Revisiting Graph Modification via Disk Scaling: From One Radius to Interval-Based Radii

Thomas Depian
Algorithms and Complexity Group, TU Wien, Austria

Frank Sommer
Institute of Computer Science, Friedrich Schiller University Jena, Germany

Abstract

For a fixed graph class Π, the goal of Π-Modification is to transform an input graph G into a graph $H \in \Pi$ using at most k modifications. Vertex and edge deletions are common operations, and their (parameterized) complexity for various Π is well-studied. Classic graph modification operations such as edge deletion do not consider the geometric nature of intersection graphs such as (unit) disk graphs. This led Fomin et al. [ITCS' 25] to introduce scaling as a geometric graph modification operation for unit disk graphs: For a given radius r, each modified disk will be rescaled to radius r.

In this paper, we generalize their model by allowing rescaled disks to choose a radius within a given interval $[r_{\min}, r_{\max}]$ and study the (parameterized) complexity (with respect to k) of the corresponding problem Π-Scaling. We show that Π-Scaling is in XP for every graph class Π that can be recognized in polynomial time. Furthermore, we show that Π-Scaling: (1) is NP-hard and FPT for cluster graphs, (2) can be solved in polynomial time for complete graphs, and (3) is W[1]-hard for connected graphs. In particular, (1) and (2) answer open questions of Fomin et al. and (3) generalizes the hardness result for their variant where the set of scalable disks is restricted.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Computational geometry

Keywords and phrases NP-hardness, Parameterized Complexity, Unit Disk Graphs, Cluster Graphs, Connected Graphs

Digital Object Identifier 10.4230/LIPIcs.ESA.2026.107

Related Version *Full Version*: https://arxiv.org/abs/2603.05358 [7]

Funding *Thomas Depian*: Supported by the Vienna Science and Technology Fund (WWTF) under grant ICT22-029.
Frank Sommer: Supported by the Alexander von Humboldt Foundation.

1 Introduction

For a fixed graph class Π, the task in the Π-Modification problem is to transform an input graph G into a graph $H \in \Pi$ using at most k modifications. Common modification operations include vertex deletion, edge deletion, and edge addition [4], but other operations such as vertex-splitting have also been studied [1]. The parameterized complexity of Π-Modification is well-understood, especially when parameterized by the number k of modifications [4, 5, 20, 28, 29, 35]. Usually, we aim for an FPT-algorithm, i.e., an algorithm with running time $f(k) \cdot n^{\mathcal{O}(1)}$, where f is a computable function that depends only on k, and n is the overall instance size [5]. Some parameterized problems, however, are W[t]-hard with respect to k for some $t \geq 1$. This rules out an FPT-algorithm for k under common complexity assumptions, and we then aim for an XP-algorithm, i.e., with running time $n^{f(k)}$.

Due to their applications in wireless communication [24] and capabilities to model sensor networks [33], *geometric intersection graphs* are an important graph family. There, each vertex corresponds to a geometric object and two vertices are adjacent if and only if the two corresponding objects intersect. The classical (non-parameterized) complexity of


© Thomas Depian and Frank Sommer;
licensed under Creative Commons License CC-BY 4.0

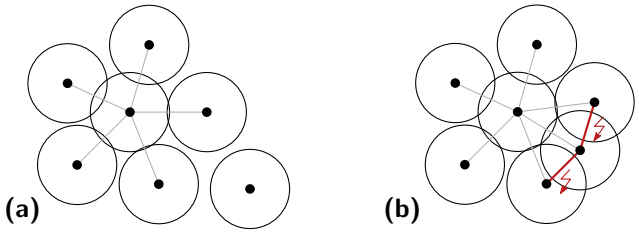


Figure 1 (a) Turning the underlying abstract graph into a $K_{1,6}$ by adding a single edge is easy. However, this is not possible while maintaining a unit disk graph as **(b)** demonstrates.

problems restricted to geometric intersection graphs is well-understood [15, 22], and also their parameterized complexity has been studied recently [6, 18, 19, 31, 34]. Disk graphs are probably the most prominent family of geometric graphs. In a disk graph $G(\mathcal{S}, r)$, each vertex corresponds to a point $p \in \mathcal{S}$ in the plane and two vertices p and q are adjacent if and only if their corresponding *closed* disks with radius $r(p)$ and $r(q)$ intersect; see also Figure 1 for an example. In unit disk graphs, each disk has radius one and their geometric nature can lead to specialized, more efficient, algorithms on these graphs. One prominent example is the Clique problem (find k vertices which are pairwise adjacent), which is NP-hard on general graphs [26], but can be solved in polynomial-time on unit disk graphs [3, 12, 14].

