the beginning, the set of active events is empty (which corresponds to the root of the decision tree). Whenever the set of events is empty, the algorithm branches into two cases depending on the type of the next event. Both branches are equivalent, with the roles of start- and endpoints interchanged. Therefore, assume the next event is the start of an interval (depicted as $\overset{a}{\vdash}$ on the left branch). It is added to the active set. This continues until either k startpoints of intervals $I_1, \ldots, I_k$ are added, or an endpoint is encountered. In the first case, the constraint

$$(I_1, \ldots, I_k) \tag{1}$$

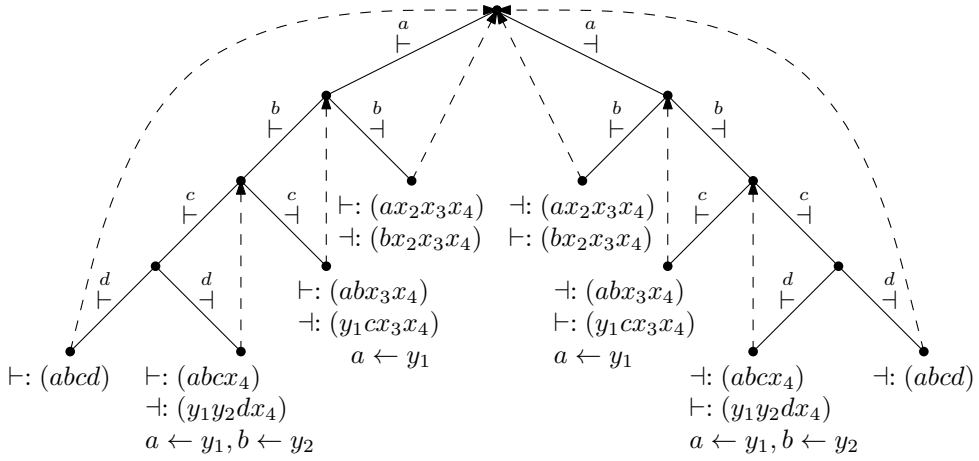


Figure 1 Tracking of active events for $k = 4$

is constructed and the set of active events is cleared (dashed arrow returning to the root). In the second case, assume the startpoints of intervals $I_1, \ldots, I_j$ have been added and the endpoint of interval I_{j+1} is encountered. Then, the two constraints

$$(I_1, \ldots, I_j, x_{j+1}, \ldots, x_k) \tag{2}$$

$$(y_1, \ldots, y_{j-1}, I_{j+1}, x_{j+1}, \ldots, x_k) \tag{3}$$

are constructed. Here, $x_{j+1}, \ldots, x_k$ and $y_1, \ldots, y_{j-1}$ are new "virtual" intervals that have not been used in previous constraints. That is, they do not correspond to actual intervals of the real line and only serve as placeholders in the active set. Furthermore, the startpoints of the intervals $I_1, \ldots, I_j$ are replaced by the startpoints of $y_1, \ldots, y_{j-1}$ in the active set of events (indicated by the dashed arrows pointing one level higher in the decision tree).

We now prove the correctness of the two chained reductions.

▶ **Lemma 3.** *A solution to the* Strong Hypergraph Coloring *instance $(\mathcal{I}, \mathcal{S}, k)$, constructed as described above, yields a balanced k-coloring for $\mathcal{I}$.*

Proof. Recall that a region is an interval spanned by two consecutive events. The proof is by induction over all regions, in the order of events. At each region, we count the number of times each of the k colors is used among the intervals containing the region. We will show that these counters differ by at most one, i. e., the coloring is balanced. Clearly, all counters are equal to zero before the first event.

In particular, we show that in regions where the number of intervals is 0 modulo k, all counters are equal, and that between these regions the counters change by at most one and all in the same direction. We distinguish the same cases as in the construction, limiting the discussion to the case that the event first added to the empty set of active events is a startpoint.

If k startpoints of intervals $I_1, \ldots, I_k$ are encountered, there is a constraint of the form (1), which ensures that all of them have different colors. Therefore, at each startpoint, a different counter increases by one. Hence, the counters differ by at most one in all regions up to the startpoint of I_k, and are all equal in the region beginning with the startpoint of I_k.

