SIAM J. DISCRETE MATH.
Vol. 36, No. 2, pp. 1200–1230

ON COVERING SEGMENTS WITH UNIT INTERVALS*

DAN BERGREN†, EDUARD EIBEN‡, ROBERT GANIAN§, AND IYAD KANJ†

Abstract. We study the problem of covering a set of segments on a line with the minimum number of unit-length intervals, where an interval covers a segment if at least one of the two endpoints of the segment falls in the unit interval. We also study several variants of this problem. We show that the restrictions of the aforementioned problems to the set of instances in which all the segments have the same length are NP-hard. This result implies several NP-hardness results in the literature for variants and generalizations of the problems under consideration. We then study the parameterized complexity of the aforementioned problems. We provide tight results for most of them by showing that they are fixed-parameter tractable for the restrictions in which all the segments have the same length, and are W[1]-complete otherwise.

Key words. segment covering, unit intervals, NP-completeness, parameterized complexity

AMS subject classifications. 68Q27, 68Q25

DOI. 10.1137/20M1336412

1. Introduction. Problem definition and motivation.

The problem of covering a set of points on the (real) line with the minimum number of closed unit-length intervals is a classical problem (e.g., see exercise 16.2-5 in [11] and exercise 5 in chapter 4 of [22]). This problem can be solved by a simple greedy algorithm that traverses the points from left to right, and in each step places an interval that starts at the leftmost uncovered point. A generalization of the above problem to the problem of covering a set of segments on the real line with the minimum number of unit intervals, where an interval *covers* a segment if at least one endpoint of the segment is in the interval, has been studied in several works [2, 3, 4]. For clarity, throughout the paper, we distinguish—in the nomenclature—the entities to be covered from those used for covering, by referring to the former as *segments* and the latter as *intervals*. It is easy to see that the greedy algorithm—described above—no longer works for this generalization. In fact, this generalization turns out to be NP-hard, even though a straightforward greedy algorithm works for the restriction of the problem to instances in which the length of each segment is at most 1 (unit). (The greedy algorithm sorts the right endpoints of all segments and considers them from left to right, each time placing an interval that starts at the next uncovered segment endpoint.)

Recently, several variants and generalizations of the above segment covering problem have been considered (as discussed below). Two natural variants arise based on whether the unit intervals can be arbitrarily chosen on the line, or are restricted to a given input set; the former version has been referred to as the *continuous* version as opposed to the latter *discrete* version. (Note that the two greedy algorithms mentioned in the previous paragraph, for the point covering and the segment covering

*Received by the editors May 6, 2020; accepted for publication (in revised form) November 28, 2021; published electronically May 23, 2022.
https://doi.org/10.1137/20M1336412
Funding: The third author acknowledges support by the Austrian Science Fund (FWF), projects P31336 and Y1329.
†School of Computing, DePaul University, Chicago, IL 60604-2301 USA (bergren2@gmail.com, ikanj@cs.depaul.edu).
‡Department of Computer Science, Royal Holloway, University of London, Egham, TW20 0EX, UK (eduard.eiben@rhul.ac.uk).
§Algorithms and Complexity Group, TU Wien, Vienna, Austria (rganian@gmail.com).

problems, can be modified to work for the discrete versions of these two problems. The modified greedy algorithm again sorts the right endpoints of all segments and considers them from left to right, each time choosing an interval that covers the next uncovered segment endpoint and extends the most to the right.) Moreover, a more restricted notion of covering has been studied as well, that we refer to henceforth as *exact covering*, in which exactly one endpoint from each segment must be covered by the unit intervals.

In this paper, we study the classical and parameterized complexity of the variants of segment covering by unit intervals discussed above, in most cases providing tight characterizations. The problem variants we study are: Continuous Segment Covering (Cont-SC); Discrete Segment Covering (Disc-SC); Continuous Exact Segment Covering (Cont-Exact-SC); and Discrete Exact Segment Covering (Disc-Exact-SC).

Related work. Arkin et al. [2, 4] studied the exact covering problem of a set of *color classes*, where each color class contains two points on the real line of the same color, with the minimum number of intervals; here an interval covers a color class if it covers exactly one point from the color class. This is precisely the notion of exact segment covering, in which each color class corresponds to a segment whose endpoints are the two points in the class. It was shown in [2] that the aforementioned problem is NP-hard, and that the case in which the intervals are restricted to be unit intervals is NP-hard as well. (Note that, w.r.t. the notion of exact covering, none of these two problems subsumes the other.)

Arkin et al. [3, 4] also studied the problem of finding a *conflict-free* covering, where a color class can have both points covered, but it is not allowed to use an interval that covers both points of any color class. They showed that both the discrete and continuous versions of the aforementioned problem are NP-hard, and gave approximation algorithms of ratios 3 and 2, respectively, for them. They also studied a problem variant in which each color class consists of a horizontal or a vertical unit-length segment in the plane, and the goal is to compute a minimum-cardinality set of axes-parallel unit squares such that exactly one point from each segment is covered by the unit squares. They showed that this variant is NP-hard, and gave an approximation algorithm of ratio 6 for it. Acharyya et al. [1] studied several variants of covering segments with axes-parallel unit squares in the plane. They obtained approximation algorithms and showed the NP-hardness of the variant in which all segments are horizontal unit segments.

The Cont-Exact-SC and Disc-Exact-SC problems under consideration are also related to an NP-hard combinatorial problem, referred to as the "Paintshop" problem [5, 18], that has applications in automotive industry. In this setting, a sequence of cars that are matched up in pairs is given, and the goal is to color the cars with two colors so that no two matched cars receive the same color, and such that the number of color changes (defined to be the number of adjacent car pairs of different colors) in the sequence is minimized. Other applications of covering line segments (referred to as "stabbing") with geometric objects (such as unit disks/squares) are in the area of networks security (see [1, 23]). Finally, we mention that there is a vast amount of literature on other notions of covering and stabbing of geometric objects [10, 15, 16, 24, 25, 26].

Our results. Our results for the Cont-SC, Disc-SC, Cont-Exact-SC, and Disc-Exact-SC problems can be summarized as follows (recall that the covering elements in all these segment covering variants are unit intervals):

(i) The restrictions of Cont-SC, Disc-SC, Cont-Exact-SC, and Disc-Exact-SC to instances in which all segments have the same length are NP-hard.

This NP-hardness result has important implications. First, it strengthens and implies several NP-hardness results in the literature about segment covering. The NP-hardness of Cont-Exact-SC implies the NP-hardness result stated in Theorem 6 of [2]. Moreover, since we (can) assume that the uniform segment length in these restrictions of Cont-SC and Disc-SC is more than 1 (otherwise, the problem is polynomial-time solvable by a simple greedy algorithm), our NP-hardness result for Disc-SC implies the NP-hardness result in Theorem 1 of [3] (since the segment length is more than 1 unit and the intervals are unit intervals, the covering obtained is automatically a conflict-free covering). Second, the NP-hardness results for Cont-SC, Disc-SC, Cont-Exact-SC, and Disc-Exact-SC refine the complexity of these problems. For Cont-SC and Disc-SC, we already know that the slices of these problems consisting of instances in which each segment has length at most 1 unit are solvable in polynomial time by a greedy algorithm. (Note that we do not know if the same holds for Cont-Exact-SC and Disc-Exact-SC, as we do not know the complexity of their restrictions to instances in which each segment has length at most 1.) The above result shows that the slices of Cont-SC, Disc-SC, Cont-Exact-SC, and Disc-Exact-SC, consisting of instances in which all segments have the same length, are NP-hard.

The crucial insight required for our NP-hardness results is that, while the problems are one-dimensional, instances where the length of all segments differs significantly from the length of all intervals in fact behave like two-dimensional objects. We employ this in our proof by devising a series of two reductions, where we begin by considering a two-dimensional segment covering problem whose instances are "nicely" embedded on a grid. We show that this aforementioned problem is NP-hard via a reduction from the restriction of Planar Vertex Cover to instances that are also nicely embedded on a grid. It is worth noting that, while the idea of proving NP-hardness by reducing from a problem with nice embedding properties (e.g., Planar 3-SAT) has been used in previous work [1, 4], the presented reduction stands out due to requiring complex "modularly constructed gadgets." We compose the above reduction with a second one that maps the segment covering problem on the grid to our one-dimensional segment covering problems.

(ii) We show that the restrictions of Cont-SC, Disc-SC, Cont-Exact-SC, and Disc-Exact-SC to instances in which all segments have the same length are fixed-parameter tractable (FPT). The FPT algorithm for Cont-SC combines several algorithmic ideas. (The other FPT algorithms are similar.) It starts by computing an approximate solution of ratio 3 whose intervals contain *all* (input) segment endpoints. The algorithm then branches on all possibilities to determine how the approximate solution "interacts" with an optimal solution. Based on the determined interaction, the algorithm identifies endpoints of segments in the approximate solution, called *anchors*, around which the intervals in an optimal solution are *anchored* (i.e., placed). The goal then becomes to assign the endpoints of the segments to the anchors, where assigning an endpoint to an anchor means that the endpoint (and hence the associated segment) is covered by the interval in the optimal solution placed around that anchor. The algorithm then exploits the restriction that all segments have the same length, to define a domination relation among the anchors affecting each segment, which is then revealed through further branching. With these domination relations revealed, the resulting problem can be modeled as an instance of 2-Sat, which is solvable in polynomial time.

(iii) We show that DISC-SC and CONT-SC are W[1]-complete. Membership in W[1] is proved using the characterization of W[1] by Chen, Flum, and Grohe [7], whereas the W[1]-hardness is proved via an FPT-reduction from the MULTI-COLORED CLIQUE problem. This reduction is quite involved, requiring gadget constructions that extend beyond the standard toolkit used in conventional W[1]-hardness reductions from MULTICOLORED CLIQUE. The W[1]-completeness results, in conjunction with the results in (ii) above, provide tight results for the parameterized complexity of DISC-SC and CONT-SC. Note that, while the restrictions of DISC-SC and CONT-SC to instances in which all segments have equal length have the same classical complexity as their general counterparts, these restrictions exhibit a different behavior in terms of their parameterized complexity. The parameterized complexity of CONT-EXACT-SC and DISC-EXACT-SC remains open.

2. Preliminaries. We assume familiarity with the basic notation and terminology used in graph theory and parameterized complexity. We refer the reader to the standard books [13, 14] for more information on these subjects. The asymptotic notation $\mathcal{O}^*$ suppresses a polynomial factor in the input length. For $r \in \mathbb{N}$, we write $[r]$ as shorthand for the set $\{1, \ldots, r\}$.

2.1. Parameterized complexity. A *parameterized problem* Q is a subset of $\Omega^* \times \mathbb{N}$, where Ω is a fixed alphabet. Each instance of the parameterized problem Q is a pair (x, k), where $k \in \mathbb{N}$ is called the *parameter*. We say that the parameterized problem Q is *FPT* [14], if there is a (parameterized) algorithm, also referred to as an FPT *algorithm*, that decides whether an input (x, k) is a member of Q in time $f(k) \cdot |x|^{O(1)}$, where f is a computable function. Let FPT denote the class of all FPT parameterized problems.

A parameterized problem Q is FPT-*reducible* to a parameterized problem Q' if there is an algorithm, called an FPT-*reduction*, that transforms each instance (x, k) of Q into an instance (x', k') of Q' in time $f(k) \cdot |x|^{O(1)}$, such that $k' \leq g(k)$ and $(x, k) \in Q$ if and only if $(x', k') \in Q'$, where f and g are computable functions. By FPT-*time* we denote time of the form $f(k) \cdot |x|^{O(1)}$, where f is a computable function and $|x|$ is the input instance size.

Based on the notion of FPT-reducibility, a hierarchy of parameterized complexity, *the* W-*hierarchy* $\bigcup_{t \geq 0} \mathsf{W}[t]$, where $\mathsf{W}[t] \subseteq \mathsf{W}[t+1]$ for all $t \geq 0$, has been introduced, in which the 0th level W[0] is the class FPT. The hardness and completeness have been defined for each level W[i] of the W-hierarchy for $i \geq 1$ [14]. It is commonly believed that $\mathsf{W}[1] \neq \mathsf{FPT}$ (see [14]). The notion of W[1]-hardness has served as the main working hypothesis of fixed-parameter intractability.

2.2. Segment covering problems. Let $I = [a, b]$ be an interval on the rational line.[1] We say that an interval $I' = [x, y]$ covers I if at least one endpoint of I lies on I'; that is, if either $a \in [x, y]$ or $b \in [x, y]$. In this paper, we consider the problem of covering a set of intervals (not necessarily of the same length) by unit-length intervals. To distinguish the intervals to be covered from those used for covering, for clarity, throughout the paper we will refer to the former (i.e., an interval to be covered) by a segment. We formulate two versions of the problem, depending on whether or not the covering unit intervals are restricted to a given set:

[1] We assume that the numbers are represented as rationals.

DISCRETE SEGMENT COVERING (DISC-SC)
Given: A set Γ of n segments on the rational line; a set $\mathcal{I}$ of unit intervals on the rational line; $k \in \mathbb{N}$.
Parameter: k.
Question: Can the segments in Γ be covered by at most k intervals from $\mathcal{I}$?

CONTINUOUS SEGMENT COVERING(CONT-SC)
Given: A set Γ of n segments on the rational line; $k \in \mathbb{N}$.
Parameter: k.
Question: Can the segments in Γ be covered by at most k unit intervals?

We also consider variants of the above two problems that have been studied in the literature, where one requires *exactly* one endpoint of each segment in Γ to be covered:

DISCRETE EXACT SEGMENT COVERING (DISC-EXACT-SC)
Given: A set Γ of n segments on the rational line; a set $\mathcal{I}$ of unit intervals on the rational line; $k \in \mathbb{N}$.
Parameter: k.
Question: Does there exist a subset $\mathcal{S} \subseteq \mathcal{I}$ of k intervals such that, for each segment in Γ, exactly one endpoint of the segment is covered by the intervals in $\mathcal{S}$?

CONTINUOUS EXACT SEGMENT COVERING(CONT-EXACT-SC)
Given: A set Γ of n segments on the rational line; $k \in \mathbb{N}$.
Parameter: k.
Question: Does there exist a set $\mathcal{S}$ of k intervals such that, for each segment in Γ, exactly one endpoint of the segment is covered by the intervals in $\mathcal{S}$?

We denote by DISC-EQUAL-SC, CONT-EQUAL-SC, DISC-EQUAL-EXACT-SC, and CONT-EQUAL-EXACT-SC the restrictions of DISC-SC, CONT-SC, DISC-EXACT-SC, and CONT-EXACT-SC, respectively, to instances in which all segments in Γ have the same length. The restrictions of CONT-EQUAL-SC, and DISC-EQUAL-SC to instances in which the length of the segments is at most 1 unit (i.e., at most that of the covering unit intervals) can be easily solved in polynomial time using a greedy approach; therefore, we will assume throughout this paper that the length of the segments in the instances of DISC-EQUAL-SC and CONT-EQUAL-SC is more than 1 unit.

Remark 2.1. There are simple polynomial-time FPT-reductions from CONT-SC and CONT-EXACT-SC to DISC-SC and DISC-EXACT-SC, respectively. The reduction from CONT-SC to DISC-SC follows from the fact that, for an instance of CONT-SC, we can always assume that the left endpoint of each covering unit interval is an endpoint of a segment; if this is not the case for a covering unit interval, we can shift it to the right until its left endpoint coincides with a segment's endpoint.

The reduction from CONT-EXACT-SC to DISC-EXACT-SC follows from the observation that we can transform any solution of CONT-EXACT-SC into a (valid) solution of the same cardinality each of whose intervals either starts at or just before a segment endpoint, or ends at or just before a segment endpoint. The number of unit intervals in the resulting instance of DISC-EXACT-SC obtained is linear in the number of segments in the original instance.

3. Parameterized complexity of DISC-SC and CONT-SC. In this section, we will show that DISC-SC and CONT-SC are W[1]-complete. We start by showing

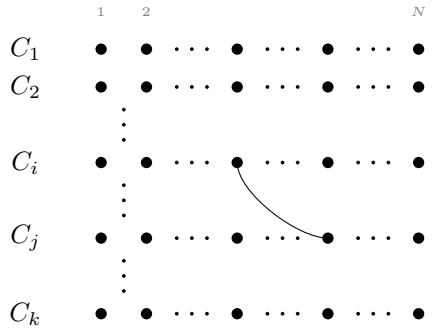

FIG. 1. *An instance of* MULTI-COLORED CLIQUE.

the W[1]-completeness of DISC-SC, which is easier to explain as the set of covering unit intervals is restricted; we then explain at the end how to modify the proof in order to lift this restriction, and obtain a W[1]-hardness proof for CONT-SC as well. We would like to note here that while the reduction we design is not inspired by any particular previous work, reductions of a similar flavor have been used to obtain W[1]-hardness for, e.g., several variants of SAT [20, 6].