Traditionally, vertex- and edge-based modifications were the primary modification operations for Π-Modification, even when restricted to geometric intersection graphs [34]. However, some of these operations, such as edge addition, ignore the underlying geometry as seen in Figure 1. This raises the question of their appropriateness for unit disk and other geometric intersection graphs. As a response to this, two "geometry-preserving" modification operations were recently studied: (a) disk dispersion [9–11, 16], where disks can move within a bounded radius, and (b) disk scaling [17], where disks can receive a radius different from 1.

We focus on the latter model, motivated by the observation that in applications such as sensor networks, the position of a sensor is fixed but its transmission range, modeled as the radius, can be adapted [33]. More precisely, in the model introduced by Fomin et al. [17], the input contains a non-negative number $\alpha \neq 1$, and each modified disk receives radius α, i.e., all scaled disks are either *shrunken* ($\alpha < 1$) or *expanded* ($\alpha > 1$). They studied disk-scaling for obtaining an independent set, an acyclic graph, or a connected graph, and provided FPT-algorithms with respect to k, as well as (complementing) W[1]- and NP-hardness results. A limitation of their model is that each scaled disk receives the same radius α. In particular, all modified disks are either shrunken or expanded.[1] In this work, we generalize their model by allowing each rescaled disk to choose a radius within a given interval $[r_{\min}, r_{\max}]$, hence enabling disk shrinking and expanding simultaneously whenever $r_{\min} < 1$ and $r_{\max} > 1$:

Scaling To Π (Π-Scaling)
Input: Set $\mathcal{S}$ of n points in the plane, $0 < r_{\min} \leq r_{\max} \in \mathbb{R}$, and $k \in \mathbb{N}$.
Question: Does there exist a set $\mathcal{T} \subseteq \mathcal{S}$ and a radius assignment $r \colon \mathcal{S} \to \mathbb{R}_{>0}$ such that $|\mathcal{T}| \leq k$, $r_{\min} \leq r(p) \leq r_{\max}$ for all $p \in \mathcal{T}$ and $r(p) = 1$ otherwise, and $G(\mathcal{S}, r) \in \Pi$?

[1] Fomin et al. [17] also studied an optimization variant where one scales (at most) k disks within a given interval $[\alpha, 1]$ for $0 < \alpha < 1$ and minimizes the sum of all radii. This model also has the limitation that all disks are either shrunken or expanded. We do not consider this "min-variant" in this work, but discuss it, along with other interesting generalizations, in Section 8.

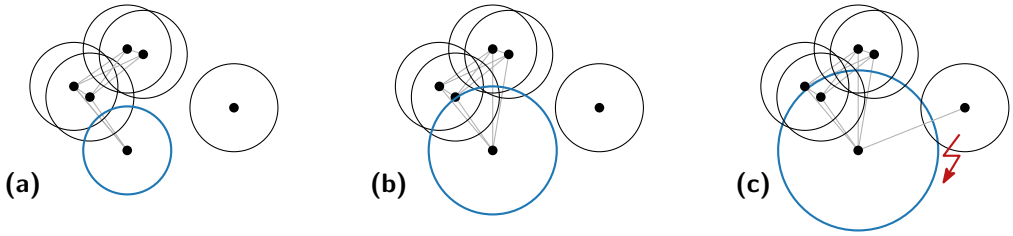


Figure 2 We can turn **(a)** into a cluster graph by scaling the blue disk as shown in **(b)**. However, if we enlarge the disk too much as in **(c)**, the resulting disk graph might no longer be a cluster graph.

Intuitively, Π-Scaling asks whether one can modify at most k radii in a given unit disk graph $G(\mathcal{S}, \mathbb{1})$ to values in $[r_{\min}, r_{\max}]$ such that the resulting disk graph belongs to Π.