Consider that only $j < k$ startpoints of the intervals $I_1, \ldots, I_j$ are encountered. Since these startpoints were added to the set of active events during the construction, the intervals $I_1, \ldots, I_j$ do all occur in one constraint. This constraint forces them to have different colors, and therefore the colors of $I_1, \ldots, I_j$ are exactly the colors with increased count. Now, we distinguish two subcases

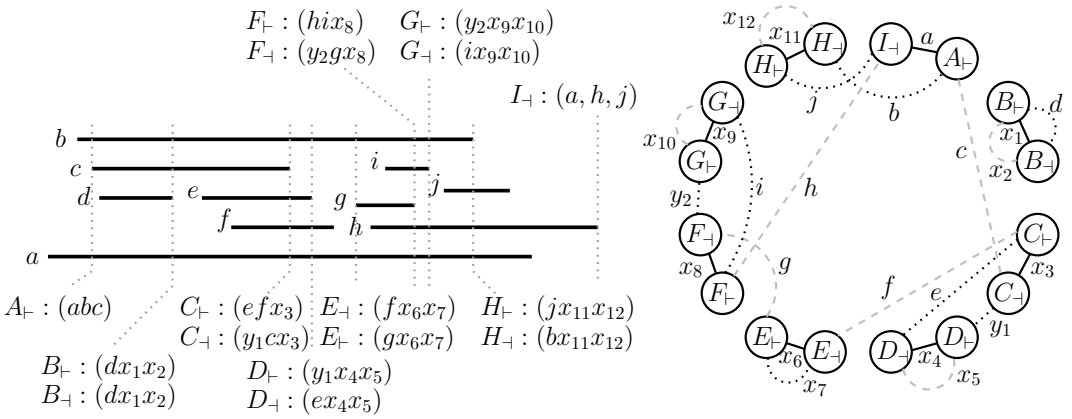


Figure 2 Complete example of the constraints and bipartite EDGE COLORING instance for $k = 3$ with a valid coloring.

depending on the next event. If the next event is a startpoint of some interval, denoted by I_{j+1}, it will also be part of the same constraint. Hence, I_{j+1} has a different color whose count is not yet increased. The second subcase is that the next event is the endpoint of some interval, denoted by I_{j+1}. The interval I_{j+1} is forced to have the same color as one of the intervals $I_1, \ldots, I_j$. This is because there is a constraint of the form (2) that collects all other colors in variables $x_{j+1}, \ldots, x_k$, which occur together with I_{j+1} in a constraint of the form (3). Thus, the previously increased counter for the color of I_{j+1} decreases again. Because of the constraint of the form (2), the virtual intervals $y_1, \ldots, y_{j-1}$ must have all of the colors of $I_1, \ldots, I_j$ except for the color of I_{j+1}. Hence, the colors of $y_1, \ldots, y_{j-1}$ are exactly all remaining colors with increased count. Therefore, we are in the same situation as before encountering the endpoint of I_{j+1}. By repeating the above argument, the claim follows in this case.

If the event first added is an endpoint, the argument is symmetrical. ◀

▶ **Lemma 4.** *A* STRONG HYPERGRAPH COLORING *instance* $(\mathcal{I}, \mathcal{S}, k)$ *constructed as described above can be reduced to a bipartite* EDGE COLORING *instance.*

Proof. We need to show that each interval occurs in at most two constraints. Once this is proved, it is possible to build a multigraph with the constraints as vertices and edges between them if they share a common interval. Further, it has to be shown that this multigraph is bipartite. To show both parts at once, we color the constraints in $\mathcal{S}$ with the two colors $\vdash$ and $\dashv$. It then suffices to show that every interval can occur in at most one $\vdash$-constraint and in at most one $\dashv$-constraint.

Consider Figure 2 for an illustration of the constructed constraints and the respective bipartite EDGE COLORING instance.

We color a constraint with $\vdash$ if all involved events belonging to nonvirtual intervals are startpoints, and with $\dashv$ if all these events are endpoints (see Figure 1). All nonvirtual intervals therefore occur in exactly two constraints, constructed when the start- and endpoint get removed from the active set of events. A virtual x-interval always occurs in a pair of subsequent differently colored constraints, and is not used anywhere else. For the left branch of the decision tree, a virtual y-interval occurs first in a $\dashv$-constraint, and its startpoint is then added to the list of active events. Then, it will be used in a constraint of type either (1) or (2), both of which are of type $\vdash$. The argument is symmetrical for the right branch of the decision tree. ◀

▶ **Theorem 5.** *Every set of* n *intervals* $\mathcal{I}$ *has a balanced* k*-coloring for any* $k \in \mathbb{N}$*, and it can be found in* $O(n \log n + kn \log k)$ *time.*

Proof. By Lemmas 3 and 4, MINIMUM IMBALANCE INTERVAL k-COLORING can be reduced to EDGE COLORING with fixed k in a bipartite multigraph. The maximum degree of this multigraph is k, since by construction no constraint has more than k elements. By Kőnig's theorem [16] existence follows.