We show the W[1]-hardness of DISC-SC via an FPT-reduction from the W[1]-hard problem MULTI-COLORED CLIQUE: given a graph G with a proper k-coloring of its vertices, where each color class has the same cardinality, decide if there exists a *colorful* clique $Q \subseteq V(G)$ (i.e., a clique in which no two vertices have the same color) of size k [19]; the parameter is k. Let (G, k) be an instance of MULTI-COLORED CLIQUE. We assume that each color class C_i, $i \in [k]$, has the same cardinality N [12]. See Figure 1 for an illustration.

The reduction involves constructing three types of gadgets: vertex-selection gadgets, edge-verification gadgets, and edge-synchronization gadgets. The purpose of the vertex-selection gadgets is to encode that k colorful vertices in G are selected, and the purpose of the edge-verification gadgets and the edge-synchronization gadgets is to encode that the selected vertices form a clique in G. All the intervals and segments in the construction lie on the same (horizontal) line.

Vertex gadgets. For each color class $C_i = \{v_1^i, \ldots, v_N^i\}, i \in [k]$, we place a sequence $\mathcal{S}^i$ of N interleaved unit intervals $[v_r^i, v_r'^i]$, $r \in [N]$, on the line, where the starting point of each interval is separated from the starting point of the next by a distance of $1/N$ (i.e., the starting points of the N intervals subdivide a 1 unit distance into N equal parts); that is, $|v_r^i v_{r+1}^i| = 1/N$ for $r \in [N-1]$. Each of these N intervals encodes a vertex of C_i. We add the intervals in $\mathcal{S}^i$, for $i \in [k]$, to $\mathcal{I}$ as covering unit intervals. Two sequences $\mathcal{S}^i, \mathcal{S}^j$, corresponding to two different color classes, are placed far apart on the line so that no two of their intervals overlap. For each sequence $\mathcal{S}^i$, we add the segment $S_i = [v_1^i, v_N^i]$ to Γ, which ensures that any solution must contain at least one interval from $\mathcal{S}^i$ in order to cover S_i. See Figure 2 for an illustration.

Edge-gadgets. For each set of edges E_{ij}, between color classes C_i and C_j, where $i < j \in [k]$, let $m_{ij} = |E_{ij}|$. We place two interleaved sequences of unit intervals. The construction of the two sequences is identical, and is done as follows. Set $M = m_{ij}$. The first sequence $\mathcal{S}_{ij}^1$ consists of unit intervals $[e_r^1, e_r'^1]$, $r \in [M]$, such that $|e_r^1 e_{r+1}^1| = 1/M$ for $r \in [M-1]$; that is, the left endpoints of two consecutive intervals in this

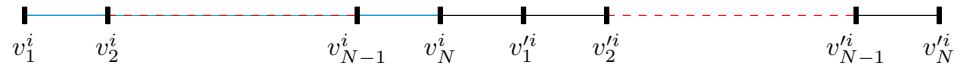

FIG. 2. *Illustration for the vertex gadget construction. The segment* $S_i = [v_1^i, v_N^i]$ *is shown in cyan color. (*Color available online.*)*

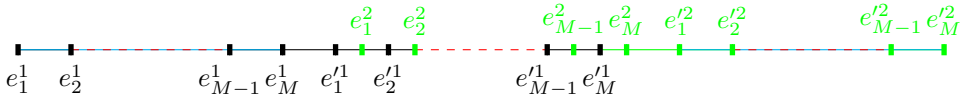

FIG. 3. *Illustration for the edge-gadget construction. The two segments* $S_{ij}^1 = [e_1^1, e_M^1]$ *and* $S_{ij}^2 = [e_1'^2, e_M'^2]$ *are shown in cyan color. (*Color available online.*)*

sequence are at distance $1/M$. Similarly, $\mathcal{S}_{ij}^2$ consists of unit intervals $[e_r^2, e_r'^2]$, $r \in [M]$, such that $|e_r^2 e_{r+1}^2| = 1/M$ for $r \in [M-1]$. We place $\mathcal{S}_{ij}^1$ and $\mathcal{S}_{ij}^2$ on the line in an interleaving fashion, such that the first interval $[e_1^2, e_1'^2]$ in $\mathcal{S}_{ij}^2$ starts $1/(2M)$ units after the first interval $[e_1^1, e_1'^1]$ in $\mathcal{S}_{ij}^1$ ends (and hence, $[e_r^2, e_r'^2]$ in $\mathcal{S}_{ij}^2$ starts $1/(2M)$ units after $[e_r^1, e_r'^1]$ in $\mathcal{S}_{ij}^1$ ends for $r \in [M]$). Putting it differently, the sequence $\mathcal{S}_{ij}^2$ is shifted $1 + 1/(2M)$ units to the right from $\mathcal{S}_{ij}^1$. The reason behind this placement is that, if we assume that copies of the same interval are chosen from $\mathcal{S}_{ij}^1, \mathcal{S}_{ij}^2$, a property that will be ensured by the edge-synchronization gadgets discussed below, then the $1/(2M)$ units gap between an interval $[e_r^1, e_r'^1]$ in $\mathcal{S}_{ij}^1$ and its copy $[e_r^2, e_r'^2]$, $r \in [M]$, in $\mathcal{S}_{ij}^2$ is covered by *any* choice of an interval from $\mathcal{S}_{ij}^1$ and its copy in $\mathcal{S}_{ij}^2$, *except* the choice of $[e_r^1, e_r'^1]$ and $[e_r^2, e_r'^2]$. This property will be crucial for encoding vertex-edge incidences.

Now the two sequences $\mathcal{S}_{ij}^1, \mathcal{S}_{ij}^2$, corresponding to E_{ij}, are placed far apart from other vertex gadgets and edge-gadgets so that their intervals are disjoint from the intervals of other gadgets. We add the unit intervals of $\mathcal{S}_{ij}^1$ and $\mathcal{S}_{ij}^2$ to $\mathcal{I}$ as covering unit interval. We add the two segments $S_{ij}^1 = [e_1^1, e_M^1]$ and $S_{ij}^2 = [e_1'^2, e_M'^2]$ to Γ; these two segments ensure that any solution must contain at least one interval from each of $\mathcal{S}_{ij}^1$ and $\mathcal{S}_{ij}^2$. See Figure 3 for an illustration.

Edge-synchronization gadgets. For each edge-gadget consisting of two sequences $\mathcal{S}_{ij}^1$ in $\mathcal{S}_{ij}^2$ of unit intervals corresponding to the edges in E_{ij}, we construct a sequence of interleaved unit intervals, $\mathcal{S}_{ij}^3$, that is constructed identically to $\mathcal{S}_{ij}^1$ and $\mathcal{S}_{ij}^2$, and place the intervals in $\mathcal{S}_{ij}^3$ on the line so that they are disjoint from all covering intervals in other gadgets, including those in $\mathcal{S}_{ij}^1$ and $\mathcal{S}_{ij}^2$. We add the intervals in $\mathcal{S}_{ij}^3$ to $\mathcal{I}$ as covering unit intervals. We add the segment $S_{ij}^3 = [e_1^3, e_M^3]$ to Γ, which ensures that any solution must contain at least one interval from $\mathcal{S}_{ij}^3$. The intervals in $\mathcal{S}_{ij}^3$ will ensure that, in the desired solution, three copies of the same interval are chosen, one from each of $\mathcal{S}_{ij}^1$, $\mathcal{S}_{ij}^2$, and $\mathcal{S}_{ij}^3$.

Connecting the gadgets. We encode vertex-edge incidences in G by adding segments between vertex gadgets and corresponding edge-gadgets. For a vertex v_r, $r \in [N]$, in color class C_i (resp., C_j) in G, and an edge e_{ij} incident to v_r and to some vertex in color class C_j (resp., C_i), let $[e_s^1, e_s'^1]$ and $[e_s^2, e_s'^2]$ be the intervals corresponding to e_{ij} in $\mathcal{S}_{ij}^1$ and $\mathcal{S}_{ij}^2$, respectively. Do the following (see Figure 4 for an illustration):

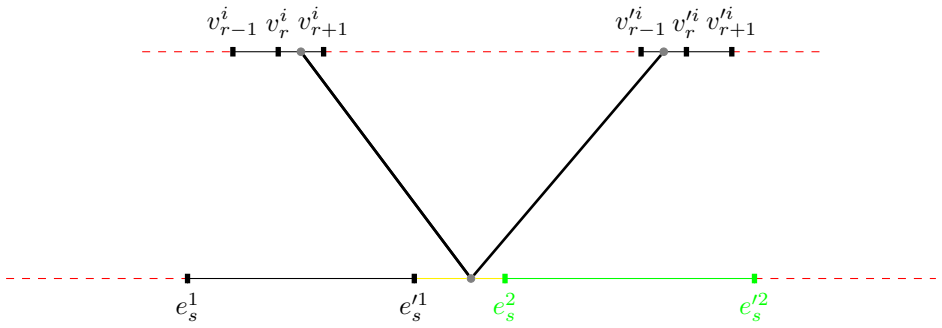

FIG. 4. *Illustration for the connection between a vertex gadget and an edge-gadget that encodes vertex-edge incidency. Only the relevant portions of the gadgets are shown, and for clarity, the two gadgets are drawn on top of one another, rather than on the same line. The yellow interval* $[e'^1_s e^2_s]$ *is the region of the edge-gadget that encodes incidences to edge* e_{ij}*, represented in the gadget by the two copies* $[e^1_s e'^1_s]$ *and* $[e^2_s e'^2_s]$*. The edge-synchronization gadgets will ensure that two copies of the same edge are selected from each edge-gadget. Under this assumption, the interval* $[e'^1_s e^2_s]$ *is* not *covered if and only if the two copies* $[e^1_s e'^1_s]$ *and* $[e^2_s e'^2_s]$*, corresponding to edge* e_{ij}*, are selected in the edge-gadget. (*Color available online.*)*

1. if $r < N$, create a segment with one endpoint (strictly) between v^i_r and v^i_{r+1} in $\mathcal{S}^i$ (resp., between v^j_r and v^j_{r+1} in $\mathcal{S}^j$), and the other (strictly) between e'^1_s and e^2_s; and
2. if $r > 1$, create a segment with one endpoint (strictly) between v'^i_{r-1} and v'^i_r in $\mathcal{S}^i$ (resp., between v'^j_{r-1} and v'^j_r in $\mathcal{S}^j$), and the other (strictly) between e'^1_s and e^2_s.

Add the created segments in (1) and (2) above to Γ.

We encode edge synchronization for each triplet of sequences $\mathcal{S}^1_{ij}$, $\mathcal{S}^2_{ij}$, and $\mathcal{S}^3_{ij}$, corresponding to the edges in E_{ij}, where $|E_{ij}| = m_{ij}$, as follows. For each $s \in [m_{ij}-1]$, do the following (see Figure 5 for an illustration):

(i) Create a segment with one endpoint (strictly) between e^1_s and e^1_{s+1} and the other (strictly) between e'^3_s and e'^3_{s+1}.
(ii) Create a segment with one endpoint (strictly) between e^2_s and e^2_{s+1} and the other (strictly) between e'^3_s and e'^3_{s+1}.
(iii) Create a segment with one endpoint (strictly) between e^3_s and e^3_{s+1} and the other (strictly) between e'^1_s and e'^1_{s+1}.
(iv) Create a segment with one endpoint (strictly) between e^3_s and e^3_{s+1} and the other (strictly) between e'^2_s and e'^2_{s+1}.

Add the created segments in (i)–(iv) above to Γ.

3.1. Putting it all together.

LEMMA 3.1. *For any* $i < j \in [k]$*, three intervals, one in each of* $\mathcal{S}^1_{ij}$*,* $\mathcal{S}^2_{ij}$*, and* $\mathcal{S}^3_{ij}$*, cover all segments with one endpoint lying on an interval in* $\mathcal{S}^1_{ij} \cup \mathcal{S}^2_{ij}$ *and the other endpoint lying on an interval in* $\mathcal{S}^3_{ij}$ *if and only if the three intervals are copies of the same interval; that is, the three intervals are of the form* $[e^1_s, e'^1_s], [e^2_s, e'^2_s], [e^3_s, e'^3_s]$ *for some* $s \in [m_{ij}]$*.*

Proof. For $s \in [m_{ij}]$, let $I^1_s = [e^1_s, e'^1_s]$, $I^2_s = [e^2_s, e'^2_s]$, and $I^3_s = [e^3_s, e'^3_s]$. To prove the lemma, we first note the following assertions that follow from steps (i)–(iv) in the encoding of edge synchronization. All segments with one endpoint lying on an

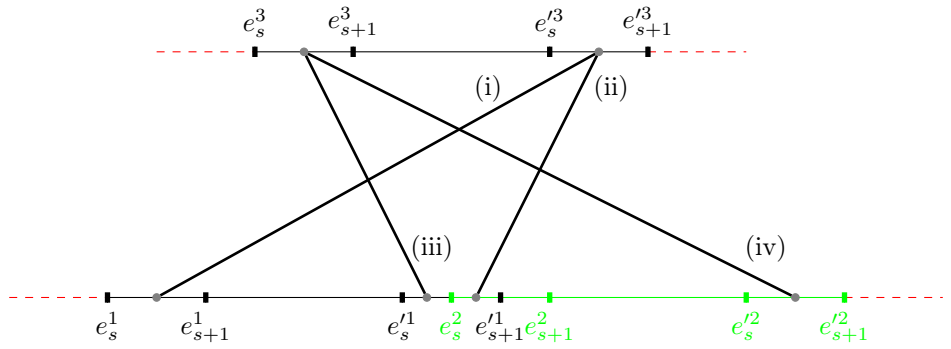

FIG. 5. *Illustration of the edge-synchronization gadget construction. For clarity, only the relevant portion of the figure is shown, and $\mathcal{S}^3_{ij}$ is drawn on top of $\mathcal{S}^1_{ij}$ and $\mathcal{S}^2_{ij}$. (*Color available online.*)*

interval in $\mathcal{S}^1_{ij} \cup \mathcal{S}^2_{ij}$ and the other endpoint lying on an interval in $\mathcal{S}^3_{ij}$ are covered if and only if, for every $s \in [m_{ij} - 1]$, the following hold:

(A1) Either an interval I^1_p with $p \leq s$, or I^3_r with $r > s$ is chosen in the solution. This follows from step (i) in the encoding of edge synchronization.

(A2) Either an interval I^2_q with $q \leq s$, I^1_p with $p > s$, or I^3_r with $r > s$, is chosen in the solution. This follows from step (ii) in the encoding of edge synchronization in conjunction with (A1).

(A3) Either an interval I^3_r with $r \leq s$ is chosen in the solution, or an interval I^1_p and an interval I^2_q with $p, q > s$ are chosen in the solution. This follows from steps (iii) and (iv) in the encoding of edge synchronization.

First, note that, for any $s \in [m_{ij}]$, choosing the three copies I^1_s, I^2_s, I^3_s of the same interval satisfy all three assertions above, and hence cover all the segments in question. To prove the converse, suppose that three intervals I^1_p, I^2_q, I^3_r are chosen from $\mathcal{S}^1_{ij}$, $\mathcal{S}^2_{ij}$, and $\mathcal{S}^3_{ij}$, respectively, that cover all segments with one endpoint lying on an interval in $\mathcal{S}^1_{ij} \cup \mathcal{S}^2_{ij}$ and the other endpoint lying on an interval in $\mathcal{S}^3_{ij}$. We will show that $p = q = r$.

If $r = m_{ij}$, then (A3) applied with $s = m_{ij} - 1$ implies that $p, q > m_{ij} - 1$, and hence $p = q = r = m_{ij}$. Similarly, if $r = 1$, then (A1) and (A2) applied with $s = 1$ (and noting that $r = 1$) imply that $p = q = 1$. Assume now that $1 < r < m_{ij}$. First, apply (A3) with $s = r - 1$; this yields that $p, q \geq r$. Then, apply (A1) and (A2) with $s = r$; (A1) yields that $p \leq r$, and this, in conjunction with (A2) yields that $q \leq r$. We conclude that $p = q = r$, as stated. ∎

LEMMA 3.2. *For any $i \in [k]$ (resp., $j \in [k]$), three intervals, one in each of $\mathcal{S}^i$ (resp., $\mathcal{S}^j$), $\mathcal{S}^1_{ij}$, and $\mathcal{S}^2_{ij}$, such that the two intervals in $\mathcal{S}^1_{ij}$ and $\mathcal{S}^2_{ij}$ are copies of the same interval, cover all segments with one endpoint lying in $\mathcal{S}^i$ (resp., $\mathcal{S}^j$) and the other in $\mathcal{S}^1_{ij} \cup \mathcal{S}^2_{ij}$ if and only if the interval in $\mathcal{S}^i$ (resp., $\mathcal{S}^j$) corresponds to a vertex $v_r \in C_i$ (resp., $v_r \in C_j$), and the two intervals in $\mathcal{S}^1_{ij}$ and $\mathcal{S}^2_{ij}$ correspond to an edge $e_{ij} \in E_{ij}$ that is incident to v_r.*

Proof. Suppose that the two copies chosen from $\mathcal{S}^1_{ij}$ and $\mathcal{S}^2_{ij}$ are $[e^1_s, e'^1_s]$ and $[e^2_s, e'^2_s]$, respectively, and that they correspond to an edge e_{ij}. The only segments with one endpoint lying in $\mathcal{S}^i$ (resp., $\mathcal{S}^j$) and the other in $\mathcal{S}^1_{ij} \cup \mathcal{S}^2_{ij}$ that are covered neither by $[e^1_s, e'^1_s]$ nor by $[e^2_s, e'^2_s]$, are precisely those whose endpoints in $\mathcal{S}^1_{ij} \cup \mathcal{S}^2_{ij}$

fall on $[e'^1_s, e^2_s]$. Due to the encoding of vertex-edge incidences, the other endpoints in $\mathcal{S}^i$ (resp., $\mathcal{S}^j$) of such segments are in $[v^i_p, v^i_{p+1}]$ (resp., $[v^j_p, v^j_{p+1}]$) (if $p < N$) and $[v'^i_{p-1}, v'^i_p]$ (resp., $[v'^j_{p-1}, v'^j_p]$) (if $p > 1$), where $v_p \in C_i$ (resp., $v_p \in C_j$) is incident to e_{ij}, and hence those endpoints in $\mathcal{S}^i$ (resp., $\mathcal{S}^j$) are covered *only* by the interval $[v^i_p, v'^i_p] \in \mathcal{S}^i$ (resp., $[v^j_p, v'^j_p] \in \mathcal{S}^j$) corresponding to v_p. The statement of the lemma follows (both directions). ∎

LEMMA 3.3. DISC-SC *is in* W[1].