Our Contributions. We first show that Π-Scaling is in XP with respect to k for every graph class Π whose recognition problem can be solved efficiently (Theorem 3.3). At first glance this might be surprising, since, although we can guess the scaled disks $\mathcal{T}$, we cannot branch on the radius assignment r or the target graph $H \in \Pi$. To this end, we first guess for each $p \in \mathcal{T}$ its *furthest unscaled neighbor* $\mathsf{far}(p)$ in $G(\mathcal{S}, r)$. This allows us to reconstruct all neighbors of p and eventually the target graph H due to the geometric nature of the problem. Afterwards, we devise a linear programming (LP) formulation (Section 3.1) that determines suitable radii once the disks to scale, i.e., $\mathcal{T}$, and the resulting intersection graph H is fixed.

Next, we focus on specific graph classes and start with cluster graphs (see Section 2 for the definition of the graph classes). Cluster graphs are in particular challenging since they can require to choose a specific radius inside the interval $[r_{\min}, r_{\max}]$; see also Figure 2. We improve the upper bound implied by Theorem 3.3 and provide an FPT-algorithm for Scaling To Cluster parameterized by k (Theorem 4.1). The algorithm uses a branch-and-bound procedure to iteratively determine a new disk p to scale. However, since we do not know the final radius of p beforehand, we have to further branch to determine the adjacencies of p; essentially gradually fixing the target cluster graph H. To ensure FPT-time, we exploit the geometry within our problem. In particular, we show that there is only a bounded number of "ambiguous" neighbors and non-neighbors of p: $\mathsf{far}(p)$, this time the furthest unscaled disk in the *same* cluster as p in the solution, and $\mathsf{clo}(p)$, which is the *closest* unscaled disk not adjacent to p, i.e., $\mathsf{clo}(p)$ and p are in *different* clusters in the solution. Once $\mathsf{far}(p)$ and $\mathsf{clo}(p)$ are determined, we can essentially "read-off" the (unscaled) neighbors of p in H. To complement this result, we show that Scaling To Cluster is NP-hard for any fixed rationals $0 < r_{\min} \leq r_{\max}$ (Theorem 5.4), except for the case when $r_{\min} = 1 = r_{\max}$.

As the second special graph class we consider complete graphs. Here, it is optimal to scale each modified disk to $r_{\max}$. We show that the corresponding problem Scaling To Complete can be solved in polynomial time (Theorem 6.3) by leveraging the polynomial-time algorithm for Clique in unit disk graphs [14]. Theorems 4.1 and 6.3 settle two open questions of Fomin et al. [17]; for cluster graphs, even in our more general model with interval-based radii, for complete graphs, there is no (algorithmic) difference in the models.

As our last result, we show that the general XP-algorithm (Theorem 3.3) is for some graph classes Π the best we can hope for, i.e., cannot be improved to an FPT-algorithm. For example, if we want to obtain a graph that contains an independent set of size at least c, the problem Scaling To c-IS is W[1]-hard for c, even when $k = 0$. This follows from the known W[1]-hardness of detecting a size c independent set in a unit disk graph [5, Theorem 14.34]. In Section 7, we focus on a graph class where obtaining W[1]-hardness is "less-trivial". More

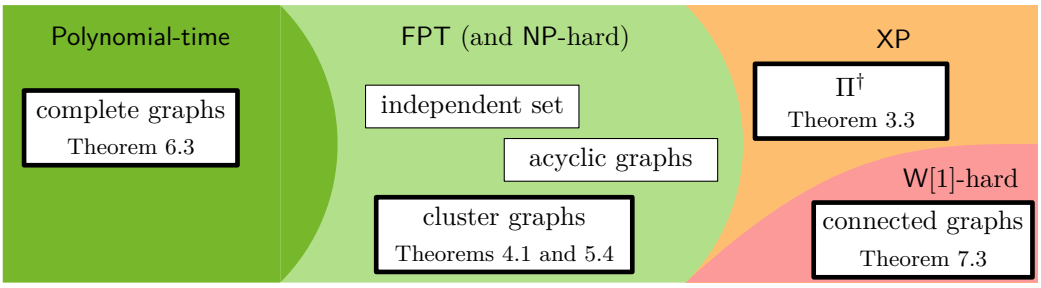


Figure 3 The (parameterized) complexity of Π-Scaling. Bold boxes are results from this paper and the remaining follow from [17]. †: Assuming that recognizing graphs in Π takes polynomial time.

concretely, we consider connected graphs, and show that Scaling To Connected is W[1]-hard parameterized by k (Theorem 7.7). More precisely, in our reduction all scaled disks get expanded (with the same radius). Hence, our result generalizes a result of Fomin et al. [17], who proved W[1]-hardness with respect to k when $r_{\min} = r_{\max} > 1$ in the variant where the set of scalable (expandable) disks is restricted. This is in sharp contrast to the case when *at least* k disks must be shrunken to a radius α, which is FPT with respect to k [17, Theorem 3]. Thus, the complexity of Π-Scaling can differ drastically depending on $r_{\min} \geq 1$ or $r_{\max} \leq 1$.

