On the Complexity of Minimising the Moving Distance for Dispersing Objects

Nicolás Honorato-Droguett
Nagoya University, Japan

Kazuhiro Kurita
Nagoya University, Japan

Tesshu Hanaka
Kyushu University, Fukuoka, Japan

Hirotaka Ono
Nagoya University, Japan

Abstract

We study Geometric Graph Edit Distance (GGED), a graph-editing model to compute the minimum edit distance of intersection graphs that uses moving objects as an edit operation. We first show an $O(n \log n)$-time algorithm that minimises the total moving distance to disperse unit intervals. This algorithm is applied to render a given unit interval graph (i) edgeless, (ii) acyclic and (iii) k-clique-free. We next show that GGED becomes strongly NP-hard when rendering a weighted interval graph (i) edgeless, (ii) acyclic and (iii) k-clique-free. Lastly, we prove that minimising the maximum moving distance for rendering a unit disk graph edgeless is strongly NP-hard over the L_1 and L_2 distances.

2012 ACM Subject Classification Theory of computation → Mathematical optimization

Keywords and phrases Intersection graphs, Optimisation, Graph modification

Digital Object Identifier 10.4230/LIPIcs.WADS.2025.36

Related Version *Full Version*: `https://doi.org/10.48550/arXiv.2502.12903` [17]

Funding *Kazuhiro Kurita*: This work is partially supported by JSPS KAKENHI Grant Numbers JP21K17812, JP22H03549, and JST ACT-X Grant Number JPMJAX2105.
Tesshu Hanaka: This work is partially supported by JSPS KAKENHI Grant Numbers JP21K17707, JP23H04388, JST CRONOS Grant Number JPMJCS24K2.
Hirotaka Ono: This work is partially supported by JSPS KAKENHI Grant Numbers JP20H05967, JP21K19765, JP22H00513, JST CRONOS Grant Number JPMJCS24K2.

1 Introduction

Graph modification is a fundamental topic to address graph similarity and dissimilarity, where a given graph is deformed by adding or deleting vertices or edges to satisfy a specific non-trivial graph property, while minimising the cost of edit operations. The problem of determining this cost is commonly known as *graph modification problem* (GMP) and has applications in various disciplines, such as computer vision [3], network interdiction [15], and molecular biology [14]. GMPs are often categorised into vertex and edge modification problems, with edit operations restricted to the vertex and edge sets, respectively.

The cost of a single edit operation in a GMP is often determined by the specific application. In theoretical studies, a unit-cost model is often assumed, where each addition or deletion of a vertex or edge has a uniform cost. However, for such models, it is known that determining whether a graph can be modified to obtain a member of a given class is NP-hard for a

19th International Symposium on Algorithms and Data Structures (WADS 2025).
Editors: Pat Morin and Eunjin Oh; Article No. 36; pp. 36:1–36:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

wide range of graphs and classes [2, 12, 19, 25]. These negative bounds of GMPs motivate alternative formulations for graph editing that consider domain-specific constraints and cost measures.

The choice of edit operations and their associated costs is a crucial aspect of GMPs, as different formulations capture different structural properties and computational challenges. Analogous to string similarity analysis, where modifications are based on biologically significant operations such as DNA mutations and repeats [20], graph modification problems should reflect the inherent constraints and structural properties of the graphs being studied. In particular, *geometric intersection graphs* (hereafter intersection graphs) provide a suitable framework for studying GMPs for scenarios where graphs represent spatial relationships (see, e.g., [6, 10, 23]). Given a collection of geometric objects $\mathcal{S}$, an *intersection graph* $G(\mathcal{S})$ is a graph where there is a one-to-one correspondence between the vertex set $V(G(\mathcal{S}))$ and $\mathcal{S}$, and two vertices are adjacent if and only if their corresponding objects intersect. This model includes many well-known graph classes, such as interval graphs and disk graphs. These graphs can be frequently found in real-world applications such as network modelling and bioinformatics [22].

Motivated by this context, this paper investigates GMPs for intersection graphs. In this context, two natural questions arise:

1. Are standard graph edit operations suitable for modifying intersection graphs?
2. How can the geometric properties of objects be exploited to overcome the hardness of GMPs?

To answer these questions, we introduce Geometric Graph Edit Distance, a model for modifying intersection graphs from a geometric perspective.

In the intersection graph model, a natural edit operation is to move the objects in $\mathcal{S}$. We treat this movement as a graph edit operation and focus on minimising the cost required to modify an intersection graph so that the resulting graph is in a specific graph class. The cost is quantified by the total moving distance, which is the sum of the distances by which objects in $\mathcal{S}$ are moved. More precisely, we define the problem as follows:

Geometric Graph Edit Distance

Input: A collection of n geometric objects $\mathcal{S}$.
Output: The minimum total moving distance of the objects in $\mathcal{S}$ so that the resulting intersection graph $G(\mathcal{S})$ is in Π.

We assume that Π is given by an oracle, i.e. we have an algorithm to determine whether the intersection graph $G(\mathcal{S})$ is in Π.

Related work

Numerous GMPs are known to be computationally hard. In the early 1980s, Lewis and Yannakakis [19] showed that vertex-deletion problems are NP-complete for any hereditary graph class. Similarly, many edge modification problems have been shown to be NP-complete, such as transforming a graph into a perfect, chordal, or interval graph [2]. As a result, the past decade has seen a growing interest in addressing these problems from the perspective of parameterised complexity. The recent survey by Crespelle et al. [5] provides a comprehensive overview of this subject (see also [8]).

Although classical GMPs focus on structural modifications of graphs, recent studies have explored models that include geometric constraints. Honorato-Droguett et al. [16] introduced the above geometric approach to graph modification, demonstrating that graphs of certain classes, such as graph completeness and the existence of a k-clique, can be efficiently obtained on interval graphs. Their work highlights how the underlying geometric properties of intersection graphs can be exploited to design appropriate modification models.

In a similar vein, Fomin et al. [10] studied the *disk dispersal* problem, where a set $\mathcal{S}$ of n disks, an integer $k \geq 0$, and a real number $d \geq 0$ are given, and the goal is to determine whether an edgeless disk graph can be realised by moving at most k disks by at most d distance each. They proved that this problem is NP-hard when $d = 2$ and $k = n$ and also FPT when parameterised by $k + d$. Furthermore, they showed that the problem becomes W[1]-hard when parameterised by k when disk movement is restricted to rectilinear directions.

Expanding on this line of research, Fomin et al. [11] conducted a parameterised complexity study of edge modification problems where *scaling* objects is considered as the edit operation. Their results illustrate how alternative edit operations in geometric intersection graphs can impact computational complexity, enabling further study of geometric modification graph models. In particular, their work includes several FPT results to achieve independence, acyclicity and connectivity on disk graphs.

Our work continues these developments by introducing Geometric Graph Edit Distance, a model that considers object movement as an edit operation to modify intersection graphs. Unlike prior studies that focus on vertex and edge modifications or object scaling, our approach explicitly considers movement costs by quantifying the total moving distance required to obtain a graph in a given class. This approach enables the exploration of new algorithmic and complexity-theoretic questions in the context of geometric intersection graphs.

Our contribution

Our results are mainly focused on interval graphs and summarised in Table 1. In this paper, we deal with the following graph classes: $\Pi_{\texttt{edgeless}}$ (edgeless graphs), $\Pi_{\texttt{acyc}}$ (acyclic graphs) and $\overline{\Pi_{k\texttt{-clique}}}$ (k-clique-free graphs).

In [16], the model presented is studied mainly for classes of dense graphs. This inspires the present paper as a subsequent work, where we instead focus on classes for sparse graphs. As two fundamental classes of sparse graphs, we consider edgeless graphs ($\Pi_{\texttt{edgeless}}$) and acyclic graphs ($\Pi_{\texttt{acyc}}$). These classes have also been studied in related work on geometric intersection graphs [10, 11].