Proof. To prove membership in W[1], we use the characterization of W[1] given by Chen, Flum, and Grohe [7]:

> A parameterized problem Q is in W[1] if and only if there is a computable function h and a nondeterministic FPT algorithm $\mathbb{A}$ for a nondeterministic-RAM machine deciding Q, such that, for each instance (x, k) of Q (k is the parameter), all nondeterministic steps of $\mathbb{A}$ take place during the last $h(k)$ steps of the computation.

By the above characterization, it suffices to exhibit such a nondeterministic FPT algorithm $\mathbb{A}$ for DISC-SC.

Given an instance $I = (\Gamma, \mathcal{I}, k)$ of DISC-SC, the algorithm $\mathbb{A}$ starts by performing a preprocessing phase. In this phase, the algorithm iterates through every two endpoints a, b, where $a < b$, of two intervals in $\mathcal{I}$ such that $[a, b]$ is a subinterval (not necessarily proper) of some interval in $\mathcal{I}$. For each two such endpoints a, b, the algorithm computes, for each interval, I, of the two intervals $[a, b], [a, b)$, the number of segments in Γ such that (at least) one of whose endpoints lies on I (and hence is covered by I). The algorithm creates a table T—indexed by intervals, and stores in $T[I]$ the number of segments in Γ covered by I. Then, the algorithm computes, for every two distinct such subintervals, I and I', of intervals in $\mathcal{I}$ (i.e., for every $I \in \{[a, b], [a, b)\}, I' \in \{[c, d], [c, d)\}$, where $a < b$ and $c < d$), the number of segments in Γ covered simultaneously by *both* I and I'; we store this number in another table M—indexed by pairs of intervals—at entry $M[I, I']$. Clearly, this computation, including the creation of the two tables T and M, can be done in polynomial time.

After the above preprocessing phase is complete, $\mathbb{A}$ (nondeterministically) guesses a set S of k covering unit intervals in $\mathcal{I}$. Sorting the endpoints of the intervals in S (from left to right) induces a partitioning, Π, of those intervals into subintervals each of which is either closed or right half-open; the union of the subintervals in Π is the same as the union of the intervals in S, and no two distinct subintervals in Π overlap. Note that, since the intervals in Π are pairwise disjoint, a segment in Γ can be covered by at most two intervals in Π, and the segment is covered by two intervals in Π if the two endpoints of the segment are contained in two different intervals in Π. Then, we iterate through the subintervals in Π, and for each subinterval I, we add the number of segments it covers, which can be fetched in constant time using the table T; let N be the total of all the numbers added. By doing so, we could be double counting some segments, which are precisely those segments each of whose two endpoints lies on one subinterval in Π. Now we iterate over (unordered) pairs of distinct subintervals in Π, and for each pair I, I', we subtract from N the number of segments in Γ that are covered by both I and I', which could be fetched in constant time using the table M. The algorithm $\mathbb{A}$ accepts its input if and only if the resulting number at the end is $|\Gamma|$.

It is not difficult to see that $\mathbb{A}$ correctly decides DISC-SC. Clearly, all the computation done by $\mathbb{A}$ after the preprocessing phase, including the nondeterministic steps, are upper bounded by $h(k)$, where h is a computable function, and hence $\mathbb{A}$ meets the required conditions in the characterization of W[1] stated above. ∎

THEOREM 3.4. DISC-SC *is* W[1]*-complete.*

Proof. By Lemma 3.3, it suffices to show that DISC-SC is W[1]-hard. We do so via a reduction from the W[1]-hard problem MULTI-COLORED CLIQUE [19]. Let (G, k) be an instance of MULTI-COLORED CLIQUE, where each color class C_i, $i \in [k]$, has the same cardinality/size N. We assume that G contains at least one edge between every pair of color classes; otherwise, the instance can be directly mapped to a no-instance of DISC-SC. We construct the instance $(\Gamma, \mathcal{I}, k')$ of DISC-SC as follows. The sets $\mathcal{I}$ and Γ are constructed by constructing the vertex gadget $\mathcal{S}^i$, for each color class C_i, $i \in [k]$, the edge-gadgets and edge-synchronization gadgets $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}$, for each set of edges E_{ij} between two color classes C_i, C_j, $i < j \in [k]$, and their connections, as described in this section. Finally, we set $k' = k + 3\binom{k}{2}$.

If (G, k) is a YES-instance of MULTI-COLORED CLIQUE, then G has a clique Q of size k, consisting of a vertex from each color class C_i, $i \in [k]$. Let S be the set of intervals in $\mathcal{I}$ chosen as follows. From each $\mathcal{S}^i$, choose the interval $[v^i_r, v'^i_r]$, where $v_r \in C_i \cap Q$, and from each of $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}$, $i < j \in [k]$, choose the three copies $[e^1_s, e'^1_s]$, $[e^2_s, e'^2_s]$, $[e^3_s, e'^3_s]$ corresponding to edge e_{ij} between the vertices from color classes C_i, C_j chosen in Q. The number of chosen covering unit intervals is precisely $k' = k + 3\binom{k}{2}$. Since an interval from each $\mathcal{S}^i$, $i \in [k]$, and each of $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}$, $i < j \in [k]$, is chosen, the segment S_i in Γ contained in $\mathcal{S}^i$, $i \in [k]$, and the segments, $S^1_{ij}, S^2_{ij}, S^3_{ij}$ in $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}$, respectively, where $i < j \in [k]$, are all covered. The other segments in Γ are either (1) segments with one endpoint lying on an interval in $\mathcal{S}^1_{ij} \cup \mathcal{S}^2_{ij}$ and the other in $\mathcal{S}^3_{ij}$, $i < j \in [k]$; or (2) segments with one endpoint in $\mathcal{S}^i$ (resp., $\mathcal{S}^j$), $i \in [k]$ (resp., $j \in [k]$), and the other in $\mathcal{S}^1_{ij} \cup \mathcal{S}^2_{ij}, i < j \in [k]$. By Lemma 3.1, the segments in (1) are covered by S, and by Lemma 3.2, the segments in (2) are covered by S. This shows that $(\Gamma, \mathcal{I}, k')$ is a YES-instance of DISC-SC.

Conversely, suppose that $(\Gamma, \mathcal{I}, k')$ is a YES-instance of DISC-SC, and let S be a solution to $(\Gamma, \mathcal{I}, k')$. Since $k' = k + 3\binom{k}{2}$, and there are precisely k' segments of the form $S_i, i \in [k]$, or $S^1_{ij}, S^2_{ij}, S^3_{ij}, i < j \in [k]$, k of which are in $\mathcal{S}^i$, $i \in [k]$, and $3\binom{k}{2}$ in $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}, i < j \in [k]$, and since the segments in any sequence in $\mathcal{S}^i, \mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}$ are far away from those in any other sequence (i.e., they cannot be simultaneously covered by the same covering unit interval), S contains exactly one interval from each $\mathcal{S}^i$, $i \in [k]$, and one interval from each of $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}, i < j \in [k]$. By Lemma 3.1, the intervals in $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}, i < j \in [k]$, must be copies of the same interval, and hence correspond to the same edge e_{ij} between C_i and C_j. Let Q be the set of k vertices corresponding to the intervals chosen from $\mathcal{S}^i, i \in [k]$. For any $i \in [k]$ (resp., $j \in [k]$), since S covers all segments with one endpoint in $\mathcal{S}^i$ (resp., $\mathcal{S}^j$) and the other in $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}$, $i < j \in [k]$, by Lemma 3.2 the intervals chosen from $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}$ correspond to an edge incident to the vertex in color C_i (resp., C_j) in Q. It follows that there are $\binom{k}{2}$ edges in G, corresponding to the chosen intervals in $\mathcal{S}^1_{ij}$, $i < j \in [k]$, that are incident to the vertices in Q, and hence Q is a clique in G. ∎

COROLLARY 3.5. CONT-SC *is* W[1]*-complete.*

Proof. Membership in W[1] follows from the FPT-reduction to DISC-SC (see Remark 2.1). To show the W[1]-hardness of CONT-SC, we explain how the gadgets construction in the W[1]-hardness proof of DISC-SC can be modified so that the choices of the covering unit intervals in an instance of CONT-SC are automatically restricted to those placed in $\mathcal{I}$ in the original reduction. This allows the proof of the W[1]-hardness of CONT-SC to seamlessly mimic that of DISC-SC.

We define the sequences of unit intervals $\mathcal{S}^i$, $i \in [k]$, and $\mathcal{S}^1_{ij}, \mathcal{S}^2_{ij}, \mathcal{S}^3_{ij}$, $i < j \in [k]$, and the segments $S_i, S^1_{ij}, S^2_{ij}, S^3_{ij} \in \Gamma$ as before. However, we modify the connections between the gadgets as follows.

For a vertex v_r, $r \in [N]$, in color class C_i (resp., C_j), and an edge e_{ij} incident to v_r, let $[v^i_r, v'^i_r]$ (resp., $[v^j_r, v'^j_r]$) be the interval in $\mathcal{S}^i$ (resp., $\mathcal{S}^j$) corresponding to v_r, and $[e^1_s, e'^1_s]$ and $[e^2_s, e'^2_s]$ be the intervals corresponding to e_{ij} in $\mathcal{S}^1_{ij}$ and $\mathcal{S}^2_{ij}$, respectively. If $r < N$, create a segment with one endpoint being v^i_r (resp., v^j_r), and the other (strictly) between e'^1_s and e^2_s. If $r > 1$, create a segment with one endpoint being v'^i_r (resp., v'^j_r) in $\mathcal{S}_i$ (resp., $\mathcal{S}_j$), and the other (strictly) between e'^1_s and e^2_s.

We modify the connections encoding edge synchronization, for each triplet of sequences $\mathcal{S}^1_{ij}$, $\mathcal{S}^2_{ij}$, $\mathcal{S}^3_{ij}$, corresponding to the edges in E_{ij}, where $|E_{ij}| = m_{ij}$, as follows. For each $s \in [m_{ij} - 1]$, do the following:

(i) Create a segment with one endpoint e^1_s and the other e'^3_{s+1}.
(ii) Create a segment with one endpoint e^2_s and the other e'^3_{s+1}.
(iii) Create a segment with one endpoint e^3_s and the other e'^1_{s+1}.
(iv) Create a segment with one endpoint e^3_s and the other e'^2_{s+1}.

The above modifications, together with the assumption that each covering unit interval in a solution starts at a segment's endpoint (see Remark 2.1), allows assertions (A1), A(2), (A3), and the proofs of Lemmas 3.1 and 3.2, to go through without any change. This implies that a solution of an instance of Cont-SC contains an interval from each $\mathcal{S}^i$, $i \in [k]$, and copies of the same edge from $\mathcal{S}^1_{ij}$, $\mathcal{S}^2_{ij}$, and $\mathcal{S}^3_{ij}$ for $i < j \in [k]$. The rest of the proof follows in the same fashion as the proof of Theorem 3.4. ∎

4. NP-completeness of the equal segment-length variants. In this section we show that all of our considered problems are NP-complete even when restricted to the case where all segments have equal length (i.e., Disc-Equal-SC, Cont-Equal-SC, Disc-Equal-Exact-SC, and Cont-Equal-Exact-SC). We do so via a two-step reduction through the following intermediate problem:

Grid Segment Covering
Given: A set Γ of n vertical segments, each of length 1, with endpoints on a $q \times q$ grid with $q \leq 100 \cdot n$; $k \in \mathbb{N}$.
Parameter: k.
Question: Can the segments in Γ be covered by at most k horizontal segments of length 2?

For consistency with the terminology used in this paper (in the definition of segment covering problems under consideration), we will abuse the notation and refer to the horizontal (covering) segments of length 2 as intervals, and to the vertical segments of length 1 (to be covered) as segments. We define the Exact Grid Segment Covering analogously to Grid Segment Covering, with the sole distinction being that each segment in Γ must have precisely one endpoint covered by the solution (i.e., the set of intervals).

The main technical obstacle on the way to the NP-hardness of our problems of interest lies in showing that Grid Segment Covering and Exact Grid Segment Covering are NP-hard; indeed, this is significantly more difficult than the reductions from Grid Segment Covering to Disc-Equal-SC and Cont-Equal-SC and from Exact Grid Segment Covering to Disc-Equal-Exact-SC and Cont-Equal-Exact-SC, which can be found at the end of this section.

We will need some additional notions to obtain the NP-hardness of (Exact) Grid Segment Covering. An *orthogonal drawing* of a graph is an embedding in the plane such that vertices are drawn as points and edges are drawn as sequences of horizontal and vertical line segments. An orthogonal drawing is *plane* if edges intersect only at their endpoints. The following theorem shows that every planar graph admits a plane orthogonal drawing.

Theorem 4.1 (Theorem 5.9 of [21]). *Given a planar graph G of degree at most 3 with $n > 4$ vertices, there is a linear time algorithm that constructs a plane orthogonal drawing of G on an $\lfloor \frac{n}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor$ grid with at most $\lfloor \frac{n}{2} \rfloor + 1$ bends[2] in total, and with the property that there is a spanning tree of $n-1$ straight-line edges, while all nontree edges have at most one bend.*

4.1. Grid Segment Covering is NP-complete. We reduce from a restricted variant of the Planar 3-Vertex Cover problem [17], which is formalized below.

Planar 3-Vertex Cover
Given: A planar graph G with maximum degree 3; $\ell \in \mathbb{N}$.
Question: Does there exists a set of vertices S of size at most ℓ such that each edge of G is incident with a vertex in S?

The starting point of our reduction is an instance of Planar 3-Vertex Cover with an orthogonal plane embedding whose edges have no bends. To show that the problem remains NP-hard even when restricted to such instances, we will make use of the following folklore observation (see, e.g., the works of Clark, Colbourn, and Johnson [9] and of Chlebík and Chlebíková [8]).

Observation 4.2. *Let G be a graph and let $G_{x,y}$ be the graph obtained from G by replacing the edge $\{x, y\} \in E(G)$ by a path xx_1y_1y, where x_1, y_1 are two new vertices. Then G admits a vertex cover of size k if and only if $G_{x,y}$ admits a vertex cover of size $k + 1$.*

Proof. If X is a vertex cover of G of size at most k, then X clearly covers all the common edges of $G_{x,y}$ and G. Moreover, since $\{x, y\}$ is an edge of G, $|\{x, y\} \cap X| \geq 1$. If $x \in X$, then clearly $X \cup \{y_1\}$ is a vertex cover of $G_{x,y}$; otherwise, $y \in X$ and $X \cup \{x_1\}$ is a vertex cover of $G_{x,y}$. On the other hand, if X' is a vertex cover of $G_{x,y}$, then $X' \setminus \{x_1, y_1\}$ covers all the edges that G and $G_{x,y}$ have in common. Now if $x_1 \notin X'$, then clearly $\{x, y_1\} \subseteq X'$ and $X' \setminus \{x_1, y_1\}$ is a vertex cover of G of size $|X'| - 1$. Similarly, if $y_1 \notin X'$, then $X' \setminus \{x_1, y_1\}$ is a vertex cover of G of size $|X'| - 1$. Otherwise, $\{x_1, y_1\} \subseteq X'$ and $(X' \setminus \{x_1, y_1\}) \cup \{x\}$ is a vertex cover of G of size $|X'| - 1$. ∎

We can now establish the NP-hardness of the desired restriction.