Finally, we would like to emphasize that many (algorithmic) results obtained by Fomin et al. [17] directly transfer to our more general model. This is due to the observation that they considered graph classes where it is always safe to either choose only $r_{\max}$ or only $r_{\min}$ as the radius for each scaled disk. We summarize our results and this observation in Figure 3.

Full proofs of statements marked by ★ are deferred to the full version [7].

2 Preliminaries

We assume familiarity with the basic notions from graph theory [8] and the foundations of parameterized complexity theory [5]. For our algorithmic results, we assume the *Real RAM* computational model, i.e., that we can perform arithmetic operations on real numbers in unit time [13, 17]. For an integer $p \geq 1$, we define $[p] := \{1, 2, \ldots, p\}$. Let G be a simple and undirected graph with vertex set $V(G)$ and edge set $E(G)$. We write V and E if G is clear from the context. For a vertex set $V' \subseteq V$, we let $G[V']$ denote the subgraph of G *induced* by V'. For a vertex $v \in V$, we let $N_G(v) = \{u \in V : uv \in E\}$ denote the *neighborhood* of v.

Graph Classes. A graph G is *complete* if and only if $uv \in E$ for every pair $u, v \in V$. We call G a *cluster graph* if every connected component C of G is a clique, also referred to as *cluster*. Equivalently, G is a cluster graph if and only if it does not contain an *induced* P_3, i.e., three vertices $u, v, w \in V$ such that $uv, vw \in E$ but $uw \notin E$. We only consider induced P_3s. Hence, a P_3*-free* graph is a graph without an induced P_3, i.e., a cluster graph.

Disk Graphs. For two points $p, p' \in \mathbb{R}^2$ in the plane, we denote with $\|p - p'\|_2$ their *Euclidean distance*. We use $x(p)$ and $y(p)$ to denote the x- and y-coordinates of p, respectively. Let $\mathcal{S} = \{p_1, \ldots, p_n\}$ be a set of n points in the plane. A *radius assignment* $r \colon \mathcal{S} \to \mathbb{R}_{>0}$ assigns to each point $p \in \mathcal{S}$ a positive *radius* $r(p)$. Given $\mathcal{S}$ and r, we define the corresponding *disk graph* $G(\mathcal{S}, r) := (\mathcal{S}, \{p_i p_j : i < j, \|p_i - p_j\|_2 \leq r(p_i) + r(p_j)\})$ as the graph that contains one vertex for each point and two vertices p_i and p_j are adjacent if and only if their closed disks of radii $r(p_i)$ and $r(p_j)$ intersect. If $r(p_i) = r(p_j)$ holds for all $p_i, p_j \in \mathcal{S}$, then $G(\mathcal{S}, r)$ is a

unit disk graph. In this case, we may assume without loss of generality $r(p) = 1$ for all $p \in \mathcal{S}$ and write $G(\mathcal{S}, \mathbb{1})$ as shorthand. Throughout this paper, we identify disks in $G(\mathcal{S}, r)$ with their center point $p \in \mathcal{S}$. Let $\mathcal{I} = (\mathcal{S}, r_{\min}, r_{\max}, k)$ be an instance of Π-Scaling. We also call k the *budget*. For a radius assignment r, we define $\mathcal{T}(r) := \{p \in \mathcal{S} : r(p) \neq 1\}$. We call the disks $\mathcal{T}(r)$ *scaled* and all remaining disks $\mathcal{S} \setminus \mathcal{T}(r)$ *unscaled*. Note that in Π-Scaling, it is sufficient to ask for a radius assignment r, since we can set $\mathcal{T} = \mathcal{T}(r)$. Although Π-Scaling is defined as a decision problem for complexity-theoretic reasons, we remark that all presented algorithms can output, for yes-instances, a witnessing radius assignment r (i.e., a *solution*).