Table 1 Summary of our results. In this table, L_1 and L_2 are the Manhattan and Euclidean distances, respectively. The terms IG, UIG and UDG are abbreviations of interval graphs, unit interval graphs and unit disk graphs, respectively.

Problem Type	Graph	Target Graph Class	Metric	Weighted	Complexity
minsum	UIG	$\Pi_{\texttt{edgeless}}$	$L_2 (= L_1)$	No	$O(n \log n)$
	UIG	$\Pi_{\texttt{acyc}}$	$L_2 (= L_1)$	No	$O(n \log n)$
	UIG	$\overline{\Pi_{k\texttt{-clique}}}$	$L_2 (= L_1)$	No	$O(n \log n)$
	IG	$\Pi_{\texttt{edgeless}}$	$L_2 (= L_1)$	Yes	strongly NP-hard
	IG	$\Pi_{\texttt{acyc}}$	$L_2 (= L_1)$	Yes	strongly NP-hard
	IG	$\overline{\Pi_{k\texttt{-clique}}}$	$L_2 (= L_1)$	Yes	strongly NP-hard for any $1 \leq k \leq n$
minimax	UDG	$\Pi_{\texttt{edgeless}}$	L_2, L_1	Yes	strongly NP-hard

As we shall detail, $\Pi_{\texttt{acyc}}$ is contained in $\overline{\Pi_{k\texttt{-clique}}}$ in our context. As a result, one might argue that the distinction of both classes is irrelevant. However, we still consider them distinctively, as forests are a well-known class of graphs. Our analysis highlights the computational complexity of modifying intersection graphs while considering movement-based edit operations, a perspective distinct from prior work that focuses on exclusively modifying the graph structure.

Paper Organisation. Section 2 formally describes the definitions needed to address the above ideas. Section 3 presents the problem Interval Dispersal and shows that it can be solved in $O(n \log n)$ time. Using this algorithm, we establish that Geometric Graph Edit Distance can also be solved in $O(n \log n)$ time for classes $\Pi_{\texttt{edgeless}}$, $\Pi_{\texttt{acyc}}$, and $\overline{\Pi_{k\texttt{-clique}}}$ on unit interval graphs. Section 4 demonstrates that Geometric Graph Edit Distance becomes strongly NP-hard on weighted interval graphs for classes $\Pi_{\texttt{edgeless}}$, $\Pi_{\texttt{acyc}}$, and $\overline{\Pi_{k\texttt{-clique}}}$. Section 5 shows that the minimax version of Geometric Graph Edit Distance is strongly NP-hard on weighted unit disk graphs for $\Pi_{\texttt{edgeless}}$ under both the L_1 and L_2 distance metrics. Section 6 concludes with remarks on our results and potential future directions.

Due to space restrictions, we omit in-depth explanations and all full proofs of statements with a $\star$-mark. The reader is referred to the full version of this paper [17] for these details.

2 Preliminaries

This section provides the main definitions used in the paper, referencing geometry, graph, and convexity terminology from textbooks [1, 4, 7, 24].

Objects. An *interval* I is a line segment on the real line of length $\text{len}(I) \in \mathbb{R}^+$. Intervals are assumed to be open, unless explicitly stated otherwise. An interval such that $\text{len}(I) = 1$ is called *unit interval.* The *left endpoint* $\ell(I)$ of an interval I is the point that satisfies $\ell(I) \le y$ for any $y \in I$. Similarly, the *right endpoint* $r(I)$ of I is the point that satisfies $y \le r(I)$ for any $y \in I$. The *centre* $c(I)$ of I is the point $c(I) = (r(I) - \ell(I))/2$. Throughout the paper, we assume that the indices of a collection of intervals $\mathcal{I} = \{I_1, \ldots, I_n\}$ follow the order given by centres of intervals. That is, $c(I_i) \le c(I_{i+1})$ for all $1 \le i \le n-1$. However, it is not assumed that collections are ordered when given as the input graph. Given a radius $r > 0$ and a point $p \in \mathbb{R}$, a *disk* D centred at p is the set $D = \{x \in \mathbb{R}^2 \mid \|x, p\|_2 \le r\}$. An *open disk* D is a disk without its boundary circle; that is, $D = \{x \in \mathbb{R}^2 \mid \|x, p\|_2 < r\}$. We assume that the disks are open, unless we mention the contrary. A *unit disk* is a disk of radius $r = 1/2$.

Graphs. Throughout the paper, a graph $G = (V, E)$ is assumed to be a simple, finite, and undirected graph with vertex set V and edge set E. An *edgeless graph* is a graph $G = (V, E)$ such that $E = \emptyset$. A *k-clique* of a graph $G = (V, E)$ is a subset $W \subseteq V$ such that $|W| = k$ and for all $u, v \in W$, $u \ne v$, $\{u, v\} \in E$, for $k \le n$. If such W exists in V, we say that G *contains a k-clique.* An *interval graph* is an intersection graph $G(\mathcal{I}) = (V, E)$ where the vertex set $V = \{v_1, \ldots, v_n\}$ corresponds to a collection of intervals $\mathcal{I} = \{I_1, \ldots, I_n\}$ and an edge $\{v_i, v_j\} \in E$ exists if and only if $I_i \cap I_j \ne \emptyset$, for any $1 \le i, j \le n$, $i \ne j$. An interval graph is called *unit interval graph* if $\text{len}(I) = 1$ for all $I \in \mathcal{I}$. Similarly, a *disk graph* is an intersection graph $G(\mathcal{D}) = (V, E)$ where the vertex set $V = \{v_1, \ldots, v_n\}$ corresponds to a disk collection $\mathcal{D} = \{D_1, \ldots, D_n\}$. An edge $(v_i, v_j) \in E$ exists if and only if $D_i \cap D_j \ne \emptyset$, for any $1 \le i, j \le n$, $i \ne j$. A *unit disk graph* is a disk graph in which the collection contains exclusively unit disks. Unless stated otherwise, all intersection graphs are assumed to be *unweighted*. A *weighted intersection graph* assigns a multiplicative weight, called the *distance weight*, to the moving distance function of each object. The formal definition of distance weight appears in later sections when required. An (infinite) set of graphs Π is a *graph class* (or simply a class), and we say that G *is in* Π if $G \in \Pi$. A graph class Π is *non-trivial* if infinitely many graphs belong to Π and infinitely many graphs do not belong to Π. In this paper, we deal with the following non-trivial classes: (i) $\Pi_{\texttt{edgeless}} = \{G : G \text{ is an edgeless graph.}\}$, (ii) $\Pi_{\texttt{acyc}} = \{G : G \text{ is an acyclic graph.}\}$, (iii) $\Pi_{k\texttt{-clique}} = \{G : G \text{ contains a } k\text{-clique.}\}$ and (iv) $\overline{\Pi_{k\texttt{-clique}}} = \{G : G \notin \Pi_{k\texttt{-clique}}\}$.