Theorem 4.3. Planar 3-Vertex Cover *is NP-hard when restricted to instances with n vertices and a plane orthogonal drawing on an $n \times n$ grid, with no bends, even when such an embedding is provided as input.*

Proof. We reduce from Planar 3-Vertex Cover. Given an instance G of Planar 3-Vertex Cover we can, by Theorem 4.1, construct a plane orthogonal drawing Ω of G on an $\lfloor \frac{n}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor$ grid with at most one bend per edge. Now we create from G and Ω a reduced graph G' together with a straight-line plane orthogonal drawing Ω' of G' as follows. The graph G' is precisely G after we subdivide every

[2] A *bend* is the meeting point of a horizontal and a vertical line in the drawing of an edge.

edge with a bend in Ω twice. We obtain the embedding Ω' from Ω by embedding all the vertices in $V(G') \cap V(G)$ using Ω. At this point, we also refine the grid underlying Ω by a factor of 2—more formally, we replace each cell in the grid underlying Ω with a 2×2 subgrid.

For every edge $e = \{x, y\} \in E(G)$ with a bend, let x_e and y_e be the two newly created vertices of G' after the subdivision of the edge e, such that G' contains the path xx_ey_ey. We let $\Omega'(x_e)$ be the point where $\Omega(e)$ bends and let $\Omega'(y_e)$ be the point in the middle between $\Omega'(x_e)$ and $\Omega'(y)$. The edges in G' are embedded by Ω' as straight lines between their endpoints. It is easy to see that Ω is a straight-line plane orthogonal drawing of G'. Moreover, it follows by applying Observation 4.2 to each edge with a bend that G admits a vertex cover of size at most k if and only if G' admits a vertex cover of size $k + m$, where m is the number of edges with a bend in Ω. ∎

With Theorem 4.3 in hand, we can proceed to the description of the reduction strategy. Given an n-vertex instance (G, ℓ) of Planar 3-Vertex Cover, the first step is to invoke Theorem 4.3 to obtain a plane orthogonal drawing Ω of G on an $n \times n$ grid satisfying the property that every edge is a horizontal or a vertical line segment in this grid. Next, we refine the grid underlying Ω by a factor of 100—more formally, we replace each cell in the grid underlying Ω with a 100×100 subgrid. We will use the following coordinate system to refer to points on the grid: let the leftmost point of a vertex appearing in G have 100 as its x-coordinate, and let the bottommost point of a vertex appearing in G have 100 as its y-coordinate.

We first outline the reduction. The reduction will represent each vertex v in G with a *vertex gadget* $\alpha(v)$. This gadget consists of a set of segments placed along the border of a geometric object that is roughly centered around the position of v in Ω, and that extend in the directions of the edges incident to v. These vertex gadgets will have the following properties:

- *Vertical connections*: If v and w are adjacent vertices in G, then there is an "interface" spanning a 2×3 subgrid such that either $\alpha(v)$ is placed at the bottom right of the subgrid and $\alpha(w)$ at the top left, or vice versa. This property will be used by the *edge gadget* $\beta(uv)$.
- *Duality of choice*: There are two "optimal configurations" of intervals that allow us to cover all segments in $\alpha(v)$: one uses the minimum number of intervals required but does not help us cover the segments located in the edge gadgets connected to $\alpha(v)$, while the other requires one extra interval but also helps us cover segments in the edge gadgets connected to $\alpha(v)$. These optimal configurations cover each segment in $\alpha(v)$ only once.

The following lemma formalizes the precise properties required from the vertex gadgets, and its proof gives the technical details of how such gadgets can be constructed. Further intuition about the construction of the gadgets is provided in Figure 6.

Lemma 4.4. *Given G, ℓ, Ω as above, in polynomial time we can construct a mapping α from the vertex set $V(G)$ that maps each $v \in V(G)$ to a gadget $\alpha(v)$. Each such gadget $\alpha(v)$ consists of a set of $|\alpha(v)|$ segments and up to three "link" points, each corresponding to an edge incident to v, with the following properties:*

1. *Any interval that can cover segments from $\alpha(v)$ cannot cover segments from any $\alpha(w)$ for $w \neq v$.*
2. *There exists no set of intervals of size less than $\mathsf{cost}(\alpha(v)) = \frac{|\alpha(v)|-1}{2}$ that covers all segments in $\alpha(v)$; moreover, there exists a set of intervals of size $\mathsf{cost}(\alpha(v))$ which is an exact covering of all segments in $\alpha(v)$.*

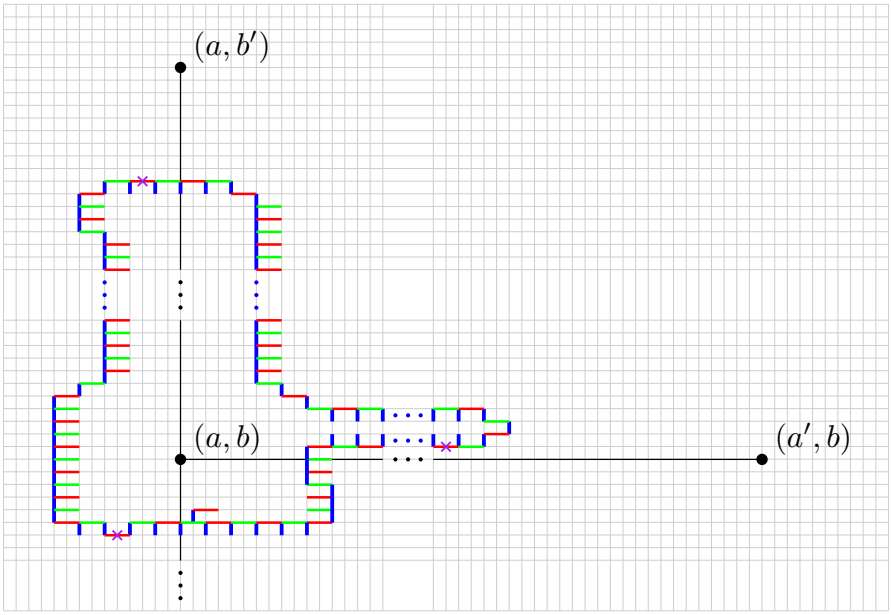

FIG. 6. *A vertex gadget (blue) for a vertex v located at point (a, b). Link points are marked by purple crosses. The set of green intervals has size $\mathsf{cost}(\alpha(v))$ and exactly covers $\alpha(v)$. The set of red intervals has size $\mathsf{cost}(\alpha(v)) + 1$ and covers $\alpha(v)$ and all the link points. (*Color available online.*)*

3. *There exists no set of intervals of size less than $\mathsf{cost}(\alpha(v)) + 1$ that covers all segments in $\alpha(v)$ together with at least one link point of $\alpha(v)$; moreover, there exists a set of intervals of size $\mathsf{cost}(\alpha(v)) + 1$ which is an exact covering of all segments in $\alpha(v)$ and additionally covers all link points of $\alpha(v)$.*
4. *For each edge $vw \in E(G)$ with two link points (x_v, y_v) and (x_w, y_w), it holds that either $x_v = x_w + 2$ and $y_v = y_w - 3$ or vice versa.*

Proof. Let v be a vertex located at point (a, b), and recall that a and b are multiples of 100. The set of segments that make up $\alpha(v)$ will be divided into four subsets (denoted $\alpha_{\downarrow}(v)$, $\alpha_{\leftarrow}(v)$, $\alpha_{\uparrow}(v)$, $\alpha_{\rightarrow}(v)$—each corresponding to one "arm" of $\alpha(v)$). The precise segments that appear in each arm depend on the drawing of the edges incident to v, and we describe them for each of the four arms below. For ease of presentation, we will often say "add a segment below (i, j)" or "add a segment above $(i, j-1)$" as shorthand for "add a segment between (i, j) and $(i, j-1)$." We refer to Figure 6, which will help in understanding the construction formalized below and the subsequent explanations.

A. Definition of $\alpha_{\downarrow}(v)$:

1. If v has no incident edge connected from below, then for each $i \in \{a-8, a-6, \ldots, a+8, a+10\}$ add a *bottom* segment below $(i, b-5)$. Furthermore, add a special *marker segment* above $(a+1, b-5)$.
2. If v has an incident edge connected from below leading to a vertex w with coordinates (a, b'), then for each $i \in \{a-8, a-6, \ldots, a+8, a+10\}$ add a segment below $(i, b-5)$ (this is similar to the previous case). As before, add a special marker segment above $(a+1, b-5)$. Moreover, denote the point $(a-5, b-6)$ as the *v-w link point* in $\alpha(v)$.

Observe that, in both configurations, any single interval can only cover at most two nonmarker segments in $\alpha_{\downarrow}(v)$. Moreover, we will later see that the only two other segments in $\alpha(v)$ which can be covered by an interval that covers at least one segment in $\alpha_{\downarrow}(v)$ are the segments below $(a-10, b-4)$ (in $\alpha_{\leftarrow}(v)$) and $(a+10, b-4)$ (in $\alpha_{\rightarrow}(v)$); both of these segments will always be present in $\alpha(v)$.

For both configurations of $\alpha_{\downarrow}(v)$, consider the following set of intervals[3]: $\{(a-8, b-5), (a-4, b-5), (a, b-5), (a+4, b-5), (a+8, b-5)\}$. We call this interval set the *noncovering solution* for $\alpha_{\downarrow}(v)$. Note that the noncovering solution is an optimal covering for $\alpha_{\downarrow}(v)$. To see this, recall that any interval can only cover at most two segments in $\alpha_{\downarrow}(v)$, and observe that there exists a unique interval I that covers the marker segment as well as two nonmarker segments (the interval with first coordinate a). The noncovering solution not only includes I, but also ensures that each other interval also covers two segments.

Before proceeding, we make two more critical remarks. First of all, no set of intervals of the same size as the noncovering solution can cover all segments in $\alpha_{\downarrow}(v)$ and the position of the v-w link point (located at first coordinate $a-5$) simultaneously. At the end of the construction, it will become clear that the noncovering solutions for the individual arms of α can be combined in a way which has each interval covering precisely two nonmarker segments (representing an optimal covering for $\alpha(v)$). Crucially, due to the "parity" of the link points compared to the location of the marker segment, any such optimal solution for $\alpha(v)$ cannot cover any of the link points in the vertex gadget.

However, if we allow the use of a single extra interval for $\alpha_{\downarrow}(v)$, then we can cover all segments in $\alpha_{\downarrow}(v)$, the v-w link point, and also interface with the rest of $\alpha(v)$ in a way that allows us to optimally cover the three other arms while covering all v-w link points located in those arms. This is achieved by a "parity swap" which uses the extra interval to cover the marker separately; we call this the *covering solution*, consisting of:

$$\{(a-10, b-5), (a-6, b-6), (a-2, b-5), (a+1, b-4), (a+2, b-5), (a+6, b-5), (a+10, b-5)\}.$$

Note that the covering solution uses two more intervals than the noncovering solution for $\alpha_{\downarrow}(v)$—this is due to the fact that it also helps cover two segments in other arms, which will later (in $\alpha_{\rightarrow}(v)$) save one interval and results in a total cost difference of $+1$.

B. Definition of $\alpha_{\leftarrow}(v)$:

1. If v has no incident edge connected from the left, then for each $i \in \{b-4, \ldots, b+4, b+5\}$ add a *left* segment below $(a-10, i)$.
2. If v has an incident edge connected from the left and this edge leads to a vertex w with coordinates (a', b), then add the following segments:
   - for each $i \in \{a-10, a-12, a-14, \ldots, a-28\}$, the segment below $(i, b-4)$;
   - the segment below $(a-30, b-3)$;
   - for each $i \in \{a-28, a-26, \ldots, a-14\}$, the segment below $(i, b-2)$, and furthermore mark $(a-23, b-2)$ as the v-w link point in $\alpha(v)$; and
   - the segments below $(a-12, b-1)$, $(a-10, b)$, $(a-10, b+1)$, $(a-10, b+2)$, $(a-10, b+3)$, $(a-10, b+4)$, $(a-10, b+5)$.

If the noncovering solution was used for $\alpha_{\downarrow}(v)$, then the segment below $(a-10, b-4)$ is not covered yet. Such a noncovering solution may be extended towards $\alpha_{\leftarrow}(v)$ by using the following intervals:

[3]For brevity, we represent intervals by specifying their left endpoint.

1. $(a-10,b-4)$,$(a-10,b-2)$,$(a-10,b)$,$(a-10,b+2)$,$(a-10,b+4)$; and
2. $(a-12,b-4)$, $(a-16,b-4)$, $(a-20,b-4)$, $(a-24,b-4)$, $(a-28,b-4)$, $(a-30,b-3)$, $(a-26,b-2)$, $(a-22,b-2)$, $(a-18,b-2)$, $(a-14,b-2)$, $(a-10,b)$, $(a-10,b+2)$, $(a-10,b+4)$.

We will call the intervals above the noncovering solution for $\alpha_{\leftarrow}(v)$; as before, optimality can be argued by noting that each interval covers precisely two segments. Observe that, as before, this solution does not cover the endpoints of any v-w link points. Furthermore, the segment below $(a-8,b+6)$ (the only segment in $\alpha_{\uparrow}(v) \cup \alpha_{\rightarrow}(v)$ which may be covered together with a segment from $\alpha_{\leftarrow}(v)$) is not (and cannot be) covered by the noncovering solution. For completeness, we list the covering solution for $\alpha_{\leftarrow}(v)$ below—note that the covering solution for $\alpha_{\downarrow}(v)$ already covers the first segment in $\alpha_{\leftarrow}(v)$, and the covering solution for $\alpha_{\leftarrow}(v)$ will also cover the first segment in $\alpha_{\uparrow}(v)$:

1. $(a-10,b-3)$,$(a-10,b-1)$,$(a-10,b+1)$,$(a-10,b+3)$,$(a-10,b+5)$;
2. $(a-14,b-4)$, $(a-18,b-4)$, $(a-22,b-4)$, $(a-26,b-4)$, $(a-30,b-4)$; $(a-28,b-2)$, $(a-24,b-2)$, $(a-20,b-2)$, $(a-16,b-2)$, $(a-12,b-1)$, $(a-10,b+1)$, $(a-10,b+3)$, $(a-10,b+5)$.

C. Definition of $\alpha_{\uparrow}(v)$:

1. If v has no incident edge connected from above, then add the segments below $(a-8,b+6)$, $(a-8,b+7)$, $(a-6,b+8)$, $(a-4,b+8)$, $\ldots$, $(a+4,b+8)$, $(a+6,b+7)$, $(a+8,b+6)$.
2. If v has an incident edge connected from above and this edge leads to a vertex w with coordinates (a,b'), add the following segments:
   - the segment below $(a-8,b+6)$;
   - for each integer j such that $b+7 \leq j \leq b'-13$, the segments below $(a-6,j)$;
   - the segments below $(a-8,b'-12)$, $(a-8,b'-11)$, $(a-8,b'-10)$, $(a-6,b'-9)$, $(a-4,b'-9)$, $(a-2,b'-9)$, $(a,b'-9)$, $(a+2,b'-9)$, $(a+4,b'-10)$, and denote the point $(a-3,b'-9)$ as the v-w link point in $\alpha(v)$;
   - for each integer j such that $b'-11 \geq j \geq b+7$, the segment below $(a+6,j)$; and
   - the segment below $(a+8,b+6)$.

If the noncovering solution was used for $\alpha_{\leftarrow}(v)$, then the segment below $(a-8,b+6)$ is not covered yet. Such a noncovering solution may be extended towards $\alpha_{\uparrow}(v)$ by adding the following intervals. Note that, as before, no extension of the same size and with the same properties can cover any of the v-w link points, and that the extension does not cover the "adjacent" segment in $\alpha_{\rightarrow}(v)$, which will be the segment below $(a+10,b+5)$:

1. $(a-8,b+6)$, $(a-6,b+7)$, $(a-2,b+7)$, $(a+2,b+7)$, $(a+6,b+6)$; and
2. $(a-8,b+6)$, $(a-6,b+8)$, $(a-6,b+10)$, $\ldots$, $(a-6,b'-14)$, $(a-8,b'-12)$, $(a-8,b'-10)$, $(a-6,b'-9)$, $(a-2,b'-9)$, $(a+2,b'-10)$, $(a+6,b'-12)$, $(a+6,b'-14)$, $\ldots$, $(a+6,b+8)$, $(a+6,b+6)$.

As before, we also list the covering solution for $\alpha_{\uparrow}(v)$ below. Once again, the covering solution for $\alpha_{\downarrow}(v)$ already covers the first segment in $\alpha_{\uparrow}(v)$, and the covering solution for $\alpha_{\leftarrow}(v)$ will also cover the first segment in $\alpha_{\rightarrow}(v)$:

1. $(a-8,b+7)$, $(a-4,b+7)$, $(a,b+7)$, $(a+4,b+7)$, $(a+8,b+5)$; and
2. $(a-6,b+7)$, $(a-6,b+9)$,$\ldots$, $(a-6,b'-13)$, $(a-8,b'-11)$, $(a-8,b'-9)$, $(a-4,b'-9)$, $(a,b'-9)$, $(a+4,b'-11)$, $(a+6,b'-13)$, $\ldots$, $(a+6,b+9)$, $(a+6,b+7)$, $(a+8,b+5)$.