To be able to process the events, they have to be sorted in $O(n \log n)$ time. We have $O(kn)$ virtual intervals and thus $O(kn)$ edges in the EDGE COLORING instance. Finding an edge k-coloring for this multigraph with maximum degree k can be done in $O(kn \log k)$ time [8]. ◀

Note that the EDGE COLORING algorithm by Cole, Ost, and Schirra [8] uses quite involved data structures. In practice, it might be preferable to use the much simpler algorithm by Alon [2] running in $O(m \log m)$ time for an m-edge graph, which gives a worst-case bound of $O(kn \log n)$. An implementation of our algorithm in Python using a simple edge coloring algorithm based on augmenting paths can be found at http://www2.informatik.hu-berlin.de/~hueffner/intcol.py.

For an extension, recall that a matrix has the consecutive-ones property if there is a permutation of its columns such that all 1-entries appear consecutively in every row. Such a permutation can be found in linear time by the PQ-algorithm [7]. Given such a matrix, it is straightforward to construct an instance of MINIMUM IMBALANCE INTERVAL k-COLORING.

▶ **Theorem 6.** *For any hypergraph H with an $n \times m$ incidence matrix having the consecutive ones property, a balanced k-coloring can be found in $O(nm + kn \log k)$ time.*

2.4 Arcs of a Circle

In a periodic setting, the tasks $\mathcal{I}$ might be better described by a set of arcs of a circle rather than a set of intervals. In this case, there are instances that require an imbalance of two (e. g., three arcs that intersect exactly pairwise and $k = 2$). We show that two is also an upper bound and a coloring with maximal imbalance two can be found in polynomial time.

▶ **Theorem 7.** *The maximal imbalance for arcs of a circle is two, and finding a coloring with imbalance at most two can be done in $O(n \log n + kn \log k)$ time.*

Proof. Define a point on the circle, called zero, and consider counterclockwise orientation. We build an instance of MINIMUM IMBALANCE INTERVAL k-COLORING by "unfolding" the circle at zero in the following way. Consider only arcs that do not span the full circle. Map all such arcs not containing zero to intervals of same length at the same distance right of zero on the real line. Map the arcs containing zero to intervals of same length such that the positive part of the interval has the same length as the part of the arc in counterclockwise direction from zero. Finally, map the arcs containing the full circle to intervals spanning all of the instance constructed so far. Use the above algorithm to obtain a coloring of the intervals with imbalance at most one at every point. By reversing the mapping, the obtained coloring of the arcs has imbalance at most two (each point on the circle is mapped to at most two points of the real line). ◀

2.5 Online Algorithms

In load balancing problems, it is often more realistic to assume an online scenario, where not all information is known in advance, but is rather arriving piece-by-piece, and irrevocable decisions have to be made immediately. In our setting, this means that intervals arrive in order of their startpoint, including the information of their endpoint, and a color has to be assigned to them immediately.

The problem of finding a proper coloring (i. e., a coloring where no two intersecting intervals have the same color) of intervals in an online setting has found considerable interest [15, 11]. In these works, the objective is to use a minimum number of colors. In contrast, we consider a fixed number

of colors and the minimization of the imbalance. We show that in contrast to the offline scenario, here the imbalance can become arbitrarily large.

▶ **Theorem 8.** *In online* MINIMUM IMBALANCE INTERVAL k-COLORING*, the imbalance is unbounded.*

Proof. We first consider the case $k = 2$ with colors "$+1$" and "-1". Denote the signed imbalance $\mathrm{simb}(x)$ to be the sum of the colors of the intervals containing x. Note that $\mathrm{imb}(x) = |\mathrm{simb}(x)|$.

In the following, we outline how a sequence of intervals can be constructed such that no online algorithm can yield a bounded imbalance. Initially, $\mathrm{simb} \equiv 0$. Set $L = [0, 1]$ and $R = [2, 3]$. Let L_ℓ, R_ℓ, L_r, and R_r denote the start- and endpoints of the current L and R, respectively. Repeat the following steps.

- Present the interval $[(L_\ell + L_r)/2, (R_\ell + R_r)/2]$ to the online algorithm.
- If it chooses color $+1$, set $R \leftarrow [R_\ell, (R_\ell + R_r)/2]$, else $R \leftarrow [(R_\ell + R_r)/2, R_r]$.
- Set $L \leftarrow [(L_\ell + L_r)/2, L_r]$.