3 Rendering Unit Interval Graphs Edgeless in $O(n \log n)$ time

We show that a graph in $\Pi_{\texttt{edgeless}}$ can be obtained in $O(n \log n)$ time given a collection of n unit intervals. We start by defining a problem that we call Interval Dispersal and then use the algorithm designed to obtain a graph in $\Pi_{\texttt{edgeless}}$, $\Pi_{\texttt{acyc}}$ and $\overline{\Pi_{k\texttt{-clique}}}$. Interval Dispersal receives as input a collection $\mathcal{I}$ of n intervals and a real $s \geq 1$, and asks for the minimum value of the total moving distance to obtain a collection $\mathcal{I}'$ that satisfies $c(I'_j) - c(I'_i) \geq s$ for each $I'_i, I'_j \in \mathcal{I}'$, $i < j$. When $s = 1$, Interval Dispersal is equivalent to Geometric Graph Edit Distance on unit interval graphs and $\Pi_{\texttt{edgeless}}$. For simplicity, the intervals are assumed to be open. This avoids the need to address infinitesimally small distances required to separate closed intervals. We must first introduce some basic definitions and notation to describe the algorithm. Given a collection of n intervals $\mathcal{I} = \{I_1, \dots, I_n\}$, let $D = (d_1, \dots, d_n)$ be a vector such that d_i is the moving distance applied to I_i. We denote by $\mathcal{I}^D = \{I_1^D \dots, I_n^D\}$ the collection of intervals such that $c(I_i^D) = c(I_i) + d_i$. The set $\mathcal{D}(\mathcal{I}) \subseteq \mathbb{R}^n$ is the set of vectors that describe the moving distance applied to intervals such that the condition of Interval Dispersal is satisfied. In other words, for all $D = (d_1, \dots, d_n) \in \mathcal{D}(\mathcal{I})$, $c(I_j^D) + c(I_i^D) \geq s$ holds for $i < j$. We use $\mathcal{D}^{opt}(\mathcal{I}) \subseteq \mathcal{D}(\mathcal{I})$ to denote the subset of vectors in $\mathcal{D}(\mathcal{I})$ that minimises the total moving distance applied to intervals; i.e. $\mathcal{D}^{opt}(\mathcal{I}) = \{D = (d_1, \dots, d_n) \in \mathcal{D}(\mathcal{I}) \mid \sum_{1 \leq i \leq n} |d_i| = \min_{D' = (d'_1, \dots, d'_n) \in \mathcal{D}(\mathcal{I})} \sum_{1 \leq i \leq n} |d'_i|\}$.

Intuitively, we aim to find a vector $D \in \mathcal{D}^{opt}(\mathcal{I})$ to move each interval so that the distance between each pair of intervals is at least s. Given an arbitrary $D \in \mathcal{D}^{opt}(\mathcal{I})$, the order of $\mathcal{I}^D$ may be different from the order of $\mathcal{I}$. However, it was previously shown [16] that there is always a vector $D \in \mathcal{D}^{opt}(\mathcal{I})$ such that the order of $\mathcal{I}^D$ preserves the order of $\mathcal{I}$. This implies that there always exists an optimal solution of Interval Dispersal for which checking the inequality $(c(I_{i+1}) + d_{i+1}) - (c(I_i) + d_i) \geq s$ for $\leq i \leq n-1$ is sufficient.

We now define the *equispace function*, which moves intervals so that the distance between their centres is exactly s, maintaining the order induced by interval centres.

▶ **Definition 1** (Equispace function). *Let $(\mathcal{I}, s)$ be an instance of Interval Dispersal where $\mathcal{I}$ is a collection of unit intervals. The* equispace function *of $\mathcal{I}$ to a point x is a function $E : \mathcal{I} \times \mathbb{R} \to \mathbb{R}$ defined as:*

$$E(\mathcal{I}, x) = \sum_{i=1}^{n} f_i(x), \quad f_i(x) = |x - c(I_i) - (n-i)s|.$$

The vector that describes the moving distances given by $E(\mathcal{I}, x)$ is defined as $E_x(\mathcal{I}) = (e_1, \dots, e_n) = (\alpha_1 f_1(x), \dots, \alpha_n f_n(x))$ where $\alpha_i = 1$ if $x \geq c(I_i) + (n-i)s$ and $\alpha_i = -1$ otherwise, for $1 \leq i \leq n$. We also denote by $\mathcal{I}^{E_x(\mathcal{I})} = \{I_1^{E_x(\mathcal{I})} \dots, I_n^{E_x(\mathcal{I})}\}$ the collection of intervals where $c(I_i^{E_x(\mathcal{I})}) = c(I_i) + \alpha_i f_i(x)$ for $1 \leq i \leq n$. ⌟

By the above, $E_x(\mathcal{I}) \in \mathcal{D}(\mathcal{I})$ for all $x \in \mathbb{R}$. Moreover, $c(I_{i+1}^{E_x(\mathcal{I})}) - c(I_i^{E_x(\mathcal{I})}) = s$ for all $1 \leq i \leq n-1$. We first prove that for certain collections of intervals, minimising E gives a vector contained in $\mathcal{D}^{opt}(\mathcal{I})$.

▶ **Lemma 2** ($\star$)**.** *The equispace function $E(\mathcal{I}, x)$ is a piecewise-linear convex function.*

We define the *set of breakpoints of* $E(\mathcal{I}, x)$ to be the set $B_\mathcal{I} = \{b_1^\mathcal{I}, \ldots, b_n^\mathcal{I}\} = \{c(I_i) + (n-i)s \mid I_i \in \mathcal{I},\ 1 \leq i \leq n\}$. Given a collection of intervals $\mathcal{I}$, we define the equispace function $E(\mathcal{I}, x)$ as a sequence of linear functions $E_1(\mathcal{I}, x), \ldots, E_{|\mathcal{I}|+1}(\mathcal{I}, x)$. The slope of $E_i(\mathcal{I}, x)$ is less than the slope of $E_j(\mathcal{I}, x)$ for $1 \leq i < j \leq |\mathcal{I}|$. Since the equispace function is convex and piecewise linear, the points that minimise E are located within a range $b_\ell \leq x \leq b_r$, where $b_\ell \leq b_r$ and $b_\ell, b_r \in B_\mathcal{I}$. We prove that b_ℓ and b_r can be easily found.

▶ **Lemma 3** ($\star$)**.** *The minimum value of $E(\mathcal{I}, x)$ is given by the breakpoint $b_{(n+1)/2}^\mathcal{I}$ if n is odd, and by breakpoints $b_{n/2}^\mathcal{I}$ and $b_{(n/2)+1}^\mathcal{I}$ otherwise.*

By Lemma 3, the minimum value of E for an arbitrary collection of intervals $\mathcal{I}$ is given by the median value(s) of $B_\mathcal{I}$. We now show which collections allow minimising E to obtain a vector in $\mathcal{D}^{opt}(\mathcal{I})$, characterised as follows:

▶ **Definition 4** (Optimally Equispaceable Collections)**.** *Given a collection of intervals $\mathcal{I}$, we say that $\mathcal{I}$ is* optimally equispaceable *if there exists a $D \in \mathcal{D}^{opt}(\mathcal{I})$ such that $D = E_{x^*}(\mathcal{I})$ and $x^* \in \arg\min_{x \in \mathbb{R}} E(\mathcal{I}, x)$. Equivalently, $\mathcal{I}$ is optimally equispaceable if $E_{x^*}(\mathcal{I}) \in \mathcal{D}^{opt}(\mathcal{I})$ for all $x^* \in \arg\min_{x \in \mathbb{R}} E(\mathcal{I}, x)$.*

▶ **Lemma 5.** *Let $\mathcal{I} = \{I_1, \ldots, I_n\}$ be a collection of unit intervals such that $c(I_{i+1}) - c(I_i) \leq s$ for $1 \leq i \leq n-1$. Then $\mathcal{I}$ is optimally equispaceable. Moreover, there exists a $D \in \mathcal{D}^{opt}(\mathcal{I})$ such that $c(I_{i+1}^D) - c(I_i^D) = s$ holds for all $1 \leq i \leq n-1$.*

Proof. We only prove the latter, as the existence of D in $\mathcal{D}^{opt}(\mathcal{I})$ directly implies the optimal equispaceability of $\mathcal{I}$. That is, we show that $\mathcal{I}^D$ satisfies $c(I_{i+1}^D) - c(I_i^D) = s$, for $1 \leq i \leq n-1$. By the definition of Interval Dispersal, we have $c(I_{i+1}^D) \geq c(I_i^D)$ and $c(I_{i+1}^D) - c(I_i^D) \geq s$ for $1 \leq i \leq n-1$. Suppose that there exists a pair of intervals I_i and I_{i+1} that satisfies $c(I_{i+1}^D) - c(I_i^D) > s$. Let $s' = c(I_{i+1}^D) - c(I_i^D)$ and $\delta = s' - s$. We show how to obtain a total moving distance D' such that $\sum_{d \in D'} |d| < \sum_{d \in D} |d|$ and $c(I_{i+1}^{D'}) - c(I_i^{D'}) = s$.