D. Definition of $\alpha_{\rightarrow}(v)$:

1. If v has no incident edge connected from the right, then add a segment below $(a+10, b+5)$, $(a+10, b+4)$, $\ldots$, $(a+10, b-1)$, $(a+12, b-2)$, $(a+12, b-3)$, $(a+12, b-4)$.
2. If v has an incident edge connected from the right and this edge leads to a vertex w with coordinates (a', b), then add the following segments:
   - the segment below $(a+10, b+5)$;
   - for each $i \in \{a+12, \ldots, a'-22\}$, the segment below $(i, b+4)$;
   - the segment below $(a'-20, b+3)$;
   - for each $i \in \{a'-22, a'-24, \ldots, a+14, a+12\}$, the segment below $(i, b+2)$, and furthermore, mark $(a'-25, b+1)$ as the v-w link point in $\alpha(v)$; and
   - the segments below $(a+10, b+1)$, $(a+10, b)$, $(a+10, b-1)$, $(a+12, b-2)$, $(a+12, b-3)$, $(a+12, b-4)$.

If the noncovering solution was used for the three arms so far, then the segments below $(a+10, b+5)$ and above $(a+10, b-4)$ are not covered yet. Such a noncovering solution may be extended towards $\alpha_{\rightarrow}(v)$ by adding the following intervals.

1. $(a+10, b+4)$, $(a+10, b+2)$, $(a+10, b)$, $(a+10, b-2)$, $(a+10, b-4)$; and
2. $(a+10, b+4)$, $(a+14, b+4)$, $\ldots$, $(a'-26, b+4)$, $(a'-22, b+3)$, $(a'-24, b+2)$, $(a'-28, b+2)$, $\ldots$, $(a+16, b+2)$, $(a+12, b+2)$, $(a+10, b)$, $(a+10, b-2)$, $(a+10, b-4)$.

As before, we also list the covering solution for $\alpha_{\rightarrow}(v)$ below. Here, note that the covering solutions for $\alpha_{\uparrow}(v)$ and $\alpha_{\downarrow}(v)$ have already covered the segments below $(a+10, b+5)$ and $(a+10, b-4)$.

1. $(a+10, b+3)$, $(a+10, b+1)$, $(a+10, b-1)$, $(a+10, b-3)$; and
2. $(a+12, b+4)$, $(a+16, b+4)$, $\ldots$, $(a'-24, b+4)$, $(a'-22, b+2)$, $(a'-26, b+1)$, $(a'-30, b+1)$, $\ldots$, $(a+14, b+1)$, $(a+10, b+1)$, $(a+10, b-1)$, $(a+10, b-3)$.

This completes the construction of the modular vertex gadget. Property 1 required by Lemma 4.4 follows from the construction, while properties 2 and 3 follow from the provided configurations and the observations made concerning the possibility of covering at most two segments by a single interval (along with the fact that covering a link point in addition to the marker segment requires the use of an additional interval). For property 4, observe that the v-w link points in $\alpha(v)$ and $\alpha(w)$ always form corners of a 2×3 subgrid, as required. ∎

After applying Lemma 4.4 to construct the vertex gadgets, we will construct, for each edge vw, an edge gadget $\beta(vw)$. Let (x_v, y_v) and (x_w, y_w) be the two link points in $\alpha(v)$ and $\alpha(w)$, respectively, for the edge vw such that $x_w = x_v + 2$ and $y_w = y_v - 3$. The edge gadget $\beta(vw)$ consists of the following segments: $[(x_v, y_v), (x_v, y_v - 1)]$, $[(x_v + 1, y_v - 1), (x_v + 1, y_v - 2)]$, and $[(x_v + 2, y_v - 2), (x_v + 2, y_v - 3)]$. We now proceed to the NP-hardness proof.

THEOREM 4.5. GRID SEGMENT COVERING *and* EXACT GRID SEGMENT COVERING *are* NP*-complete.*

Proof. Inclusion in NP is trivial. To show NP-hardness, we reduce from PLANAR 3-VERTEX COVER restricted to graphs each of which is given with a plane orthogonal bendless drawing on an $n \times n$ grid, where n is the number of vertices in the graph; this restriction remains NP-hard by Theorem 4.3. Given an instance (G, ℓ) of this problem, we refine the grid provided on the input by a factor of 100, replace all vertices with vertex gadgets as per Lemma 4.4, and replace all edges with edge-gadgets. Set $k = \ell + |E(G)| + \sum_{v \in V(G)} \mathsf{cost}(\alpha(v))$. Now, consider the instance (Σ, k) of (EXACT) GRID SEGMENT COVERING.

First, we claim that if (G, ℓ) is a YES-instance, then so is (Σ, k). Indeed, assume there exists a vertex cover X of cardinality at most ℓ in G. Consider now the set of intervals obtained as follows:

1. For each $x \in X$, the set of intervals that form a covering solution for $\alpha(x)$.
2. For each $v \notin X$, the set of intervals that form a noncovering solution for $\alpha(x)$.
3. Since X was a vertex cover, each edge gadget $\beta(vw)$ must have at least one of its segments covered by a v-w link point. This means that the remaining one or two segments can always be covered by 1 additional interval. Note that this interval can be placed in a way which covers only the segments in $\beta(vw)$ that remain uncovered. More formally, let (x_v, y_v) and (x_w, y_w) be the two link points for the edge vw such that $x_w = x_v + 2$ and $y_w = y_v - 3$. If both of the segments $[(x_v, y_v), (x_v, y_v - 1)]$ and $[(x_v + 2, y_v - 2), (x_v + 2, y_v - 3)]$ are covered, then we can cover the segment $[(x_v + 1, y_v - 1), (x_v + 1, y_v - 2)]$ with interval $[(x_v + 1, y_v - 1), (x_v + 3, y_v - 1)]$. If only $[(x_v + 2, y_v - 2), (x_v + 2, y_v - 3)]$ is covered then the segments $[(x_v, y_v), (x_v, y_v - 1)]$ and $[(x_v + 1, y_v - 1), (x_v + 1, y_v - 2)]$ can be covered by interval $[(x_v, y_v - 1), (x_v + 2, y_v - 1)]$. Finally, the segments $[(x_v + 1, y_v - 1), (x_v + 1, y_v - 2)]$ and $[(x_v + 2, y_v - 2), (x_v + 2, y_v - 3)]$ can be covered by interval $[(x_v + 1, y_v - 2), (x_v + 3, y_v - 2)]$.

The total number of intervals used in items 1 and 2 above is upper-bounded by $\ell + \sum_{v \in V(G)} \mathsf{cost}(\alpha(v))$, while the number of intervals used in point 3 is precisely $|E(G)|$. Since all segments are now covered, the claim holds.

To conclude the proof, it suffices to argue that if (Σ, k) is a YES-instance, then so is (G, ℓ). To this end, consider a set of intervals I of cardinality at most k which cover all segments in Σ. The first task will be to normalize I. Initially, for each edge-gadget $\beta(vw)$ that is covered by at least two intervals such that neither of these intervals interacts with a vertex gadget, move an arbitrary one of these intervals so as to cover not only $\beta(vw)$ but also the v-w link point of either $\alpha(v)$ or $\alpha(w)$.

Next, for each $\alpha(v)$, let $I[\alpha(v)]$ be the set of intervals that intersect with at least one endpoint of a segment in $\alpha(v)$.

1. If $I[\alpha(v)]$ does not intersect an endpoint of any segment in an edge-gadget (i.e., it does not intersect any link points), then replace $I[\alpha(v)]$ with a noncovering solution for $\alpha(v)$. Lemma 4.4 guarantees that the set of intervals remains a solution after this replacement.
2. If $I[\alpha(v)]$ intersects at least one endpoint of a segment of an edge-gadget, then it follows from the construction that $|I[\alpha(v)]| \geq \mathsf{cost}(\alpha(v)) + 1$. Indeed, as argued in Lemma 4.4, it is not possible to use $\mathsf{cost}(\alpha(v))$ intervals to cover all the nonmarker segments, the unique marker segment, and a link point in $\alpha(v)$ (the marker segment forces a parity that is opposite to that used by the link points). Hence, we may replace $I[\alpha(v)]$ by the covering solution for $\alpha(v)$, and the set of intervals will remain a valid solution after this replacement.

Now to obtain a solution for (G, ℓ), it suffices to consider the vertex cover X containing all vertices v, where $I[\alpha(v)]$ represents a covering solution. Note that by the construction of k and the fact that each edge gadget requires the use of at least 1 interval that does not interact with the vertex gadgets at all, it follows that the total number of segments in I must be at least $|E| + \sum_{v \in V(G)} \mathsf{cost}(\alpha(v))$ plus the number of vertex gadgets where a covering solution is used. Hence, we have $|X| \leq \ell$. Finally, note that for each edge-gadget, at least one of its link points must be covered by an interval, and hence each edge in G must be incident to at least one vertex in X. ∎

4.2. Reductions from the Grid Segment Covering problem.

THEOREM 4.6. DISC-EQUAL-SC, CONT-EQUAL-SC, DISC-EQUAL-EXACT-SC, CONT-EQUAL-EXACT-SC *are all* NP*-complete.*

Proof. Given k intervals, it is easy to verify that all segments are covered and hence all four problems are in NP. Towards showing NP-hardness, first note that the NP-hardness of DISC-EQUAL-SC and DISC-EQUAL-EXACT-SC follow directly from the NP-hardness of CONT-EQUAL-SC and CONT-EQUAL-EXACT-SC, respectively, by Remark 2.1. To show that CONT-EQUAL-SC and CONT-EQUAL-EXACT-SC are NP-hard, we give a reduction from GRID SEGMENT COVERING and EXACT GRID SEGMENT COVERING, respectively. Both reductions are actually identical. Let (Γ, k) be an instance of GRID SEGMENT COVERING (EXACT GRID SEGMENT COVERING) with endpoints on a $q \times q$ grid. We construct an instance (Γ', k') of CONT-EQUAL-SC (CONT-EQUAL-EXACT-SC) as follows. For a segment $I \in \Gamma$ with endpoints (w, h) and $(w, h+1)$, we add in Γ' the segment $[\frac{(q+3)h+w}{2}, \frac{(q+3)(h+1)+w}{2}]$ and we let $k' = k$. Note that if a segment $I' \in \Gamma'$ has one endpoint at $b \in \mathbb{Q}$, then there exist $h \in \mathbb{N}$ and $w \in \mathbb{Q}$ (with $0 \leq w \leq q$) such that $b = \frac{(q+3)h+w}{2}$. Hence, if a unit interval covers endpoints $b_1 = \frac{(q+3)h_1+w_1}{2}$ and $b_2 = \frac{(q+3)h_2+w_2}{2}$, then it follows that $h_1 = h_2$, because otherwise $|b_2 - b_1| \geq \frac{3}{2}$.

We claim that (Γ', k') is a YES-instance of CONT-EQUAL-SC (CONT-EQUAL-EXACT-SC) if and only if (Γ, k) is a YES-instance of GRID SEGMENT COVERING (EXACT GRID SEGMENT COVERING).

First, let $\mathcal{S}'$ be a set of intervals covering Γ'. We construct $\mathcal{S}$ as follows. Let $[a, a+1] \in \mathcal{S}'$. Note that we can write a as $\frac{(q+3)h+w}{2}$ such that $h \in \mathbb{N}$ and $0 \leq w < q+3$. Furthermore, since there is no endpoint of a segment in Γ' between $\frac{(q+3)h+q}{2}$ and $\frac{(q+3)(h+1)}{2}$, we can assume that $w \leq q-2$ (if it is not the case, we can replace $[a, a+1]$ by either $[\frac{(q+3)h+q-2}{2}, \frac{(q+3)h+q}{2}]$ or $[\frac{(q+3)(h+1)}{2}, \frac{(q+3)(h+1)+2}{2}]$). Finally, we add the horizontal interval with endpoints (w, h) and $(w+2, h)$ into $\mathcal{S}$. Note that, for every segment $I \in \Gamma$ with endpoints (w, h) and $(w, h+1)$, there is a segment $I' = [\frac{(q+3)h+w}{2}, \frac{(q+3)(h+1)+w}{2}]$ in Γ' and an interval $[a, a+1] \in \mathcal{S}'$ that covers I'. Now $a = \frac{(q+3)\bar{h}+\bar{w}}{2}$ for $\bar{h} \in \mathbb{N}$ and $0 \leq \bar{w} \leq q-2$, and $\mathcal{S}$ contains the interval with endpoints $(\bar{w}, \bar{h})$ and $(\bar{w}+2, \bar{h})$. Since $[a, a+1]$ covers the interval $[\frac{(q+3)h+w}{2}, \frac{(q+3)(h+1)+w}{2}]$, it follows that $\bar{h} \in \{h, h+1\}$ and $\bar{w} \leq w \leq \bar{w}+2$, and hence I is covered by an interval in $\mathcal{S}$. Furthermore, observe that if exactly one endpoint of I' is covered, then also exactly one endpoint of I is covered.

On the other hand, let $\mathcal{S}$ be a set of horizontal intervals covering Γ. We create $\mathcal{S}'$ as follows. For every interval $S \in \mathcal{S}$ with endpoints (w, h) and $(w+2, h)$, we add to $\mathcal{S}'$ the interval $S' = [\frac{(q+3)h+w}{2}, \frac{(q+3)h+w}{2} + 1]$. It is easy to see that if $S \in \mathcal{S}$ covers the segment $I \in \Gamma$ with endpoints $(\bar{w}, \bar{h})$ and $(\bar{w}, \bar{h}+1)$, then $h \in \{\bar{h}, \bar{h}+1\}$ and $w \leq \bar{w} \leq w+2$ and hence S' covers the interval $I' = [\frac{(q+3)\bar{h}+\bar{w}}{2}, \frac{(q+3)(\bar{h}+1)+\bar{w}}{2}]$ obtained from I by the reduction. Finally, it is again easy to see that if exactly one endpoint of I was covered by an interval in $\mathcal{S}$, then also exactly one endpoint of I' is covered by an interval in $\mathcal{S}'$. ∎

5. FPT algorithms for the equal segment-length variants. In this section, we give FPT algorithms for CONT-EQUAL-SC, CONT-EQUAL-EXACT-SC, DISC-EQUAL-SC, and DISC-EQUAL-EXACT-SC. Before proceeding to the technical details, we first discuss and give an overview of the FPT algorithm for CONT-EQUAL-SC.

Let (Γ, k) be an instance of Cont-Equal-SC. The FPT algorithm starts by computing an approximate solution, $\mathcal{I}_{apx}$, for Γ of size at most $3k$ (assuming that a solution of size k exists) whose intervals contain all endpoints of the segments in Γ. The algorithm then guesses (i.e., branches on all possibilities) how $\mathcal{I}_{apx}$ interacts with a solution, $\mathcal{I}_{opt}$, for Γ of size at most k. Based on this guess, the algorithm identifies endpoints of segments in $\mathcal{I}_{apx}$, called anchors, around which the intervals in $\mathcal{I}_{opt}$ are anchored (i.e., placed). The goal then becomes to assign the endpoints of the segments in Γ to the guessed anchors, where assigning an endpoint to an anchor means that the endpoint (and hence the associated segment) is covered by the interval in $\mathcal{I}_{opt}$ placed around that anchor, and then modeling the problem as an instance of 2-Sat that stipulates that, for each segment, at least one of its endpoints is covered by a unit interval placed around an anchor. The issue, however, is that for a segment, there could be four anchors whose intervals cover its (two) endpoints, and this cannot be stipulated by size-2 clauses. This issue is resolved by exploiting the crucial property that all segments have the same length, which enables us to define a notion of domination among the anchors that could potentially cover the same segment, and guess this domination. Once the domination relations among the anchors affecting each segment are revealed, encoding the covering requirement using size-2 clauses becomes possible, as the number of anchors affecting each segment can be reduced from 4 to 2. Extra clauses are then added to the 2-Sat instance to enforce that the assignment corresponds to a proper placement of k unit-length covering intervals that cover the segments in Γ. We proceed to the details.

Let (Γ, k) be an instance of Cont-Equal-SC. We start by showing that, in polynomial time, we can compute a solution to Γ that is within ratio 3 from an optimal solution that contains all segment endpoints. Even though this approximation result is worse than the ratio 2 approximation result in [4], the approximation algorithm we use is different and guarantees the crucial property that the solution contains all segment endpoints, which is needed for the FPT algorithm. First, we introduce the following notion.

Let I be a unit-length interval. We define the *left dual* (resp., *right dual*) of I, denoted, $\overline{I_L}$ (resp., $\overline{I_R}$), to be the interval that is the translation of I (along the horizontal line) to the left (resp., to the right) by a vector whose length is equal to the length of the segments in Γ. Observe that the set of segments in Γ whose right (resp., left) endpoints are covered by a unit-length interval I is the same set of segments whose left (resp., right) endpoints are covered by $\overline{I_L}$ (resp., $\overline{I_R}$).