3 A General XP-Membership Framework

Before we study Π-Scaling for different graph classes Π, we first introduce a general framework that implies XP-membership with respect to k of Π-Scaling for many graph classes Π. We condition the applicability of our framework on properties of the problem Π-Recognition, which asks, for a given graph H, whether $H \in \Pi$ holds. We next present a linear programming formulation that is a key building block in our framework.

3.1 A Linear Program for Finding Radii

In this section, we consider the problem Constrained Scaling To Graph (ConScal), defined as follows. Given a set $\mathcal{S} \subset \mathbb{R}^2$ of n points, a set $\mathcal{T} \subseteq \mathcal{S}$, two real numbers $r_{\min} \leq r_{\max} \in \mathbb{R}_{>0}$, and a graph $H = (\mathcal{S}, E)$, the question is whether there exists a radius assignment $r \colon \mathcal{S} \to \mathbb{R}_{>0}$ such that (a) $r_{\min} \leq r(p) \leq r_{\max}$ if $p \in \mathcal{T}$ and $r(p) = 1$ otherwise, and (b) $G(\mathcal{S}, r) = H$? ConScal can be seen as a restricted version of Π-Scaling, where the set of scaled disks is fixed and the target graph $H \in \Pi$ is known. Later we will "reduce" Π-Scaling to ConScal. Let $\mathcal{I} = (\mathcal{S}, \mathcal{T}, H, r_{\min}, r_{\max})$ be an instance of ConScal. Without loss of generality, we assume $G(\mathcal{S} \setminus \mathcal{T}, \mathbb{1}) = H[\mathcal{S} \setminus \mathcal{T}]$; otherwise $\mathcal{I}$ is a no-instance. We sketch a linear program $\mathcal{P}(\mathcal{I})$ to solve ConScal efficiently; see the full version [7] for details.[2]

We introduce one variable $x_p \in \mathbb{R}$ for each $p \in \mathcal{T}$ representing its radius $r(p) = x_p$. Furthermore, we introduce one "slack-variable" $\varepsilon \geq 0$ that helps us to model strict inequalities. To enforce $r_{\min} \leq r(p) \leq r_{\max}$, we add the constraints $r_{\min} \leq x_p$ and $x_p \leq r_{\max}$. Next, we consider the *unscaled neighborhood* $N_H(p) \setminus \mathcal{T}$ of each $p \in \mathcal{T}$. We must ensure that $up \in E(G(\mathcal{S}, r))$ for every $u \in N_H(p) \setminus \mathcal{T}$. Since $u \notin \mathcal{T}$ implies $r(u) = 1$, this yields the constraint $x_p + 1 \geq \|p - u\|_2$. Similarly, for every point $u \in \mathcal{S} \setminus (N_H(p) \cup \mathcal{T})$, we must have $up \notin E(G(\mathcal{S}, r))$, resulting in the constraint $x_p + 1 \leq \|p - u\|_2 - \varepsilon$. Observe that this encodes a strict inequality if $\varepsilon > 0$ holds. Finally, we consider every pair $p, q \in \binom{\mathcal{T}}{2}$. If $pq \in E(H)$, we introduce the constraint $x_p + x_q \geq \|p - q\|_2$, and otherwise we add $x_p + x_q \leq \|p - q\|_2 - \varepsilon$. These constraints resemble the definitions of (non-)edges in disk graphs provided that $\varepsilon > 0$ holds. We "reward" this by maximizing ε in the optimization function. One can now show:

▶ **Lemma 3.1 (★).** *$\mathcal{I}$ is a yes-instance of* ConScal *if and only if $\mathcal{P}(\mathcal{I})$ admits a solution with $\varepsilon > 0$.*

State-of-the art linear programming solvers are polynomial in the number of variables, constraints, *and* encoding size of the involved numbers [25, 32], where in $\mathcal{P}(\mathcal{I})$ the latter can be unbounded. However, it is known that linear programs with η variables and m constraints can be optimized in $2^{\mathcal{O}(\eta \cdot \log(\eta))} \cdot m$ time (in the Real RAM model) [2]. Since the linear program $\mathcal{P}(\mathcal{I})$ has $|\mathcal{T}| + 1$ variables and $\mathcal{O}(n^2)$ constraints, we get:

▶ **Proposition 3.2.** ConScal *can be solved in $2^{\mathcal{O}(|\mathcal{T}| \cdot \log(|\mathcal{T}|))} \cdot n^{\mathcal{O}(1)}$ time.*

[2] Fomin et al. [17] also used reduction to linear programs to determine suitable radii in their "min-variant".