We divide the proof into three cases: (i) $d_i \geq 0$, (ii) $d_{i+1} \leq 0$ and (iii) $d_i \leq 0$ and $d_{i+1} \geq 0$. For case (i), it follows that $d_j \geq d_{j-1} \geq 0$ for $i+1 \leq j \leq n$ and $(c(I_{i+1}^D) - \delta) - c(I_i^D) = c(I_{i+1}) + (d_{i+1} - \delta) - (c(I_i) + d_i) = s$ holds. Let $D'(d'_1, \ldots, d'_n) = (d_1, \ldots, d_i, d_{i+1} - \delta, \ldots, d_n - \delta)$. The dispersal condition is satisfied by $\mathcal{I}^{D'}$. Furthermore, since $\delta > 0$, the total moving distance satisfies $\sum_{d \in D'} |d| = \sum_{j=1}^{i} |d_j| + \sum_{j=i+1}^{n} d_j - \delta < \sum_{d \in D} |d|$, which contradicts the optimality of D.

For case (ii), $d_j \leq d_{j+1}$ for $1 \leq j \leq i$ holds, and the argument for case (i) applies analogously for $D' = (d'_1, \ldots, d'_n) = (d_1 + \delta, \ldots, d_i + \delta, d_{i+1}, \ldots, d_n)$.

We only need to prove case (iii). Let $\delta = s' - s$ as in the previous cases. If $\delta \leq d_{i+1}$, then we move the intervals as in the first case. If $\delta \leq -d_i$, then we move intervals as in the second case. In both cases, the same argument applies and the total moving distance contradicts the optimality of D. Thus we assume that $\delta > d_{i+1}, -d_i$ holds. Without loss of generality, we move intervals I_j for $i+1 \leq j \leq n$ by d_{i+1} to the left by $\delta' = d_{i+1}$ and intervals I_j for $1 \leq j \leq i$ to the right by $\delta'' = (c(I_{i+1}^D) - \delta') - c(I_i^D) - s$. Then $(c(I_{i+1}^D) - \delta') - (c(I_i^D) + \delta'') = s$ holds since $d_{i+1} - \delta' = 0$. Let $D' = (d'_1, \ldots, d'_n) = (d_1 + \delta'', \ldots, d_i + \delta'', d_{i+1} - \delta', \ldots, d_n - \delta')$. The inequality $\sum_{d \in D'} |d| = \sum_{j=1}^{i} d_j + \delta'' + \sum_{j=i+1}^{n} d_j - \delta' < \sum_{d \in D} |d|$ holds since $\delta', \delta'' > 0$, which contradicts the optimality of D. Therefore, in an optimal solution, $\mathcal{I}$ must satisfy $c(I_{i+1}) + d_{i+1} - (c(I_i) + d_i) = c(I_{i+1}^D) - c(I_i^D) = s$, for $1 \leq i \leq n-1$. ◀

Let $\mathcal{I} = \{I_1, \dots, I_n\}$ and $\mathcal{J} = \{J_1, \dots, J_m\}$ be two collections of unit intervals and let $x_1, x_2 \in \arg\min_{x\in\mathbb{R}} E(\mathcal{I}, x)$, $x_1 \le x_2$, and $y_1, y_2 \in \arg\min_{x\in\mathbb{R}} E(\mathcal{J}, x)$, $y_1 \le y_2$, be the breakpoints that minimise E for $\mathcal{I}$ and $\mathcal{J}$, respectively. We say that $\mathcal{I}$ *and* $\mathcal{J}$ *intersect when equispaced* when $y_1 \le x_2 + |\mathcal{J}|s$. In other words, $\mathcal{I}$ and $\mathcal{J}$ intersect when equispaced whenever there exist points $x_1 \le x \le x_2$ and $y_1 \le y \le y_2$ such that there exist $I \in \mathcal{I}^{E_x(\mathcal{I})}$ and $I \in \mathcal{J}^{E_y(\mathcal{J})}$ for which $c(J) - c(I) < s$.

▶ **Lemma 6** ($\star$). *Given that $\mathcal{I}\cup\mathcal{J} = \{I_1, \dots, I_n, J_1, \dots, J_m\}$, $\mathcal{I}\cup\mathcal{J}$ is optimally equispaceable if and only if $y_1 \le x_2 + |\mathcal{J}|s$.*

Corollary 7 is directly implied by Lemma 6.

▶ **Corollary 7.** *If $y_1 > x_2 + |\mathcal{J}|s$, then $\mathcal{I} \cup \mathcal{J}$ is not optimally equispaceable. Moreover, the minimum total moving distance for dispersing $\mathcal{I} \cup \mathcal{J}$ is equal to $E(\mathcal{I}, x) + E(\mathcal{J}, y)$ for arbitrary $x_1 \le x \le x_2$ and $y_1 \le y \le y_2$.*

Given a collection $\mathcal{I}$ of n unit intervals, we note that $\mathcal{I}$ can be partitioned into $m \le n$ subcollections $\mathcal{I}_{a_1,b_1}, \dots, \mathcal{I}_{a_m,b_m}$ such that for all $1 \le i \le m$, $c(I_{j+1}) - c(I_j) \le s$ for $a_i \le j \le b_i - 1$. By Lemma 5, each $\mathcal{I}_{a_i,b_i}$ is an optimally equispaceable collection. We use Lemma 6 and prove the statement of Lemma 8.

▶ **Lemma 8** ($\star$). *Let $\mathcal{I} = \{I_1, \dots, I_n\} = \mathcal{I}_{a_1,b_1} \cup \dots \cup \mathcal{I}_{a_m,b_m}$ be a collection of n unit intervals partitioned as above. If there exist integers $\alpha_1, \dots, \alpha_k$ such that $\mathcal{I}_{a_{\alpha_i},b_{\alpha_i}}$ and $\mathcal{I}_{a_{\alpha_i+1},b_{\alpha_i+1}}$ intersect when equispaced, then there exists an optimal solution for dispersing $\mathcal{I}$ that disperses the intervals in a way that $c(I_{j+1}) + d_{j+1} - (c(I_j) + d_j) = s$ holds for $1 \le i \le k$ and $a_{\alpha_i} \le j < b_{\alpha_i+1}$.*

Outline of Algorithm 1

Given a collection of unit intervals $\mathcal{I}$ and a dispersal value $s \ge 1$, the algorithm starts by sorting and partitioning $\mathcal{I}$ into $m \le n$ disjoint subcollections $\mathcal{I}_{a_1,b_1}, \dots, \mathcal{I}_{a_m,b_m}$ such that each $\mathcal{I}_{a_i,b_i}$ satisfies Lemma 5. Subsequently, the optimal breakpoints are determined for each $E(\mathcal{I}_{a_i,b_i}, x)$. Whenever there exist two subcollections $\mathcal{I}_{a_i,b_j}$, $i \le j$ and $\mathcal{I}_{a_k,b_\ell}$, $k \le \ell$ that intersect when equispaced, the algorithm considers both subcollections as a unique subcollection $\mathcal{I}_{a_i,b_\ell} = \mathcal{I}_{a_i,b_j} \cup \mathcal{I}_{a_k,b_\ell}$ and recursively determines the optimal breakpoints of $E(\mathcal{I}_{a_i,b_\ell}, x)$ using the breakpoint sets of $E(\mathcal{I}_{a_i,b_j}, x)$ and $E(\mathcal{I}_{a_k,b_\ell}, x)$. Lemma 8 ensures that this recursion partitions $\mathcal{I}$ into non-intersecting subcollections when equispaced. Lastly, the algorithm returns the total moving distance, which is calculated as the sum of the optimal values of E for each subcollection.