The approximation algorithm alluded to above, denoted APX-ALGO, finds a set of unit-length intervals of minimum cardinality that covers the endpoints of all the segments in Γ; the problem of covering a set of N points on a line by the minimum number of unit-length intervals is known to be solvable in $\mathcal{O}(N \lg N)$ time by a greedy approach (e.g., see problem 16.2-5 in [11]).

Fact 5.1. *APX-ALGO has approximation ratio* 3.

Proof. Consider an optimal solution, and for each interval I in the optimal solution, add both its left and right duals $\overline{I_L}$ and $\overline{I_R}$. We obtain a solution that contains all segment endpoints and whose cardinality is at most thrice that of the optimal solution. Since APX-ALGO produces an optimal solution for covering the endpoints of all segments in Γ, the result follows. ∎

Let $\mathcal{I}_{apx}$ be the set of unit intervals obtained by applying APX-ALGO to the instance Γ.

Based on the above, if $|\mathcal{I}_{apx}| > 3k$, the instance (Γ, k) is a no-instance of Cont-Equal-SC. Assume henceforth that $|\mathcal{I}_{apx}| \leq 3k$. Every endpoint of a segment in Γ is contained in an interval of $\mathcal{I}_{apx}$. Without loss of generality, we can assume that the intervals in $\mathcal{I}_{apx}$ are pairwise disjoint, and that each starts at an endpoint of a segment in Γ (see also Remark 2.1). If not, we can process the intervals in $\mathcal{I}_{apx}$ to ensure this property, while maintaining the property that every endpoint of a segment in Γ is contained in an interval of $\mathcal{I}_{apx}$. To do so, we repeatedly pick the leftmost two overlapping intervals in $\mathcal{I}_{apx}$, say I_r, I_s where I_r starts before I_s. We shift I_s to the right until it no longer overlaps with I_r, while starting at an endpoint in Γ and retaining the set of segment endpoints contained in $I_r \cup I_s$. If at some point this shifting results in an interval that is devoid of segment endpoints, we remove this interval from $\mathcal{I}_{apx}$. Clearly, at the end of this process, the set $\mathcal{I}_{apx}$ consists of pairwise-disjoint intervals that contain all endpoints of the segments in Γ, each of whose intervals starts at an endpoint of a segment in Γ, and satisfying $|\mathcal{I}_{apx}| \leq 3k$.

Let $\mathcal{I}_{opt}$ be an optimal solution for Γ, and assume that $|\mathcal{I}_{opt}| \leq k$. Without loss of generality, we will assume that $\mathcal{I}_{opt}$ is chosen so as to maximize the number of intervals that are common to both $\mathcal{I}_{opt}$ and $\mathcal{I}_{apx}$ (i.e., maximize $|\mathcal{I}_{opt} \cap \mathcal{I}_{apx}|$).

Definition 5.2. *An endpoint $\mathfrak{a}$ of an interval in $\mathcal{I}_{apx}$ is called an anchor if there is an interval I in $\mathcal{I}_{opt}$ that contains $\mathfrak{a}$, in which case we say that I* induces *$\mathfrak{a}$.*

The FPT-algorithm for Cont-Equal-SC performs the following steps:

Step (1). *Guessing the anchors*: The FPT-algorithm starts by guessing how $\mathcal{I}_{opt}$ interacts with $\mathcal{I}_{apx}$. First, it guesses the number k' of intervals that are common to both $\mathcal{I}_{apx}$ and $\mathcal{I}_{opt}$ (i.e., $|\mathcal{I}_{apx} \cap \mathcal{I}_{opt}|$). Then, the algorithm guesses the k' common intervals, removes them from $\mathcal{I}_{apx}$, and updates Γ and the parameter k accordingly (by removing from Γ all segments covered by the k' intervals, and setting $k = k - k'$). By the same arguments made above about $\mathcal{I}_{apx}$, we can assume from now on that the intervals in $\mathcal{I}_{opt}$ are disjoint, and that each starts at a segment endpoint. Every interval $I \in \mathcal{I}_{opt}$ intersects two consecutive intervals in $\mathcal{I}_{apx}$. Otherwise, if I intersects only one interval $I' \in \mathcal{I}_{apx}$, since each interval in $\mathcal{I}_{opt}$ starts at a segment endpoint, and the intervals in $\mathcal{I}_{apx}$ cover all segment endpoints, the intersection of I and I' would include the left endpoint of I, and all segment endpoints in I must also be in I'. This contradicts our choice of $\mathcal{I}_{opt}$ as an optimal solution that maximizes $|\mathcal{I}_{apx} \cap \mathcal{I}_{opt}|$ (since I could be shifted left to obtain an optimal solution containing I'). Hence, if I contains the left (resp., right) endpoint of I', then it must contain the right (resp., left) endpoint of the predecessor (resp., successor) interval of I' in $\mathcal{I}_{apx}$. Next, for each endpoint of an interval in $\mathcal{I}_{apx}$, the algorithm guesses whether it is an anchor. Let Υ be the set of guessed anchors.

Step (2). *Restructuring the anchors*: From Step (1), if an anchor $\mathfrak{a} \in \Upsilon$ is the right (resp., left) endpoint of an interval $I' \in \mathcal{I}_{apx}$, then the interval $I \in \mathcal{I}_{opt}$ that induces $\mathfrak{a}$ intersects the successor (resp., predecessor) of I' in $\mathcal{I}_{apx}$, and hence the left (resp., right) endpoint of the successor (resp., predecessor) of I' must be an anchor induced by I as well. If after Step (1) Υ does not conform to the above, then we can reject the guess, as there will be another guess that satisfies the above property. Based on this property, we will remove from Υ the anchors that are right endpoints of intervals in $\mathcal{I}_{apx}$. After removing these anchors, each interval in $\mathcal{I}_{opt}$ induces exactly one anchor in Υ that is the left endpoint of an interval in $\mathcal{I}_{apx}$. Since the intervals in $\mathcal{I}_{apx}$ are pairwise disjoint, any two anchors in Υ are more than 1-unit apart. Consequently, if $|\Upsilon| > k$, then we can reject the guess in Step (1), as no solution of size at most k realizing the guess exists.

Step (3). *Domination among anchors*: Let $\mathfrak{a}, \mathfrak{b}$ be two anchors, and let $\mathcal{S} \subseteq \Gamma$. We say that $\mathfrak{a}$ *dominates* $\mathfrak{b}$ w.r.t. $\mathcal{S}$, written as $\mathfrak{a} \succeq_{\mathcal{S}} \mathfrak{b}$ or $\mathfrak{b} \preceq_{\mathcal{S}} \mathfrak{a}$, if the set of segments in $\mathcal{S}$ covered by (the interval in $\mathcal{I}_{opt}$ inducing) $\mathfrak{a}$ is a superset of the set of segments in $\mathcal{S}$ covered by (the interval in $\mathcal{I}_{opt}$ inducing) $\mathfrak{b}$. For convenience, we will define the notion of an *empty anchor*, denoted $\otimes$; the set of segments covered by the empty anchor is the empty set ϕ, and hence every anchor dominates $\otimes$. We will use the notion of domination, in conjunction with a guessing process, to reduce the instance (Γ, k), resulting from Steps (1) and (2) above, to an instance of 2-SAT, which can then be solved in polynomial time. To do so, we consider the intervals in $\mathcal{I}_{apx}$ from left to right, and construct an instance $\mathcal{F}$ of 2-SAT. We initialize $\mathcal{F}$ to the empty set, and we will add clauses to $\mathcal{F}$ as follows.

Let $I \in \mathcal{I}_{apx}$ be the interval currently under consideration (when scanning the intervals in $\mathcal{I}_{apx}$ from left to right), and let $\mathcal{S}$ be the set of segments whose left endpoints are on I. (We are not concerned at this point about the set of segments whose right endpoint are on I since those have been considered earlier in the process.) Observe that, since all segments in Γ have the same length, and all covering intervals have the same length, the right endpoints of the segments in $\mathcal{S}$ fall either on one or on two intervals in $\mathcal{I}_{apx}$. (Otherwise, there would be two segments whose left endpoints lie on I, and hence of distance at most 1 unit, but whose right endpoints are more than one unit apart.) This property, which again stems from the fact that all segments in Γ have the same length, is *crucial*, and is the key idea behind the FPT algorithm, as will be seen below.

We treat the more complex case in which the right endpoints of the segments in $\mathcal{S}$ fall on two intervals I_1, I_2 in $\mathcal{I}_{apx}$, where I_2 is the successor of I_1 in $\mathcal{I}_{apx}$; the case where they fall on one interval is simpler, and is a subcase of the treated case, as we explain below. Partition $\mathcal{S}$ into $\mathcal{S}_1, \mathcal{S}_2$, where $\mathcal{S}_1$ consists of those segments in $\mathcal{S}$ whose right endpoints are on I_1, and $\mathcal{S}_2$ of those whose right endpoints are on I_2. Let $\mathfrak{a}$ be the left endpoint of I if it is an anchor, and $\mathfrak{a} = \otimes$ otherwise; let $\mathfrak{b}$ be the left endpoint of the successor of I in $\mathcal{I}_{apx}$ if its an anchor, and $\mathfrak{b} = \otimes$ otherwise; let $\mathfrak{u}$ be the left endpoint of I_1 if it is an anchor, and $\mathfrak{u} = \otimes$ otherwise; let $\mathfrak{s}$ be the left endpoint of I_2 if it is an anchor, and $\mathfrak{s} = \otimes$ otherwise; and let $\mathfrak{t}$ be the left endpoint of the successor of I_2 in $\mathcal{I}_{apx}$ if it is an anchor, and $\mathfrak{t} = \otimes$ otherwise. Observe that, since each of the anchors $\mathfrak{a}$ and $\mathfrak{u}$ covers a prefix (possibly empty) of the sequence of segments in $\mathcal{S}_1$ when ordered from left to right, one of the two anchors must dominate the other w.r.t. $\mathcal{S}_1$. Similarly, each of $\mathfrak{b}$ and $\mathfrak{s}$ covers a suffix (possibly empty) of the sequence of segments in $\mathcal{S}_1$, and hence one must dominate the other w.r.t. $\mathcal{S}_1$. W.r.t. $\mathcal{S}_2$, one of $\mathfrak{a}$ and $\mathfrak{s}$ must dominate the other, and one of $\mathfrak{b}$ and $\mathfrak{t}$ must dominate the other. Therefore, (i) either $\mathfrak{a} \preceq_{\mathcal{S}_1} \mathfrak{u}$ or $\mathfrak{u} \preceq_{\mathcal{S}_1} \mathfrak{a}$ and (ii) either $\mathfrak{b} \preceq_{\mathcal{S}_1} \mathfrak{s}$ or $\mathfrak{s} \preceq_{\mathcal{S}_1} \mathfrak{b}$; and w.r.t. the segments in $\mathcal{S}_2$, we have (iii) either $\mathfrak{a} \preceq_{\mathcal{S}_2} \mathfrak{s}$ or $\mathfrak{s} \preceq_{\mathcal{S}_2} \mathfrak{a}$, and (iv) either $\mathfrak{b} \preceq_{\mathcal{S}_2} \mathfrak{t}$ or $\mathfrak{t} \preceq_{\mathcal{S}_2} \mathfrak{b}$.

The algorithm now guesses, for each of cases (i)–(iv) above, which anchor dominates the other. This guessing results in four cases w.r.t. each of $\mathcal{S}_1$ and $\mathcal{S}_2$ that are described below, and hence results in sixteen cases overall. If the right endpoints of the segments in $\mathcal{S}$ fall on one interval I_1, then the guessing results only in the four cases w.r.t. $\mathcal{S}_1$ distinguished below (and anchor $\mathfrak{t}$ would not be needed). We will create Boolean variables corresponding to endpoints of segments in Γ. A Boolean variable of the form $x_{\mathfrak{h}}$, where x is an endpoint of a segment $S \in \Gamma$ and $\mathfrak{h}$ is an anchor, is true if and only if point x (and hence S) is covered by the interval inducing $\mathfrak{h}$. We will then form an instance of 2-SAT that encodes the instance (Γ, k) of CONT-EQUAL-SC under the assumed guess. For simplicity of the presentation, we make the following

FIG. 7. *Illustration for Cases* 1-4 *w.r.t.* $\mathcal{S}_1$ *and* $\mathcal{S}_2$. *The two red circles designate the endpoints* x, y *of a segment* $S \in \mathcal{S}_1$, *and the green circles designate the endpoints* x', y' *of a segment* $S' \in \mathcal{S}_2$, *where* x *is to the left of* y *and* x' *is to the left of* y'. *For clarity, only the endpoints of* S, S' *are shown. If the guess w.r.t.* $\mathcal{S}_1$ *is that* $\mathfrak{a} \succeq \mathfrak{u}$ *and* $\mathfrak{s} \succeq \mathfrak{b}$ *and w.r.t.* $\mathcal{S}_2$ *is that* $\mathfrak{s} \succeq \mathfrak{a}$ *and* $\mathfrak{b} \succeq \mathfrak{t}$ *then clauses* $\{x_{\mathfrak{a}} \vee y_{\mathfrak{s}}\}$ *and* $\{x'_{\mathfrak{b}} \vee y'_{\mathfrak{s}}\}$ *are added to* $\mathcal{F}$. *(*Color available online.*)*

assumptions. If a Boolean variable $x_{\mathfrak{h}}$, associated with anchor $\mathfrak{h}$, is such that either $\mathfrak{h} = \otimes$, or the distance between x and $\mathfrak{h}$ is more than 1 unit (i.e., more than the length of a unit-length covering interval), then we set/fix the value of $x_{\mathfrak{h}}$ to FALSE. We start by associating with every endpoint x of a segment in Γ two Boolean variables as follows. Let $\mathfrak{h}$ and $\mathfrak{h}'$ be the two anchors directly to the left and right, respectively, of x. We associate the two Boolean variables $x_{\mathfrak{h}}$ and $x_{\mathfrak{h}'}$ with x. (Note that $\mathfrak{h}$ or $\mathfrak{h}'$ could be $\otimes$, or of distance more than 1 unit from x, and in which case the corresponding Boolean variable would be set to FALSE.) The four cases distinguished w.r.t. $\mathcal{S}_1$ are (see Figure 7 for an illustration) the following:

Case 1: $\mathfrak{a} \succeq \mathfrak{u}$ and $\mathfrak{b} \succeq \mathfrak{s}$. For each endpoint x on I such that x is the left endpoint of a segment in $\mathcal{S}_1$, add the clause $\{x_{\mathfrak{a}} \vee x_{\mathfrak{b}}\}$ to $\mathcal{F}$.

Case 2: $\mathfrak{a} \succeq \mathfrak{u}$ and $\mathfrak{s} \succeq \mathfrak{b}$. For each endpoint x on I such that x is the left endpoint of a segment S in $\mathcal{S}_1$, let y be the right endpoint of S. Add the clause $\{x_{\mathfrak{a}} \vee y_{\mathfrak{s}}\}$ to $\mathcal{F}$.

Case 3: $\mathfrak{u} \succeq \mathfrak{a}$ and $\mathfrak{b} \succeq \mathfrak{s}$. For each endpoint x on I such that x is the left endpoint of a segment S in $\mathcal{S}_1$, let y be the right endpoint of S. Add the clause $\{x_{\mathfrak{b}} \vee y_{\mathfrak{u}}\}$ to $\mathcal{F}$.

Case 4: $\mathfrak{u} \succeq \mathfrak{a}$ and $\mathfrak{s} \succeq \mathfrak{b}$. For each endpoint x on I such that x is the left endpoint of a segment S in $\mathcal{S}_1$, let y be the right endpoint of S. Add the clause $\{y_{\mathfrak{u}} \vee y_{\mathfrak{s}}\}$ to $\mathcal{F}$.

The four cases distinguished w.r.t. $\mathcal{S}_2$ are the following:

Case 1: $\mathfrak{a} \succeq \mathfrak{s}$ and $\mathfrak{b} \succeq \mathfrak{t}$. For each endpoint x on I such that x is the left endpoint of a segment in $\mathcal{S}_2$, add the clause $\{x_{\mathfrak{a}} \vee x_{\mathfrak{b}}\}$ to $\mathcal{F}$.

Case 2: $\mathfrak{a} \succeq \mathfrak{s}$ and $\mathfrak{t} \succeq \mathfrak{b}$. For each endpoint x on I such that x is the left endpoint of a segment S in $\mathcal{S}_2$, let y be the right endpoint of S. Add the clause $\{x_{\mathfrak{a}} \vee y_{\mathfrak{t}}\}$ to $\mathcal{F}$.

Case 3: $\mathfrak{s} \succeq \mathfrak{a}$ and $\mathfrak{b} \succeq \mathfrak{t}$. For each endpoint x on I such that x is the left endpoint of a segment S in $\mathcal{S}_2$, let y be the right endpoint of S. Add the clause $\{x_{\mathfrak{b}} \vee y_{\mathfrak{s}}\}$ to $\mathcal{F}$.