This sequence of intervals is legal, since the startpoints increase strictly monotonously. In each repetition, if the algorithm chooses color $+1$, the signed imbalances in L and R increase by one. If the algorithm chooses -1, the signed imbalance decreases by one in L and remains unchanged in R, i. e., the difference of the signed imbalance in L and R increases. Therefore, the signed imbalance diverges in L or R. Since the imbalance is the absolute value of the signed imbalance, it becomes unbounded.

The construction easily generalizes to $k > 2$ colors. We only track two arbitrary colors, and whenever the algorithm assigns an untracked color to an interval, we present the same interval (with a slightly increased startpoint) again, forcing it to eventually assign a tracked color or to produce unbounded imbalance. ◀

3 Hardness of Generalizations

We consider several generalizations of MINIMUM IMBALANCE INTERVAL k-COLORING and show that they are NP-hard. Note that the hardness results of Biedl et al. [6] do not apply to the problems we consider here.

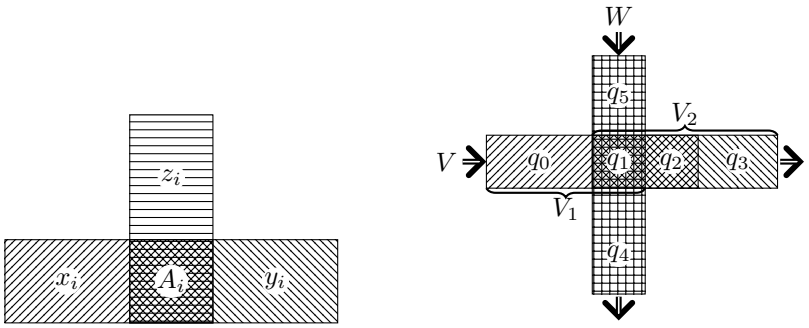
d-Dimensional Boxes. Gyárfás and Lehel [13] suggest to examine d-dimensional boxes as generalizations of intervals for coloring problems. The problem MINIMUM IMBALANCE d-BOX k-COLORING has as input an integer k and a set $\mathcal{I} = \{I_1, \ldots, I_n\}$ of n d-dimensional boxes $I_i = ([\ell_{i,1}, r_{i,1}], [\ell_{i,2}, r_{i,2}], \ldots, [\ell_{i,d}, r_{i,d}])$ for $1 \leq i \leq n$.

For every point $x = (x_1, \ldots, x_d)$, let $S(x)$ be the set of boxes that include x, i. e., $S(x)$ contains all the elements I_i such that $\ell_{i,j} \leq x_j \leq r_{i,j}$ for all $1 \leq j \leq d$. For a coloring $\chi : \mathcal{I} \to K$, a color i, and a point x, let $c_i(x)$ be the number of boxes in $S(x)$ of color i. With the analog definition of imbalance $\mathrm{imb}(\chi)$ and balance for d-dimensional boxes, the problem statement becomes:

MINIMUM IMBALANCE d-BOX k-COLORING
Instance: A set $\mathcal{I}$ of d-dimensional boxes.
Task: Find a k-coloring χ with minimal $\mathrm{imb}(\chi)$.

First note that, unlike for the case $d = 1$, a balanced coloring may not exist: already for three rectangles, some instances require imbalance two. Hence, we also have a related decision problem:

BALANCED d-BOX k-COLORING
Instance: A set $\mathcal{I}$ of d-dimensional boxes.
Question: Is there a balanced k-coloring χ?



(a) Gadget for clause $C_i = (x_i, y_i, z_i)$ (b) Gadget for crossing chains V and W

Figure 3 The gadgets of the reduction.

We show that for all $d \geq 2$ and $k \geq 2$, it is NP-complete to decide BALANCED d-BOX k-COLORING. This clearly implies NP-hardness of MINIMUM IMBALANCE d-BOX k-COLORING.

▶ **Theorem 9.** BALANCED d-BOX k-COLORING *is NP-complete for any* $d \geq 2$ *and any* $k \geq 2$.

We will reduce from NOT-ALL-EQUAL 3SAT (NAE-3SAT) [18]. Note that the classic definition of NAE-3SAT [12] allows negated variables. However, this is not needed to make the problem NP-complete [18]. Thus, in the sequel, we will assume that all variables occur only non-negatedly.