Before showing the complexity of the algorithm, we must characterise the set of breakpoints further. When a collection of unit intervals $\mathcal{I} = \{I_1, \dots, I_n\}$ is partitioned into m disjoint subcollections $\mathcal{I}_{a_1,b_1}, \dots, \mathcal{I}_{a_m,b_m}$ of intervals that satisfy Lemma 5, the set of breakpoints $B_{\mathcal{I}_{a_i,b_i}}$ is equal to $\{c(I_j) + (|\mathcal{I}_{a_i,b_i}| - j)s \mid I_i \in \mathcal{I},\ a_i \le j \le b_i\}$ for each $1 \le i \le m$. Consequently, $B_{\mathcal{I}}$ can be reformulated as follows:

$$B_{\mathcal{I}} = \left\{ c(I_j) + \left(|\mathcal{I}_{a_i,b_i}| - j + \sum_{k=i+1}^{m} |\mathcal{I}_{a_k,b_k}| \right) s \mid 1 \le i \le m,\ a_i \le j \le b_i \right\}.$$

As a result, if b and b' are the breakpoints for I in $B_{\mathcal{I}_{a_i,b_i}}$ and $B_{\mathcal{I}}$, respectively, then $b' = b - \sum_{j=i+1}^{m} |\mathcal{I}_{a_j,b_j}|$ holds. Moreover, the breakpoints of any union of subcollections $\mathcal{I}_{a_i,b_j} = \mathcal{I}_{a_i,b_i} \cup \dots \cup \mathcal{I}_{a_j,b_j}$ can be calculated in the same way by subtracting $\sum_{k=j+1}^{m} |\mathcal{I}_{a_k,b_k}|$ from any breakpoint $b \in B_{\mathcal{I}}$ calculated using an interval $I \in \mathcal{I}_{a_i,b_j}$. It follows that the order of $B_{\mathcal{I}_{a_i,b_j}}$ is the same as the order of the corresponding breakpoints in $B_{\mathcal{I}}$.

Algorithm 1 Dispersing n unit intervals in $O(n \log n)$ time.

1 **Procedure** DispersingIntervals($\mathcal{I}$,s)
2 Sort and partition $\mathcal{I}$ into $m \leq n$ subcollections $\mathcal{I}_{a_1,b_1}, \ldots, \mathcal{I}_{a_m,b_m}$ such that for all $1 \leq i \leq m$, $c(I_{j+1}) - c(I_j) \leq s$ for $a_i \leq j \leq b_i - 1$.;
3 Compute and sort $B^*_{\mathcal{I}_{a_i,b_i}}$ for all $1 \leq i \leq n$.;
4 $x^1_{a_i,b_i}, x^2_{a_i,b_i}$ are the breakpoint $b^{\mathcal{I}_{a_i,b_i}}_{(n+1)/2}$ if $|\mathcal{I}_{a_i,b_i}|$ is odd and $b^{\mathcal{I}_{a_i,b_i}}_{n/2}$ and $b^{\mathcal{I}_{a_i,b_i}}_{(n/2)+1}$ otherwise.;
5 $D^{opt} \leftarrow \bigcup_{1 \leq i \leq n} \{(B^*_{\mathcal{I}_{a_i,b_i}}, x^1_{a_i,b_i}, x^2_{a_i,b_i})\}$;
6 **while** $x^1_{a_k,b_\ell} \leq x^2_{a_i,b_j} + |\mathcal{I}_{a_k,b_\ell}|s$, $1 \leq i \leq j < k \leq \ell \leq n$ **do**
7 $B^*_{\mathcal{I}_{a_i,b_l}} \leftarrow merge(B^*_{\mathcal{I}_{a_i,b_j}}, B^*_{\mathcal{I}_{a_k,b_\ell}})$;
8 $x^1_{a_i,b_\ell}, x^2_{a_i,b_\ell} \leftarrow b^{\mathcal{I}_{a_i,b_\ell}}_{(n+1)/2}$ if $|B^*_{\mathcal{I}_{a_i,b_\ell}}|$ is odd and $x^1_{a_i,b_\ell} \leftarrow b^{\mathcal{I}_{a_i,b_\ell}}_{n/2}$, $x^2_{a_i,b_\ell} \leftarrow b^{\mathcal{I}_{a_i,b_\ell}}_{(n/2)+1}$ otherwise.;
9 $D^{opt} \leftarrow \Big(D^{opt} \setminus \Big\{(B^*_{\mathcal{I}_{a_i,b_j}}, x^1_{a_i,b_j}, x^2_{a_i,b_j}), (B^*_{\mathcal{I}_{a_k,b_l}}, x^1_{a_k,b_\ell}, x^2_{a_k,b_\ell})\Big\}\Big) \cup \Big\{(B^*_{\mathcal{I}_{a_i,b_\ell}}, x^1_{a_i,b_\ell}, x^2_{a_i,b_\ell})\Big\}$;
10 **return** $\sum_{(B^*_{\mathcal{I}_{a_i,b_j}}, x_1, x_2) \in D^{opt}} E(\mathcal{I}_{a_i,b_i} \cup \cdots \cup \mathcal{I}_{a_j,b_j}, x_1)$;

The above implies that the breakpoints of any (union of) subcollection(s) can be obtained from $B_{\mathcal{I}}$. We denote the set $\bigcup_{i \leq k \leq j} \left\{ b + s \sum_{l=k+1}^{m} |\mathcal{I}_{a_l,b_l}| \mid b \in B_{\mathcal{I}_{a_k,b_k}} \right\}$ by $B^*_{\mathcal{I}_{a_i,b_j}}$ and call it the *cumulative set of breakpoints of* $B_{\mathcal{I}_{a_i,b_j}}$. We prove that $B^*_{\mathcal{I}_{a_1,b_1}}, \ldots, B^*_{\mathcal{I}_{a_m,b_m}}$ can be found in $O(n \log n)$ time.

▶ **Lemma 9** (⋆). *Let $\mathcal{I} = \{I_1, \ldots, I_n\} = \mathcal{I}_{a_1,b_1} \cup \cdots \cup \mathcal{I}_{a_m,b_m}$ be a collection of n unit intervals partitioned as above. Then the cumulative sets of breakpoints $B^*_{\mathcal{I}_{a_1,b_1}}, \ldots, B^*_{\mathcal{I}_{a_m,b_m}}$ such that each $B^*_{\mathcal{I}_{a_i,b_i}}$ is sorted can be obtained in $O(n \log n)$ total time.*

▶ **Lemma 10** (⋆). *Let $\mathcal{I} = \{I_1, \ldots, I_n\} = \mathcal{I}_{a_1,b_1} \cup \cdots \cup \mathcal{I}_{a_m,b_m}$ be a collection of n unit intervals partitioned as above. If cumulative breakpoint sets $B^*_{\mathcal{I}_{a_1,b_1}}, \ldots, B^*_{\mathcal{I}_{a_m,b_m}}$ are given so that each $B^*_{\mathcal{I}_{a_i,b_i}}$ is sorted, then merging them into one sorted set can be done in $O(n \log n)$ total time.*

▶ **Theorem 11.** *Given a collection of unit intervals $\mathcal{I}$ and a value $s \geq 1$, Interval Dispersal can be solved in $O(n \log n)$ time.*