Case 4: $\mathfrak{s} \succeq \mathfrak{a}$ and $\mathfrak{t} \succeq \mathfrak{b}$. For each endpoint x on I such that x is the left endpoint of a segment in $\mathcal{S}_2$, let y be the right endpoint of S. Add the clause $\{y_{\mathfrak{s}} \vee y_{\mathfrak{t}}\}$ to $\mathcal{F}$.

After processing the intervals in $\mathcal{I}_{apx}$, guessing domination, associating Boolean variables, and adding clauses to $\mathcal{F}$, we add to $\mathcal{F}$ the following "enforcement" clauses. For every anchor $\mathfrak{a}$, and every two variables $x_{\mathfrak{a}}, z_{\mathfrak{a}}$, corresponding to segment endpoints x, z, respectively, associated with $\mathfrak{a}$:

(E1) If x and z are on the right (resp., left) side of $\mathfrak{a}$, and if z (resp., x) is to the right (resp., left) of x, add $\{\overline{z}_{\mathfrak{a}} \vee x_{\mathfrak{a}}\}$ (resp., $\{\overline{x}_{\mathfrak{a}} \vee z_{\mathfrak{a}}\}$) to $\mathcal{F}$; otherwise, add $\{\overline{x}_a \vee z_a\}$ (resp., $\{\overline{z}_{\mathfrak{a}} \vee x_{\mathfrak{a}}\}$) to $\mathcal{F}$.

(E2) If x and z are on opposite sides of $\mathfrak{a}$ and the distance between them is more than 1 unit, add $\{\overline{x}_{\mathfrak{a}} \vee \overline{z}_{\mathfrak{a}}\}$ to $\mathcal{F}$.

The FPT algorithm accepts if any of the guesses it makes leads to a formula $\mathcal{F}$ that is a YES-instance of 2-SAT, and rejects otherwise.

THEOREM 5.3. CONT-EQUAL-SC *can be solved in time* $\mathcal{O}((2^4 \cdot 3 \cdot e^{5/3})^k \cdot n \log n) = \mathcal{O}(2^{8k} \cdot n \log n)$, *where* $n = |\Gamma|$, *and hence is* FPT.

Proof. We first argue the correctness of the algorithm. The instance (Γ, k) is a YES-instance of CONT-EQUAL-SC if and only if there exists an optimal solution $\mathcal{I}_{opt}$ for Γ containing at most k intervals. The algorithm guesses in Step (1) the intervals in $\mathcal{I}_{apx}$ that are in $\mathcal{I}_{opt}$, and updates (Γ, k) accordingly. As explained before, we may assume that the intervals in the optimal solution sought (if it exists), $\mathcal{I}_{opt}$, are pairwise disjoint, start at segment endpoints, and that each interval in $\mathcal{I}_{opt}$ intersects two consecutive intervals in $\mathcal{I}_{apx}$. The algorithm then guesses which endpoints of intervals in $\mathcal{I}_{apx}$ are anchors (w.r.t. $\mathcal{I}_{opt}$).

The algorithm in Step (2) removes anchors from the set Υ of anchors, so that each anchor is the left endpoint of its interval in $\mathcal{I}_{apx}$. Note that after this restructuring, each interval in the sought solution $\mathcal{I}_{opt}$ contains exactly one anchor in Υ. Moreover, any two anchors are more than 1 unit apart, and hence, if $|\Upsilon| > k$, then the algorithm can safely reject the current guess, as it does in Step (2), since no solution $\mathcal{I}_{opt}$ of size k conforming to the current guess exists. It is clear that a solution $\mathcal{I}_{opt}$ to (Γ, k) exists if and only if there exists a guess of a set Υ of anchors satisfying the above conditions.

The algorithm then proceeds to determining how the intervals in the solution sought should be "anchored" (or placed) around their anchors in order to cover all segments in Γ. To do so, the algorithm considers the intervals in $\mathcal{I}_{apx}$ from left to right. For an interval I under consideration, the set of segments $\mathcal{S}$ whose left endpoints lie on I must be covered by $\mathcal{I}_{opt}$. The right endpoints of the segments in $\mathcal{S}$ lie on at most two intervals of $\mathcal{I}_{apx}$; we argue the more complicated case in which these endpoints lie on exactly two intervals, I_1, I_2, where I_2 is the successor of I_1 in $\mathcal{I}_{apx}$, as this case subsumes the other one. The set of segments $\mathcal{S}$ can be partitioned into $\mathcal{S}_1$ and $\mathcal{S}_2$, as explained in Step (3) of the algorithm, depending on which interval in I_1, I_2 the right endpoint of the segment in $\mathcal{S}$ lies on. Let the anchors $\mathfrak{a}, \mathfrak{b}, \mathfrak{u}, \mathfrak{s}, \mathfrak{t}$ be as defined in Step (3) of the algorithm. Observe that, since each of the anchors $\mathfrak{a}$ and $\mathfrak{u}$ covers a prefix (possibly empty) of the sequence of segments in $\mathcal{S}_1$ when ordered from left to right, one of the two anchors must dominate the other w.r.t. $\mathcal{S}_1$. Similarly, each of the anchors $\mathfrak{b}$ and $\mathfrak{s}$ covers a suffix (possibly empty) of the sequence of segments in $\mathcal{S}_1$ (when ordered from left to right), and hence one must dominate the other w.r.t. $\mathcal{S}_1$. W.r.t. $\mathcal{S}_2$, one of the two anchors $\mathfrak{a}$ and $\mathfrak{s}$ must dominate the other, and one of $\mathfrak{b}$ and $\mathfrak{t}$ must dominate the other. The algorithm guesses each of these domination relations, for each interval $I \in \mathcal{I}_{apx}$ and set of segments $\mathcal{S}$ whose left endpoints lie on I. If the algorithm guesses correctly, then for each segment in Γ, it assigns each of its two endpoints to an anchor such that the segment is covered by an interval inducing one of the anchors assigned to its endpoints. After guessing the domination relations among the anchors, the algorithm creates an instance $\mathcal{F}$ of 2-SAT that for each segment $S \in \Gamma$ and each endpoint of $\mathcal{S}$, corresponds a Boolean variable whose value is true if and only if the endpoint of the segment is covered by the interval inducing the anchor assigned to the endpoint (based on the domination relation). The algorithm then adds enforcement clauses to $\mathcal{F}$ ensuring (the converse) that a satisfying assignment to $\mathcal{F}$ corresponds to an assignment of segment endpoints to anchors satisfying (1) all endpoints assigned to the same anchor can be covered by a unit interval (E2); (2) if an endpoint x of a segment that is assigned to an anchor

$\mathfrak{h}$ is covered by the interval $I_{\mathfrak{h}}$ inducing $\mathfrak{h}$, then any segment endpoint assigned to $\mathfrak{h}$ that is between x and $\mathfrak{h}$ is also covered by $I_{\mathfrak{h}}$ (E1).

Given the above, it is not difficult to verify that the instance (Γ, k) is a YES-instance of Disc-Equal-SC if and only if there is a guess for the algorithm that yields a YES-instance $\mathcal{F}$ of 2-Sat, and hence that the algorithm is correct. More specifically, by Step (2), each interval in $\mathcal{I}_{opt}$ induces exactly one anchor. Therefore, if we have a solution $\mathcal{I}_{opt}$ to (Γ, k), then we can determine a satisfying assignment to $\mathcal{F}$. Conversely, if $\mathcal{F}$ is satisfiable, then from a satisfying assignment to $\mathcal{F}$, we can determine—through the anchors—a solution to the instance (Γ, k). Next we analyze the running time of the algorithm.

First, observe that computing $\mathcal{I}_{apx}$ can be done in $\mathcal{O}(n \lg n)$ time, as this can be done by sorting the endpoints in Γ. Moreover, all processing steps for the intervals in $\mathcal{I}_{apx}$ and segments Γ can be carried out within the time upper bound $\mathcal{O}(n \lg n)$. Therefore, we only need to analyze the size of the search tree needed to simulate the guesses performed by the algorithm. The algorithm performs guessing only in Steps (1) and (3).

In Step (1), the algorithm guesses a number $k' \in \{0, \ldots, k\}$, and then it guesses a subset of k' intervals in $\mathcal{I}_{apx}$. The total number of branches needed to simulate these guesses is at most $\sum_{k'=0}^{k} \binom{3k}{k'}$. For each guess of k' intervals, the algorithm removes these intervals from $\mathcal{I}_{apx}$, updates Γ, and sets $k = k - k'$. Removing k' intervals from $\mathcal{I}_{apx}$ leaves $\mathcal{I}_{apx}$ with at most $3k - k'$ intervals. The algorithm then guesses which endpoints of the (remaining) intervals in $\mathcal{I}_{apx}$ are anchors. Since—as explained in Step (2)—we can assume that the anchors in question are left endpoints of their intervals, guessing the anchors can be simulated by choosing a subset of $(k - k')$ endpoints, out of the at most $(3k - k')$ left endpoints of the (remaining) intervals in $\mathcal{I}_{apx}$. Therefore, the total number of branches needed to simulate all guesses in Step (1) is at most $\sum_{k'=0}^{k} \binom{3k}{k'} \cdot \binom{3k-k'}{k-k'}$.

In Step (3), the algorithm guesses the domination relations among anchors. At this point, the number of anchors (by Step (2)) is at most $k - k'$. In the guessing, each guess made is w.r.t. an interval $I \in \mathcal{I}_{apx}$ and the two anchors $\mathfrak{a}$ and $\mathfrak{b}$, as defined in Step (3). We will charge each guess to the two anchors that play the roles of $\mathfrak{a}$ and $\mathfrak{b}$ w.r.t. some interval $I \in \mathcal{I}_{apx}$. There are four guesses made, resulting in sixteen cases, and each of $\mathfrak{a}$ and $\mathfrak{b}$ is involved in the same number of guesses. Therefore, eight cases need to be distinguished w.r.t. each of $\mathfrak{a}$ and $\mathfrak{b}$. Since each of the at most $k - k'$ anchors can play the role of $\mathfrak{a}$ once and of $\mathfrak{b}$ once, over all intervals in $\mathcal{I}_{apx}$ (note that a domination relation involving an empty anchor is determined, not guessed), it follows that the total number of cases that each anchor can be involved in is sixteen, which results in a total number of branches of at most $2^{4(k-k')}$ over all anchors.

It follows that the size of the search tree needed to simulate all the guesses performed by the algorithm is $\sum_{k'=0}^{k} \binom{3k}{k'} \cdot \binom{3k-k'}{k-k'} \cdot 2^{4(k-k')}$. Next, we upper bound this expression.

Applying the well-known upper bound $\binom{r}{s} \leq (e \cdot r/s)^s$ on the binomial coefficient in both binomial terms $\binom{3k}{k'}$ and $\binom{3k-k'}{k-k'}$, where e is the base of the natural logarithm, and simplifying, we obtain

$$\sum_{k'=0}^{k} \binom{3k}{k'} \cdot \binom{3k-k'}{k-k'} \cdot 2^{4(k-k')} \tag{5.1}$$

$$\leq \binom{3k}{k} + (3e)^k \cdot 2^{4k} + (3e)^k \cdot \sum_{k'=1}^{k-1} 2^{4(k-k')} \cdot (k/k')^{k'} \cdot \Big(\frac{(k-k'/3)}{(k-k')}\Big)^{k-k'} \tag{5.2}$$

$$\leq (3e)^k + (3e)^k \cdot 2^{4k} + (3e)^k \cdot \sum_{k'=1}^{k-1} 2^{4k} \cdot (k/k')^{k'} \cdot e^{2k'/3} \tag{5.3}$$

$$= (3e)^k + (3e)^k \cdot 2^{4k} + (3e)^k \cdot 2^{4k} \cdot \sum_{k'=1}^{k-1} ((e^{2/3} \cdot k)/k')^{k'} \tag{5.4}$$

$$\leq (3e)^k + (3e)^k \cdot 2^{4k} + (3e)^k \cdot 2^{4k} \cdot \mathcal{O}((e^{2/3})^k) = \mathcal{O}((2^4 \cdot 3 \cdot e^{5/3})^k). \tag{5.5}$$

In inequality (5.2), we split the summation—a minor technicality—in order to avoid a denominator of 0 in the terms $(k/k')^{k'}$ and $((k-k'/3)/(k-k'))^{k-k'}$, resulting from approximating $\binom{3k}{k'}$ and $\binom{3k-k'}{k-k'}$, when $k' = 0$ and when $k' = k$, respectively. We then obtain inequality (5.3) from inequality (5.2), by upper bounding the term $((k-k'/3)/(k-k'))^{k-k'}$ by $e^{2k'/3}$. This is done by rewriting the term $((k-k'/3)/(k-k'))^{k-k'}$ in the form $(1+1/x)^x$, and using the well-known inequality $(1+1/x)^x \leq e$ for all $x > 0$. We obtain inequality (5.5) from inequality (5.4) by showing that the function $((e^{2/3} \cdot k)/k')^{k'}$ is increasing in k', which then can be used to upper bound the summation $\sum_{k'=1}^{k}((e^{2/3} \cdot k)/k')^{k'}$ by $\mathcal{O}((3 \cdot e^{2/3})^k)$. ∎

Next, we discuss how to obtain FPT algorithms for Disc-Equal-SC, Disc-Equal-Exact-SC, and Cont-Equal-Exact-SC. We focus on Disc-Equal-SC, as the FPT algorithms for Disc-Equal-Exact-SC and Cont-Equal-Exact-SC are byproducts of that for Disc-Equal-SC. The ideas leading to the FPT algorithm for Disc-Equal-SC are the same as those for Cont-Equal-SC, but their execution becomes more complicated, and the upper bound obtained on the running time of the algorithm is much worse. The complications are mainly due to the stipulation that the covering intervals cannot be arbitrarily chosen, and must be selected from the set $\mathcal{I}$, given as input. This leads to more complicated arguments for obtaining an approximate solution satisfying the desired properties (and to a worse approximation ratio), and for restructuring the anchors. In particular, we no longer can make the simplifying assumptions about how the intervals are structured in both $\mathcal{I}_{apx}$ and $\mathcal{I}_{opt}$.

We start by discussing how to obtain an approximate solution that contains all segment endpoints. The idea behind this greedy approximation algorithm we present here was actually presented in [4] as a first step for obtaining an approximation algorithm of ratio 3; however, the approximate solution of ratio 3 computed in [4] does not necessarily contain all segment endpoints (assuming that each segment endpoint is contained in some interval in $\mathcal{I}$). We explain below how to modify the algorithm in [4] to obtain an approximate solution with the desired property, at the expense of worsening the approximation ratio.

We will assume for now that each segment endpoint is contained in some interval in $\mathcal{I}$. We will discuss later how to lift this assumption. (Basically, if a segment endpoint is not contained in an interval in $\mathcal{I}$, then the other endpoint of the segment *must* be covered by some interval in $\mathcal{I}$, which can be easily enforced after the anchors have been guessed and we form $\mathcal{F}$.) The approximation algorithm, denoted $\mathbb{A}_{apx}$, sorts all segment endpoints from left to right, and considers them in the sorted order. In each step, $\mathbb{A}_{apx}$ considers the first (w.r.t. the sorted order) point, p, that is not covered yet by any interval in the partial solution computed so far. Let I_p be an interval in $\mathcal{I}$ containing p, such that, among all intervals containing p in $\mathcal{I}$, I_p extends the most to the right; we refer to p as the *defining point* of I_p. The algorithm $\mathbb{A}_{apx}$ adds I_p to the partial solution, and repeats until all segment endpoints are covered by intervals in the solution. Let $\mathcal{I}_{apx}$ be the solution computed by $\mathbb{A}_{apx}$, and let

$\mathcal{I}_{opt}$ be an optimal solution. Before we can upper bound the approximation ratio of $\mathbb{A}_{apx}$, we need to establish—in the next lemma—a few structural properties of the intervals in $\mathcal{I}_{apx}$. By a *span* we mean a stretch on the line that is not necessarily an interval/segment in $\Gamma \cup \mathcal{I}$.

LEMMA 5.4. *The following properties of the intervals in $\mathcal{I}_{apx}$ hold:*

(i) *For any interval $I \in \mathcal{I}$, the number of intervals in $\mathcal{I}_{apx}$ that contain segment endpoints that lie on I is at most 3.*

(ii) *For any span of length at most 1 (unit), the number of intervals in $\mathcal{I}_{apx}$ that contain segment endpoints that lie on the span is at most 4.*

Proof. We first state the following easy fact about $\mathbb{A}_{apx}$ that we rely on for proving both statements above.

FACT 5.5. *Let S be a span/interval of length at most 1, and let I_p be an interval chosen by $\mathbb{A}_{apx}$ whose defining point p lies to the right of (the right endpoint of) S. Then no interval chosen (by $\mathbb{A}_{apx}$) after I_p intersects S.*

The above fact is true since any interval chosen after I_p cannot contain p; otherwise, it would cover p and extend to the right more than I_p, which would contradict the choice of I_p.