> NOT-ALL-EQUAL 3SAT (NAE-3SAT)
> **Instance:** A Boolean formula with clauses $C_1, \ldots, C_m$, each having at most 3 variables.
> **Question:** Is there a truth assignment such that in every clause, not all variables have the same value?

We first consider $k = 2$ and then generalize to arbitrary k. We present the gadgets of the reduction, then show how they are combined together, and conclude by proving correctness.

For each clause $C_i = (x_i, y_i, z_i)$, we construct a *clause gadget* comprised of three rectangles (see Figure 3a). Note that all three rectangles overlap in region A_i, and only there. Then we also construct a separate rectangle r_j for every variable. Finally, we connect each r_j to all rectangles that appear in a clause gadget, and correspond to the same variable as r_j. We do this by a chain with odd number of rectangles. This ensures that in any balanced 2-coloring, r_j and the corresponding rectangle in the clause gadget have the same color. If two chains need to cross, we introduce a *crossing gadget* as seen in Figure 3b. Three rectangles are relevant for the crossing of two chains V and W. The first is V_1 and contains areas q_0, q_1, and q_2, the second is V_2, containing q_1, q_2, and q_3. Both V_1 and V_2 belong to chain V. The last rectangle contains areas q_1, q_4 and q_5 and belongs to chain W. Note that the crossing does not induce any dependencies on the colorings between chains V and W. See Figure 4 for a construction of an instance for BALANCED 2-BOX 2-COLORING.

Observe that the above construction only requires a number of rectangles polynomial in the size of the NAE-3SAT instance.

▶ **Lemma 10.** BALANCED d-BOX 2-COLORING *is NP-complete for any* $d \geq 2$.

Proof. The problem is in NP, since feasibility of a color assignment can be checked in polynomial time. For NP-hardness, we show that a NAE-3SAT instance is satisfiable if and only if the answer to the corresponding BALANCED 2-BOX 2-COLORING instance is "yes". This also implies NP-completeness for every $d \geq 2$ by taking intervals of length 0 in higher dimensions.

($\Rightarrow$) Assume that there is a satisfying assignment of the NAE-3SAT instance. Then, color the rectangles r_j according to the truth values of their corresponding variables. This coloring can be

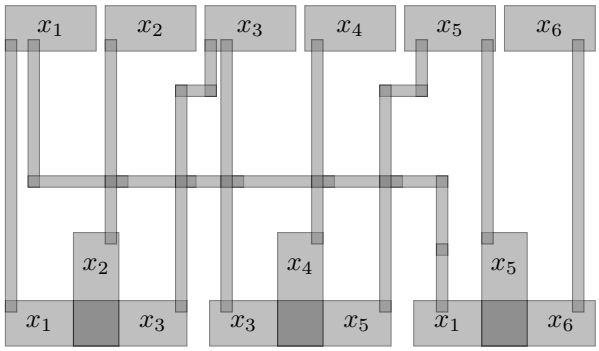


Figure 4 Example for NAE-3SAT instance $(x_1, x_2, x_3), (x_3, x_4, x_5), (x_1, x_5, x_6)$.

easily extended to all the rectangles by alternatively coloring rectangles along a chain (and crossings) starting from each r_j and ending at a clause gadget. It remains to show that $\mathrm{imb}(x) \leq 1$ holds for every point $x \in A_i$ for all $1 \leq i \leq m$. Consider A_i corresponding to clause $C_i = (x_i, y_i, z_i)$. The three rectangles that intersect at A_i have the colors corresponding to the truth values of their variables x_i, y_i, and z_i in the solution of NAE-3SAT. Since the three variables do not have all the same truth value, the three rectangles cannot have all the same color, and $\mathrm{imb}(x) \leq 1$.

($\Leftarrow$) Assume that we have a balanced 2-coloring for the constructed BALANCED 2-BOX 2-COLORING instance. Consider only the clause gadgets. We have already observed that rectangles that correspond to the same variable and appear in clause gadgets must have the same color. We can assign the truth values of the variables according to the colors in the corresponding rectangles. Since in no A_i all three rectangles have the same color, in no C_i all three variables have the same truth value, yielding a feasible solution for NAE-3SAT. ◀

Proof of Theorem 9. First apply the construction for BALANCED 2-BOX 2-COLORING and call its rectangles *reduction rectangles*. Then add $k - 2$ additional rectangles that fully contain the construction and all intersect at least in one point outside the construction; these are called *cover rectangles*. By the latter property, cover rectangles must have distinct colors in any balanced coloring. Observe that each reduction rectangle contains some point that does not intersect with other reduction rectangles but only with all the cover rectangles. This implies that the reduction rectangles have available only the two colors not used by the cover rectangles. We conclude that the problem of k-coloring the constructed instance is equivalent to the problem of 2-coloring only the reduction rectangles. ◀

Further Generalizations. The weighted version, where intervals have weights and the weighted imbalance is to be minimized, is NP-complete by reduction from PARTITION. Furthermore, the variant with multiple intervals [13] is NP-complete by reduction from NAE-3SAT. Both hardness results generalize to higher dimensions. Proofs are omitted due to space constraints.