Proof. We show the complexity of Algorithm 1. Line 2 can be done in $O(n \log n)$ time for sorting and $O(n)$ time to determine the initial m partitions. Similarly, line 3 can be done in $O(n \log n)$ time by Lemma 9. Given that each $B^*_{\mathcal{I}_{a_i,b_i}}$ is sorted, the $((|\mathcal{I}_{a_i,b_i}|+1)/2)$th element (resp. $(|\mathcal{I}_{a_i,b_i}|/2)$th and $((|\mathcal{I}_{a_i,b_i}|/2)+1)$th element) can be calculated in $O(\log |\mathcal{I}_{a_i,b_i}|)$ time using binary search on $B^*_{\mathcal{I}_{a_i,b_i}}$. This ensures that line 4 is done for all $1 \leq i \leq m$ in $O(m \log n)$ total time. We initialise D^{opt} as a doubly linked list where each node i contains the information of $(B^*_{\mathcal{I}_{a_i,b_i}}, x^1_{a_i,b_i}, x^2_{a_i,b_i})$. We show the complexity of the loop in line 6. We merge both $B^*_{\mathcal{I}_{a_i,b_j}}$ and $B^*_{\mathcal{I}_{a_k,b_\ell}}$ to obtain a sorted $B^*_{\mathcal{I}_{a_i,b_\ell}}$. Hence, the median value(s) of $B^*_{\mathcal{I}_{a_i,b_\ell}}$ can be calculated in $O(\log n)$ time by binary search. At each execution of line 7, two partitions are merged; thus the number of partitions is reduced by one unit at each iteration. Initially, there exist m partitions, and hence the loop of line 6 iterates at most $m - 1$ times. Moreover, merging m cumulative sets of breakpoints into one sorted set can be done in $O(n \log n)$ time by Lemma 10, which implies that any partial merge of these sets is also bounded by $O(n \log n)$. Consequently, the total running time of line 6 is $O(n \log n)$ time. Lastly, in line 9 the two merged sets are deleted and the new one is added. Since D^{opt} is a doubly linked list, this can be done in $O(1)$ time by connecting the previous and next node of $B^*_{\mathcal{I}_{a_i,b_j}}$ and $B^*_{\mathcal{I}_{a_k,b_\ell}}$ to a new node containing $B^*_{\mathcal{I}_{a_i,b_\ell}}$, respectively. Once there is no pair of subcollections left to merge, the total moving distance is calculated in $O(n)$ time in line 10 following the definition of cumulative set of breakpoints, which concludes that the total running time of Algorithm 1 is $O(n \log n)$ time. ◀

Theorem 11 implies the following result for $\Pi_{\texttt{edgeless}}$ on unit interval graphs when $s = 1$.

▶ **Corollary 12** (⋆). *Given a collection of n unit intervals $\mathcal{I}$, Geometric Graph Edit Distance can be solved in $O(n \log n)$ time so that $G(\mathcal{I}) \in \Pi_{\texttt{edgeless}}$.*

3.1 Classes $\Pi_{\texttt{acyc}}$ and $\overline{\Pi_{k\texttt{-clique}}}$ on Unit Interval Graphs

This section shows how to use Algorithm 1 for obtaining graphs in $\Pi_{\texttt{acyc}}$ and $\overline{\Pi_{k\texttt{-clique}}}$ on unit interval graphs. We first show the case for $\overline{\Pi_{k\texttt{-clique}}}$.

It is shown in [16] that given a collection of unit intervals $\mathcal{I}$, $G(\mathcal{I})$ does not contain a k-clique if and only if $c(I_{i+k-1}) - c(I_i) \geq 1$ for all $1 \leq i \leq n-k+1$. This inequality can be decomposed into $k-1$ inequalities of the following form: for each $0 \leq r \leq k-2$, $c(I_{i+k-1}) - c(I_i) \geq 1$ for all $1 \leq i \leq n-k+1$ such that $i \bmod k-1 = r$. If $\mathcal{I}$ is decomposed into $k-1$ subcollections such that $\mathcal{I} = \bigcup_{1 \leq i \leq k-1} \mathcal{I}_i$, $\mathcal{I}_i = \{I_j \in \mathcal{I} \mid 1 \leq j \leq n,\ j \pmod{k-1} = i\}$, then Algorithm 1 can be applied to each $\mathcal{I}_i$ independently for $s = 1$ to satisfy the above inequalities. Since unit interval graphs are chordal, G is acyclic if it is triangle-free; i.e. G is contained in $\overline{\Pi_{3\texttt{-clique}}}$. Consequently $\Pi_{\texttt{acyc}}$ is equivalent to $\overline{\Pi_{k\texttt{-clique}}}$ when $k = 3$. The above ideas imply Corollary 13.

▶ **Corollary 13.** *Given a collection of n unit intervals $\mathcal{I}$, Geometric Graph Edit Distance can be solved in $O(n \log n)$ time so that (i) $G(\mathcal{I}) \in \Pi_{\texttt{acyc}}$ and (ii) $G(\mathcal{I}) \in \overline{\Pi_{k\texttt{-clique}}}$.*

4 Minimising the Total Moving Distance for $\Pi_{\texttt{edgeless}}$ on Weighted Interval Graphs is Hard

In this section we show that Geometric Graph Edit Distance is strongly NP-hard on weighted interval graphs for $\Pi_{\texttt{edgeless}}$. We show a reduction from 3-Partition [13]. 3-Partition receives as input a set A of $3m$ elements, a bound $B \in \mathbb{Z}^+$ and a size $s(a) \in \mathbb{Z}^+$ such that $B/4 < s(a) < B/2$ and $\sum_{a \in A} s(a) = mB$, and the task is to decide whether A can be partitioned into m disjoint sets $A_1, \ldots, A_m$ such that for $1 \leq i \leq m$, $|A_i| = 3$ and $\sum_{a \in A_i} s(a) = B$.

Given an instance (A, B, s) of 3-Partition, we construct a collection of intervals $\mathcal{I}_A$ and show that A can be partitioned if and only if $\mathcal{I}_A$ can be modified so that $G(\mathcal{I}_A) \in \Pi_{\texttt{edgeless}}$ with at most total moving distance T. Given two intervals I, I' such that $c(I) \leq c(I')$, we say that I and I' intersect if $c(I') - c(I) < (\text{len}(I') + \text{len}(I))/2$.

We show the construction of $\mathcal{I}_A$ (see Figure 1). We define $\mathcal{I}_A$ as the collection $\mathcal{I} \cup \mathcal{I}^s \cup \mathcal{I}^b$ where $\mathcal{I} = \{I_1, \ldots, I_{3m}\}$, $\mathcal{I}^s = \{I^s_1, \ldots, I^s_{m-1}\}$, $\mathcal{I}^b = \{I_\ell, I_r\}$ and,

(i) for $1 \leq i \leq 3m$, I_i is an interval such that $\text{len}(I_i) = s(a_i)$ and $c(I_i) = -s(a_i)/2$ (that is, $r(I_i) = 0$),

(ii) for $1 \leq i \leq m-1$, I^s_i is an interval where $\text{len}(I^s_i) = B$ and $c(I^s_i) = (2i-1)B + B/2$ and

(iii) I_ℓ and I_r are intervals such that $\text{len}(I_\ell) = \text{len}(I_r) = 3Bm^2 + \max_{a \in A} s(a)$, $c(I_\ell) = -3Bm^2/2$ and $c(I_r) = (2m-1)B + 3Bm^2/2$.

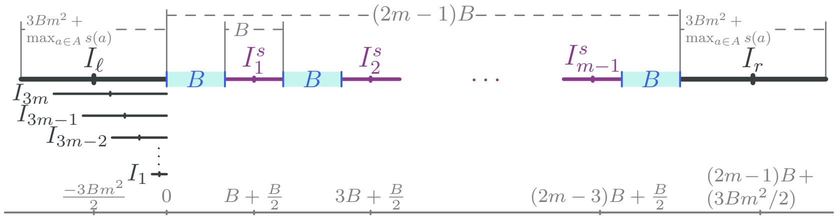

Figure 1 Reduction Overview.