Proof of (i). Order the intervals in $\mathcal{I}_{apx}$ intersecting I from left to right w.r.t. their right endpoints, and note that this order is the same as the order in which $\mathbb{A}_{apx}$ selects these intervals. Let I_1 be the first interval w.r.t. this order, and let p_1 be the rightmost segment endpoint in $I_1 \cap I$. If I_2 exists, by the choice of I_2, it must extend to the right at least as much as I. Therefore, I_2 must cover all segment endpoints lying on I, and hence the defining point of I_3 (if it exists) must be to the right of I. The statement now follows from Fact 5.5.

Proof of (ii). The proof is similar to that of (i), with the only difference being that the span may not be an interval in $\mathcal{I}$. Again, order the intervals containing segment endpoints that lie on the span, S, from left to right; let this order be $I_1, I_2, \ldots, I_r$. Let p_1 be the rightmost segment endpoint in $I_1 \cap S$. If I_2 covers the right point of S, then the point defining I_3 is to the right of S, and $r \leq 3$ by Fact 5.5. Assume now that the right point of I_2 is on S. If the right point of I_3 is on S, then all segment endpoints on S that are covered by I_2 are also covered by I_3; this would contradict the choice of I_2 (given that its defining point is after p_1 on S). Therefore, the right point of I_3 is to the right of S, and hence the defining point of I_4 (if it exists) is to the right of S. It follows that $r \leq 4$ by Fact 5.5. ∎

LEMMA 5.6. *$|\mathcal{I}_{apx}| \leq 11|\mathcal{I}_{opt}|$, and hence the approximation ratio of $\mathbb{A}_{apx}$ is 11.*

Proof. For an endpoint p of a segment $S \in \Gamma$, define its *dual* to be the other endpoint of S.

Partition the intervals in $\mathcal{I}_{apx}$ into two sets, $\mathcal{I}^{\cap}_{apx}$ and $\overline{\mathcal{I}^{\cap}_{apx}}$, where $\mathcal{I}^{\cap}_{apx}$ consists of those intervals that contain points from intervals in $\mathcal{I}_{opt}$, and $\overline{\mathcal{I}^{\cap}_{apx}} = \mathcal{I}_{apx} \setminus \mathcal{I}^{\cap}_{apx}$. By part (i) of Lemma 5.4, for each interval $I \in \mathcal{I}_{opt}$, the number of intervals in $\mathcal{I}^{\cap}_{apx}$ containing points from I is at most three. It follows that $|\mathcal{I}^{\cap}_{apx}| \leq 3|\mathcal{I}_{opt}|$.

To upper bound $|\overline{\mathcal{I}^{\cap}_{apx}}|$, observe that for each interval $I' \in \overline{\mathcal{I}^{\cap}_{apx}}$, and for each point $p \in I'$, its dual must lie on some interval in $\mathcal{I}_{opt}$. Therefore, we can equivalently upper bound the number of intervals in $\mathcal{I}_{apx}$ containing points that are the duals of those lying on intervals in $\mathcal{I}_{opt}$. Let I be an interval in $\mathcal{I}_{opt}$. The duals of the points

that lie on I are contained in at most two spans S_1, S_2, where S_1 contains the left endpoints of the segments whose right endpoints lie on I, and S_2 contains the right endpoints of the segments whose left endpoints lie on I. By part (ii) of Lemma 5.4, the number of intervals in $\mathcal{I}_{apx}$, and hence in $\overline{\mathcal{I}^{\cap}_{apx}}$, containing segment endpoints on S_1 is at most four and the number of intervals containing segment endpoints on S_2 is at most four. Therefore, $|\overline{\mathcal{I}^{\cap}_{apx}}| \leq 8|\mathcal{I}_{opt}|$, and the lemma follows. ∎

THEOREM 5.7. DISC-EQUAL-SC *can be solved by an algorithm running in time* $\mathcal{O}(2^{30k} \cdot (n_\Gamma \lg n_\Gamma + n_\mathcal{I} \lg n_\mathcal{I}))$, *where* $n_\Gamma = |\Gamma|$ *and* $n_\mathcal{I} = |\mathcal{I}|$, *and hence is* FPT.

Proof. Let $(\Gamma, \mathcal{I}, k)$ be an instance of DISC-EQUAL-SC. The algorithm, $\mathbb{A}_{\mathsf{FPT}}$, starts by computing an approximate solution $\mathcal{I}_{apx}$ for $(\Gamma, \mathcal{I}, k)$, using the algorithm $\mathbb{A}_{apx}$. By Lemma 5.6, if $|\mathcal{I}_{apx}| > 11k$, then $(\Gamma, \mathcal{I}, k)$ is a no-instance of DISC-EQUAL-SC, and $\mathbb{A}_{\mathsf{FPT}}$ safely rejects. Assume, henceforth, that $|\mathcal{I}_{apx}| \leq 11k$. Consequently, the number of interval endpoints in $\mathcal{I}_{apx}$ is at most $22k$.

Next, $\mathbb{A}_{\mathsf{FPT}}$ guesses the set Υ of anchors as follows. For each endpoint p of an interval $\mathcal{I}_{apx}$, $\mathbb{A}_{\mathsf{FPT}}$ guesses whether or not $\mathfrak{p}$ is an anchor induced by an interval $I \in \mathcal{I}_{opt}$ such that no anchor induced by I is already in Υ; if the guess is that $\mathfrak{p}$ is such an anchor, then $\mathfrak{p}$ is added to Υ. Simulating the guessing of Υ can be done using a search tree of size at most 2^{22k}.

If $|\Upsilon| > k$, then the algorithm rejects the guess. Otherwise, the set Υ can be used to define a subdivision of the intervals in $\mathcal{I}_{apx}$ into maximal subintervals w.r.t. the property that each is a subinterval of some interval in $\mathcal{I}_{apx}$ and contains no anchors in its interior (i.e., as an interior point); we call each such subinterval an *anchor subinterval*. For each anchor subinterval I, let $\mathfrak{a}$ be the (possibly empty) anchor closest to the left endpoint of I from the left side (possibly the left endpoint of I itself if it is an anchor), and $\mathfrak{b}$ (possibly empty) that closest to the right endpoint of I from its right (possibly the right endpoint of I itself if it is an anchor). For each segment endpoint x on I, we create two Boolean variables $x_\mathfrak{a}, x_\mathfrak{b}$, associated with $\mathfrak{a}$ and $\mathfrak{b}$, respectively. If an anchor $\mathfrak{h} = \otimes$, then any Boolean variable $x_\mathfrak{h}$, associated with $\mathfrak{h}$, is set to FALSE. Moreover, if for a Boolean variable $x_\mathfrak{h}$, there is no interval $I \in \mathcal{I}$ that contains both points x and $\mathfrak{h}$, then we set $x_\mathfrak{h}$ to FALSE.

The algorithm then considers these anchor subintervals from left to right. For an anchor interval I, with associated anchors $\mathfrak{a}$ (left) and $\mathfrak{b}$ (right), since I is a subinterval of some interval in $\mathcal{I}_{apx}$, the right endpoints of the segments whose left endpoints are on I lie on a span S of length at most 1. By part (ii) of Lemma 5.4, the number of intervals in $\mathcal{I}_{apx}$ that contain endpoints on S is at most four. It is not difficult to verify that the endpoints on S are contained in at most $r \leq 4$ anchor subintervals $I_1, \ldots, I_r$. For each interval I_i, $i \in [r]$, let $\mathfrak{a}_i$ be the anchor to its left and $\mathfrak{b}_i$ that to its right (these anchors could be empty). As in the proof of Theorem 5.3, for each $i \in [r]$, we guess the domination relation between $\mathfrak{a}$ and $\mathfrak{a}_i$, and that between $\mathfrak{b}$ and $\mathfrak{b}_i$, and add 2-clauses involving the Boolean variables corresponding to the segments whose left endpoints are on I and right endpoints on I_i, that stipulate how these segments are covered in accordance with the guessed domination relations. Each anchor $\mathfrak{a}$ can be involved in at most four domination relations as left anchor, and at most four as right anchor. Therefore, the size of the search tree needed to simulate all the domination relations guesses is $(2^4 \cdot 2^4)^k = 2^{8k}$.

If an endpoint x of a segment S is not on any interval in $\mathcal{I}_{apx}$, then the other endpoint, y, of S must be on an interval in $\mathcal{I}$, and hence in $\mathcal{I}_{apx}$ (otherwise, the instance can be rejected). Let $\mathfrak{a}, \mathfrak{b}$ be the two anchors associated with the anchor interval containing y. We add to $\mathcal{F}$ the clause $\{\overline{y}_\mathfrak{a} \vee \overline{y}_\mathfrak{b}\}$.

Finally, we need to add enforcement clauses. This is a bit different from the way we do it in Cont-Equal-SC, again, due to the fact that the covering intervals must be chosen from $\mathcal{I}$, and cannot be arbitrarily placed. The enforcement clauses added in (E1) remain the same. However, for (E2), we need to additionally check the existence of an interval in $\mathcal{I}$ containing the two endpoints, and hence (E2) is changed as follows.

For every anchor $\mathfrak{a}$, and every two variables $x_\mathfrak{a}, z_\mathfrak{a}$, corresponding to endpoints x, z, respectively, associated with $\mathfrak{a}$, if x and z are on opposite sides of $\mathfrak{a}$ and there is no interval in $\mathcal{I}$ that contains both x and z add $\{\overline{x}_\mathfrak{a} \vee \overline{z}_\mathfrak{a}\}$ to the 2-Sat formula $\mathcal{F}$.

By the above discussions, the size of the search tree needed to simulate the guesses performed by $\mathbb{A}_{\mathsf{FPT}}$ is $2^{22k} \cdot 2^{8k} = 2^{30k}$. All the remaining steps can be performed in time $\mathcal{O}(n_\Gamma \lg n_\Gamma + n_\mathcal{I} \lg n_\mathcal{I})$ by sorting the intervals in each of Γ and $\mathcal{I}$. ∎

Corollary 5.8. Disc-Equal-Exact-SC *and* Cont-Equal-Exact-SC *can be solved in time* $\mathcal{O}(2^{30k} \cdot (n_\Gamma \lg n_\Gamma + n_\mathcal{I} \lg n_\mathcal{I}))$, *where* $n_\Gamma = |\Gamma|$ *and* $n_\mathcal{I} = |\mathcal{I}|$, *and hence are* FPT.

Proof. The algorithm for Disc-Equal-Exact-SC is the same as that for Disc-Equal-SC up to the point where the instance $\mathcal{F}$ of 2-Sat is constructed. (Note that $\mathcal{I}_{apx}$ may not be a valid approximate solution for the instance of Disc-Equal-Exact-SC, but all that it matters is that it covers all segment endpoints, and that its cardinality is within a factor of 11 from an optimal solution for the instance of Disc-Equal-Exact-SC. This is true since a solution to Disc-Equal-Exact-SC is also a solution to Disc-Equal-SC. We will stipulate that the covering must be exact by adding 2-clauses to $\mathcal{F}$.) The only difference between the two algorithms is that, after constructing $\mathcal{F}$, we add to $\mathcal{F}$ additional enforcement clauses stipulating that no segment in Γ has both endpoints covered by an interval in the solution. This can be done as follows. For each segment $S \in \Gamma$, let x and y be its left and right endpoints, respectively. For each Boolean variable $x_\mathfrak{h}$ associated with x, where $\mathfrak{h}$ is an anchor, and for each Boolean variable $y_{\mathfrak{h}'}$ associated with y, where $\mathfrak{h}'$ is an anchor, add the clause $\{\overline{x_\mathfrak{h}} \vee \overline{y_{\mathfrak{h}'}}\}$ to $\mathcal{F}$.

For Cont-Equal-Exact-SC, the result follows from the polynomial-time FPT-reduction to Disc-Equal-Exact-SC (see Remark 2.1). ∎

6. Conclusion. In this paper, we considered several variants of segment covering by unit intervals. We established the NP-hardness of the restrictions of these problems to instances in which all segments have the same length. In addition to its importance per se, this result strengthens and implies a number of NP-hardness results in the literature. We also presented parameterized complexity results for several of these problems, showing their W[1]-hardness for the general case, and presenting FPT algorithms for their restrictions to instances in which all segments have the same length. Our work gives rise to two open questions:

1. What is the parameterized complexity of Cont-Exact-SC and Disc-Exact-SC?
2. What is the complexity of the restriction of Cont-Exact-SC and Disc-Exact-SC to instances in which all segments have length at most 1 unit?

REFERENCES

[1] A. Acharyya, S. Nandy, S. Pandit, and S. Roy, *Covering segments with unit squares*, Comput. Geom. Theory Appl., 79 (2019), pp. 1–13.

[2] E. Arkin, A. Banik, P. Carmi, G. Citovsky, M. Katz, J. Mitchell, and M. Simakov, *Choice is hard*, in ISAAC, Lecture Notes in Comput. Sci. 9472, Springer, Berlin, 2015, pp. 318–328.

[3] E. Arkin, A. Banik, P. Carmi, G. Citovsky, M. Katz, J. Mitchell, and M. Simakov, *Conflict-free covering*, in CCCG, Elsevier, Amsterdam, 2015, 28.
[4] E. Arkin, A. Banik, P. Carmi, G. Citovsky, M. Katz, J. Mitchell, and M. Simakov, *Selecting and covering colored points*, Discrete Appl. Math., 250 (2018), pp. 75–86.
[5] P. Bonsma, T. Epping, and W. Hochstättler, *Complexity results on restricted instances of a paint shop problem for words*, Discrete Appl. Math., 154 (2006), pp. 1335–1343.
[6] K. Bringmann, D. Hermelin, M. Mnich, and E. J. van Leeuwen, *Parameterized complexity dichotomy for Steiner multicut*, J. Comput. System Sci., 82 (2016), pp. 1020–1043.
[7] Y. Chen, J. Flum, and M. Grohe, *Machine-based methods in parameterized complexity theory*, Theoret. Comput. Sci., 339 (2005), pp. 167–199.
[8] M. Chlebík and J. Chlebíková, *The complexity of combinatorial optimization problems on d-dimensional boxes*, SIAM J. Discrete Math., 21 (2007), pp. 158–169.
[9] B. N. Clark, C. J. Colbourn, and D. S. Johnson, *Unit disk graphs*, Discrete Math., 86 (1990), pp. 165–177.
[10] M. Claverol, E. Khramtcova, E. Papadopoulou, M. Saumell, and C. Seara, *Stabbing circles for sets of segments in the plane*, Algorithmica, 80 (2018), pp. 849–884.
[11] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed., MIT Press, Cambridge, MA, 2009.
[12] M. Cygan, F. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer, Cham, Switzerland, 2015.
[13] R. Diestel, *Graph Theory, 4th Edition*, Springer, Heidelberg, 2012.
[14] R. Downey and M. Fellows, *Fundamentals of Parameterized Complexity*, Texts Comput. Sci., Springer, Heidelberg, 2013.
[15] T. Erlebach and E. J. van Leeuwen, *Approximating geometric coverage problems*, in SODA, SIAM, Philadelphia, 2008, pp. 1267–1276.
[16] T. Erlebach and E. J. van Leeuwen, *PTAS for weighted set cover on unit squares*, in APPROX-RANDOM, Lecture Notes in Comput. Sci. 6302, Springer, Berlin, 2010, pp. 166–177.
[17] M. R. Garey and D. S. Johnson, *The rectilinear Steiner tree problem in NP complete*, SIAM J. Appl. Math., 32 (1977), pp. 826–834.
[18] A. Gupta, S. Kale, V. Nagarajan, R. Saket, and B. Schieber, *The approximability of the binary paintshop problem*, in APPROX-RANDOM, Lecture Notes in Comput. Sci. 8096, Springer, Berlin, 2013, pp. 205–217.
[19] S. Hartung and R. Niedermeier, *Incremental list coloring of graphs, parameterized by conservation*, Theoret. Comput. Sci., 494 (2013), pp. 86–98.
[20] B. M. P. Jansen, *On structural parameterizations of hitting set: Hitting paths in graphs using 2-sat*, J. Graph Algorithms Appl., 21 (2017), pp. 219–243.
[21] G. Kant, *Drawing planar graphs using the canonical ordering*, Algorithmica, 16 (1996), pp. 4–32.
[22] J. Kleinberg and E. Tardos, *Algorithm Design*, Addison-Wesley Longman, Boston, MA, 2005.
[23] K. Kobylkin, *Stabbing line segments with disks: Complexity and approximation algorithms*, in AIST, vol. 10716 of Lecture Notes in Computer Science, Springer, Cham, Switzerland, 2017, pp. 356–367.
[24] S. Langerman and P. Morin, *Covering things with things*, Discrete Comput. Geom., 33 (2005), pp. 717–729.
[25] R. Madireddy and A. Mudgal, *Stabbing line segments with disks and related problems*, in CCCG, Elsevier, Amsterdam, 2016, pp. 201–207.
[26] J. Wang, W. Li, and J. Chen, *A parameterized algorithm for the hyperplane-cover problem*, Theoret. Comput. Sci., 411 (2010), pp. 4005–4009.