4 Open Questions

- We have given a polynomial time algorithm for k-coloring hypergraphs with the consecutive-ones property, i. e., a special case of a totally unimodular incidence matrix. It would be interesting to generalize to arbitrary totally unimodular incidence matrices.
- For arcs of a circle, we have shown how to find a coloring with imbalance at most two in polynomial time, but it is not clear how to find an optimal one.

- It remains open how large the imbalance can become for d-dimensional boxes, and whether we can find polynomial-time approximations for it. We were not able to find an instance requiring an imbalance greater than 2 for the 2-dimensional case.

References

1. G. Agnarsson and M. M. Halldórsson. Strong colorings of hypergraphs. In *Proc. 2nd WAOA*, volume 3351 of *LNCS*, pages 253–266. Springer, 2005.
2. N. Alon. A simple algorithm for edge-coloring bipartite multigraphs. *Information Processing Letters*, 85(6):301–302, 2003.
3. K. M. Anstreicher. Linear programming in $O(\frac{n^3}{\ln n}L)$ operations. *SIAM Journal on Optimization*, 9(4):803–812, 1999.
4. N. Bansal. Constructive algorithms for discrepancy minimization. In *Proc. 51st FOCS*. IEEE Computer Society, 2010. To appear. Also in arXiv:1002.2259v4.
5. J. Beck and T. Fiala. "Integer making" theorems. *Discrete Applied Mathematics*, 3(1):1–8, 1981.
6. T. C. Biedl, E. Čenek, T. M. Chan, E. D. Demaine, M. L. Demaine, R. Fleischer, and M.-W. Wang. Balanced k-colorings. *Discrete Mathematics*, 254(1–3):19–32, 2002.
7. K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976.
8. R. Cole, K. Ost, and S. Schirra. Edge-coloring bipartite multigraphs in $O(E \log D)$ time. *Combinatorica*, 21(1):5–12, 2001.
9. B. Doerr. *Integral Approximation*. Habilitationsschrift, Christian-Albrechts-Universität zu Kiel, 2005.
10. B. Doerr and A. Srivastav. Multicolour discrepancies. *Combinatorics, Probability and Computing*, 12:365–399, 2003.
11. L. Epstein. Online interval coloring. In M.-Y. Kao, editor, *Encyclopedia of Algorithms*, pages 594–598. Springer, 2008.
12. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
13. A. Gyárfás and J. Lehel. Covering and coloring problems for relatives of intervals. *Discrete Mathematics*, 55(2):167–180, 1985.
14. I. Holyer. The NP-completeness of edge-coloring. *SIAM Journal on Computing*, 10(4):718–720, 1981.
15. H. A. Kierstead and W. T. Trotter. An extremal problem in recursive combinatorics. *Congressus Numerantium*, 33:143–153, 1981.
16. D. Kőnig. Gráfok és alkalmazásuk a determinánsok és a halmazok elméletére [in Hungarian: Graphs and their application to determinant theory and set theory]. *Matematikai és Természettudományi Értesítő*, 34:104–119, 1916.
17. J. Matoušek. *Geometric Discrepancy: An Illustrated Guide*, volume 18 of *Algorithms and Combinatorics*. Springer, 1999.
18. T. J. Schaefer. The complexity of satisfiability problems. In *Proc. 10th STOC*, pages 216–226. ACM, 1978.
19. A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1986.
20. J. Spencer. Six standard deviations suffice. *Transactions of the American Mathematical Society*, 289(2):679–706, 1985.
21. A. Srinivasan. Improving the discrepancy bound for sparse matrices: Better approximations for sparse lattice approximation problems. In *Proc. 8th SODA*, pages 692–701. ACM-SIAM, 1997.
22. D. de Werra. Equitable colorations of graphs. *Revue Française d'Informatique et de Recherche opérationnelle*, R-3:3–8, 1971.