For an interval $I \in \mathcal{I}_A$, we define the moving distance function $d_I : \mathbb{R} \to \mathbb{R}$ as:

$$d_I(x) = \begin{cases} |c(I) - x|, & I \in \mathcal{I}, \\ 12Bm^2|c(I) - x|, & I \in \mathcal{I}^s \cup \mathcal{I}^b. \end{cases}$$

Given an instance (A, B, s) of 3-Partition, we show the following properties.

▶ **Lemma 14** ($\star$). *Given an arbitrary partition of A of m disjoint sets $A_1, \dots, A_m$ such that $A_i = \{a_1^i, a_2^i, a_3^i\}$ for $1 \le i \le m$, $\sum_{i=1}^m 6(i-1)B + \sum_{i=1}^m (3a_1^i + 2a_2^i + a_3^i) < 3Bm^2$ holds.*

We note that Lemma 14 works for any partition of A as described above, even without the restrictions of the 3-Partition output.

▶ **Lemma 15.** *Given an instance (A, B, s) of 3-Partition, A can be partitioned into m disjoint sets $A_1, \dots, A_m$ such that for $1 \le i \le m$ $A_i = \{a_1^i, a_2^i, a_3^i\}$, $|A_i| = 3$ and $\sum_{a \in A_i} s(a) = B$ if and only if $I\mathcal{I}_A$ can be modified so that $G(\mathcal{I}_A) \in \Pi_{\texttt{edgeless}}$ with total moving distance of at most $3Bm^2$.*

Lastly, we remark that the polynomial construction of $\mathcal{I}_A$ is straightforward by iterating over A and following the definitions given at the beginning of the section. We summarise the main result of this section as follows:

▶ **Theorem 16.** *Geometric Graph Edit Distance is strongly NP-hard on weighted interval graphs for the class $\Pi_{\texttt{edgeless}}$.*

We notice that Theorem 16 can be extended to show that obtaining graphs in $\Pi_{\texttt{acyc}}$ and $\overline{\Pi_{k\texttt{-clique}}}$ is also strongly NP-hard.

In particular, when obtaining a graph in $\overline{\Pi_{k\texttt{-clique}}}$, we create $k-1$ overlapping copies of the intervals in $\mathcal{I}^s \cup \mathcal{I}^b$ and add $k-1$ overlapping intervals of size B into the spaces between intervals of $\mathcal{I}^s \cup \mathcal{I}^b$ with the same moving distance function. Any interval forms a k-clique with the k copies of overlapping intervals. Consequently, moving the intervals of $\mathcal{I}$ with total moving distance of at most $3Bm^2$ is equivalent to removing all k-cliques from $\mathcal{I}_A$ with at most the same distance. Moreover, by the chordality of interval graphs, it is sufficient to obtain a graph in $\overline{\Pi_{k\texttt{-clique}}}$ when $k = 3$ for class $\Pi_{\texttt{acyc}}$. As a result, Corollary 17 is obtained.

▶ **Corollary 17.** *Geometric Graph Edit Distance is strongly NP-hard on weighted interval graphs for classes $\Pi_{\texttt{acyc}}$ and $\overline{\Pi_{k\texttt{-clique}}}$.*

5 Minimising the Maximum Moving Distance for $\Pi_{\texttt{edgeless}}$ on Unit Disk Graphs is Hard

In this section, we deal with the minimax version of Geometric Graph Edit Distance, defined as follows:

minimax-Geometric Graph Edit Distance

Input: A collection of n geometric objects $\mathcal{S}$ and a real $K > 0$.
Task: Decide whether $\mathcal{S}$ can be modified so that $G(\mathcal{S}) \in \Pi$ by moving objects such that for all $S \in \mathcal{S}$, the moving distance of S is at most K.

We show that minimax-Geometric Graph Edit Distance is strongly NP-hard on unit disk graphs for $\Pi = \Pi_{\texttt{edgeless}}$ over the L_1 and L_2 distances by reducing from Planar 3-SAT. Specifically, we show a proof for Theorem 18.

▶ **Theorem 18.** *minimax-Geometric Graph Edit Distance is strongly* NP*-hard on unit disk graphs for* $\Pi_{\texttt{edgeless}}$ *over the* L_1 *and* L_2 *distances.*

Due to space constraints, we only give an overview of the reduction. The complete reduction and proofs can be found in the full-version of the paper [17].

5.1 Proof Overview of Theorem 18: Reducing Planar 3-SAT to minimax-Geometric Graph Edit Distance

We show a reduction from the following NP-complete variation of Planar 3-SAT [18,21,26]. Given CNF formula Φ equipped with a planar rectilinear embedding G_Φ, a set X of n variables, a set C of m clauses over X such that each $c \in C$ has length $|c| \leq 3$, each variable $x \in X$ appears in at most three clauses, and $\Phi = \bigwedge_{c \in C} c$, Planar 3-SAT asks whether Φ is satisfiable. We give a simplified overview of the reduction. The idea is to emulate each component (clauses, variables and connectors) of G_Φ using *disk gadgets* and construct a collection of disks $\mathcal{D}_\Phi$ equivalent to G_Φ. That is, our objective is to construct a $\mathcal{D}_\Phi$ such that Φ is satisfiable if and only if $\mathcal{D}_\Phi$ is a yes-instance of minimax-Geometric Graph Edit Distance for $\Pi_{\texttt{edgeless}}$. To do this, we emulate the truth assignment using a proper movement of disks. To force the disk movement, we deliberately insert intersecting disks in $\mathcal{D}_\Phi$. In particular, we insert intersecting disks in clause gadgets and restrict the movement of such disks to moving a sequence of disks such that a *free slot* of a variable gadget is used. To allow the removal of the intersection, the gadgets are connected following the structure of G_Φ using consecutive disks separated by distance K. For example, consider the boolean formula Φ and its rectilinear embedding G_Φ, illustrated in Figure 2. A skeleton of the reduction is shown in Figure 3(a), where representations of clause and variable gadgets are connected following G_Φ. Let $c = (x_1 \vee \overline{x_2} \vee x_4)$ and suppose that x_2 is assigned to *false*. This assignment implies a movement of disks that (i) removes the intersections in the clause gadget for c and (ii) blocks the truth value of the variable gadget for x_2 (see Figure 3(b)). We must block the truth value of the variable gadget so that another clause gadget c' does not use the free slot in the variable gadget for x_2 when $x_2 = \textit{true}$. Consequently, their intersections must be removed using other gadgets. It can be shown that removing all intersections in this way is equivalent to a valid assignment of variables for which $\Phi = \textit{true}$. The disks are *moved* by assigning a new location, and the distance is calculated using a function that we call *moving*

$$\Phi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee \overline{x_3} \vee \overline{x_4})$$

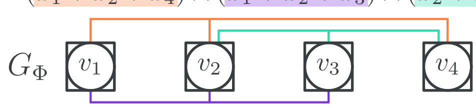

Figure 2 Reduction Overview: An arbitrary instance Φ of Planar 3-SAT with its rectilinear embedding G_Φ.

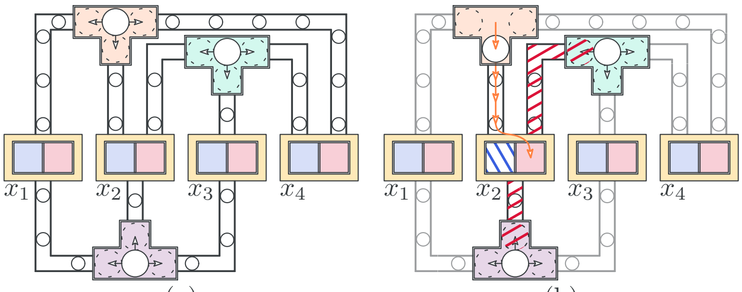

Figure 3 Reduction Overview: (a) The skeleton given by the instance (Φ, G_Φ) of Figure 2; (b) The intersection of the gadget for $c = (x_1 \vee \overline{x_2} \vee x_4)$ is removed by moving disks in a way that a free slot of the gadget for x_2 is used. Since $c = \mathit{true}$ when $x_2 = \mathit{false}$, the free slots for the other two gadgets become blocked, being unable to remove their intersection using the variable gadget for x_2.

distance function, which is the L_1 or L_2 distance metric multiplied by a *distance weight*. We employ two types of disks classified by their distance weight, called *transition disk* and *heavy disk*. The transition disks are the disks that we aim to move, whereas heavy disks are used to restrict the movement of transition disks. The moving distance function of a heavy disk is intuitively defined such that any significant movement that alters the construction exceeds a distance of K. We show that a solution that allows removing all intersections from $\mathcal{D}_\Phi$ with minimum maximum moving distance K exclusively relies on the movement of transition disks. We remark that, although heavy disks can move, their movement is negligible. Combining this condition and the above construction, it can be shown that Φ is satisfiable if and only if $\mathcal{D}_\Phi$ can be modified so that $G(\mathcal{D}_\Phi) \in \Pi_{\texttt{edgeless}}$ using minimum maximum moving distance K.

6 Concluding Remarks

The main contribution of this paper is two-fold. First, we continued the study of Geometric Graph Edit Distance originally presented in [16], showing complexity results for obtaining graphs in several classes for sparse graphs on interval graphs. In particular, we showed that obtaining a graph in $\Pi_{\texttt{edgeless}}$, $\Pi_{\texttt{acyc}}$ and $\overline{\Pi_{k\texttt{-clique}}}$ is solvable in $O(n \log n)$ time on unit interval graphs. In contrast, we showed that the problem becomes strongly NP-hard on weighted interval graphs for the same classes. Second, we defined minimax-Geometric Graph Edit Distance as a variation of the above problem and showed that it is strongly NP-hard for $\Pi_{\texttt{edgeless}}$ on weighted unit disk graphs over the L_1 and L_2 distances.

There are several directions for further research. Our results provide a comprehensive picture of the complexity of Geometric Graph Edit Distance on interval graphs. In particular, we showed that the problem becomes hard even in lower dimensions when the input is not restricted by interval size and distance weight. As a result, a potential future work is to study the complexity when exclusively one of the restrictions is applied. Another interesting direction is to study the model for $\Pi_{\texttt{edgeless}}$ in higher dimensions. Related works [9–11] suggest that our model on more complex intersection graphs becomes intractable for some of the graph classes presented in this work. In general, we deal with the edit operation that moves the objects of the given intersection graph. However, the model is not restricted to this operation. Determining Geometric Graph Edit Distance using other geometric edit operations (such as shrinking or rotating objects) is left for future research for all intersection graphs and graph classes presented in this work.

References

1 Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. doi:10.1017/cbo9780511804441.

2 Pablo Burzyn, Flavia Bonomo, and Guillermo Durán. NP-completeness results for edge modification problems. *Discrete Applied Mathematics*, 154(13):1824–1844, 2006. doi:10.1016/j.dam.2006.03.031.

3 Fan R. K. Chung and David Mumford. Chordal Completions of Planar Graphs. *J. Comb. Theory Ser. B*, 62(1):96–106, 1994. doi:10.1006/jctb.1994.1056.

4 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 3rd Edition*. MIT Press, 2009.

5 Christophe Crespelle, Pål Grønås Drange, Fedor V. Fomin, and Petr A. Golovach. A survey of parameterized algorithms and the complexity of edge modification. *Computer Science Review*, 48:100556, 2023. doi:10.1016/j.cosrev.2023.100556.

6 Mark de Berg, Sándor Kisfaludi-Bak, and Gerhard J. Woeginger. The complexity of dominating set in geometric intersection graphs. *Theoretical Computer Science*, 769:18–31, 2019. doi:10.1016/j.tcs.2018.10.007.

7 Reinhard Diestel. *Graph Theory, 5th Edition*. Graduate texts in mathematics. Springer, Berlin, Germany, 2017. doi:10.1007/978-3-662-53622-3.

8 Pål Grønås Drange. *Parameterized Graph Modification Algorithms*. PhD thesis, The University of Bergen, 2015. URL: https://bora.uib.no/bora-xmlui/handle/1956/10774.

9 Jirí Fiala, Jan Kratochvíl, and Andrzej Proskurowski. Systems of distant representatives. *Discrete Applied Mathematics*, 145(2):306–316, 2005. doi:10.1016/j.dam.2004.02.018.

10 Fedor V. Fomin, Petr A. Golovach, Tanmay Inamdar, Saket Saurabh, and Meirav Zehavi. Kernelization for spreading points. In *ESA 2023*, volume 274 of *LIPIcs*, pages 48:1–48:16, 2023. doi:10.4230/LIPICS.ESA.2023.48.

11 Fedor V. Fomin, Petr A. Golovach, Tanmay Inamdar, Saket Saurabh, and Meirav Zehavi. Parameterized geometric graph modification with disk scaling. In *ITCS 2025*, volume 325 of *LIPIcs*, pages 51:1–51:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPICS.ITCS.2025.51.

12 Fedor V. Fomin, Saket Saurabh, and Neeldhara Misra. Graph modification problems: A modern perspective. In *Frontiers in Algorithmics*, pages 3–6. Springer International Publishing, 2015. doi:10.1007/978-3-319-19647-3_1.

13 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1979.

14 Marc Hellmuth, Manuela Geiß, and Peter F. Stadler. Complexity of modification problems for reciprocal best match graphs. *Theoretical Computer Science*, 809:384–393, 2020. doi:10.1016/j.tcs.2019.12.033.

15 Hung P. Hoang, Stefan Lendl, and Lasse Wulf. Assistance and interdiction problems on interval graphs. *Discrete Applied Mathematics*, 340:153–170, 2023. doi:10.1016/j.dam.2023.06.046.

16 Nicolás Honorato-Droguett, Kazuhiro Kurita, Tesshu Hanaka, and Hirotaka Ono. Algorithms for optimally shifting intervals under intersection graph models. In *IJTCS-FAW 2024*, volume 14752, pages 66–78. Springer, 2024. doi:10.1007/978-981-97-7752-5_5.

17 Nicolás Honorato-Droguett, Kazuhiro Kurita, Tesshu Hanaka, and Hirotaka Ono. On the complexity of minimising the moving distance for dispersing objects, 2025. doi:10.48550/arXiv.2502.12903.

18 Donald E. Knuth and Arvind Raghunathan. The problem of compatible representatives. *SIAM Journal on Discrete Mathematics*, 5(3):422–427, 1992. doi:10.1137/0405033.

19 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-Complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

20 Heng Li and Richard Durbin. Fast and accurate short read alignment with burrows–wheeler transform. *Bioinformatics*, 25(14):1754–1760, 2009. doi:10.1093/bioinformatics/btp324.

21 David Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, 11(2):329–343, 1982. doi:10.1137/0211025.

22 Terry A. McKee and F. R. McMorris. *Topics in Intersection Graph Theory.* Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics, 1999. doi:10.1137/1.9780898719802.

23 Fahad Panolan, Saket Saurabh, and Meirav Zehavi. Contraction decomposition in unit disk graphs and algorithmic applications in parameterized complexity. *ACM Trans. Algorithms*, 20(2):15, 2024. doi:10.1145/3648594.

24 Franco P. Preparata and Michael Ian Shamos. *Computational Geometry - An Introduction.* Texts and Monographs in Computer Science. Springer, 1985. doi:10.1007/978-1-4612-1098-6.

25 R. Sritharan. Graph modification problem for some classes of graphs. *Journal of Discrete Algorithms*, 38-41:32–37, 2016. doi:10.1016/j.jda.2016.06.003.

26 Craig A. Tovey. A simplified NP-complete satisfiability problem. *Discrete Applied Mathematics*, 8(1):85–89, 1984. doi:10.1016/0166-218x(84)90081-7.