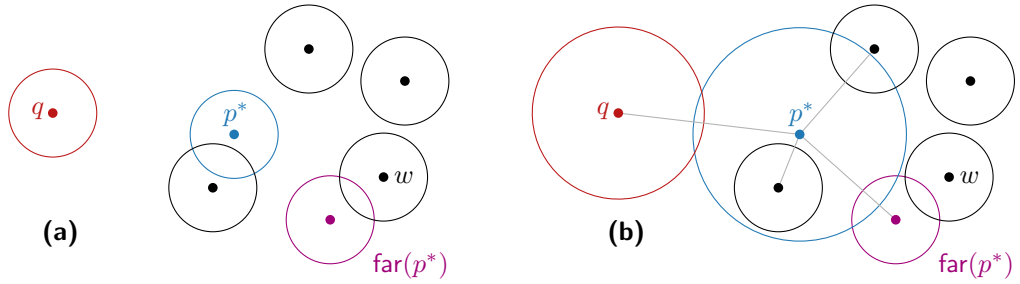


Figure 4 Let $\mathcal{T} = \{p^*, q\}$. The furthest unscaled neighbor $\mathsf{far}(p^*)$ (purple) of p^* (blue) determines if $p^*p' \in E(H)$ for $p' \in \mathcal{S} \setminus \mathcal{T}$. For the edge to q (red), we branch to fix if it exists (as here) or not. The disk graph **(a)** before and **(b)** after the scaling operations.

3.2 The XP-Membership Framework

We build upon Proposition 3.2 and show that Π-Scaling is in XP with respect to k for every graph class Π for which Π-Recognition can be solved in polynomial time.

The main idea is to provide a Turing-reduction to $n^{f(k)}$ instances of ConScal. To construct these instances, we exhaustively consider all possibilities to construct the set $\mathcal{T}$ of at most k scaled disks and apply the following steps separately for each constructed $\mathcal{T}$. Observe that if the radius of the modified disks would be fixed in the input, then the graph H that we obtain after scaling is solely determined by our choice of $\mathcal{T}$. However, since each modified disk can choose a radius from $[r_{\min}, r_{\max}]$, we must further branch to determine which target graph H we want to obtain. To this end, for each possible set $\mathcal{T}$, we determine for the scaled disks their (possible) neighborhood in H by guessing for each $p^* \in \mathcal{T}$ its furthest unscaled neighbor $\mathsf{far}(p^*)$, i.e., $\mathsf{far}(p^*) = \arg\max_{p \in (\mathcal{S} \setminus \mathcal{T}) \cap N_H(p^*)} \|p - p^*\|_2$ (if it exists). Observe that the neighborhood of p^* in the final disk graph cannot be arbitrary. In particular, as we increase $r(p^*)$, the set of (unscaled) neighbors of p^* can only increase. Thus, $\mathsf{far}(p^*)$ determines all unscaled neighbors of p^*: For $p' \in \mathcal{S} \setminus \mathcal{T}$ we have $p^*p' \in E(H)$ if and only if $\|p^* - p'\|_2 \leq \|p^* - \mathsf{far}(p^*)\|_2$; see Figure 4. Finally, we guess the adjacencies among scaled disks. Note that this captures all relevant, i.e., possible, graphs H we could obtain by scaling the disks in $\mathcal{T}$. In particular, each of the $2^{\mathcal{O}(k^2)} \cdot n^{\mathcal{O}(k)}$ branches corresponds to an instance of ConScal. We discard branches with $H \notin \Pi$, which can be checked efficiently since Π-Recognition can be solved in polynomial time.

▶ **Theorem 3.3** (★). *Let* Π *be a graph class such that* Π-Recognition *can be solved in polynomial time. Then,* Π-Scaling *can be solved in* $2^{\mathcal{O}(k^2)} \cdot n^{\mathcal{O}(k)}$ *time.*

Note that we only require the polynomial-time algorithm for Π-Recognition to efficiently verify that $H \in \Pi$ holds. Relaxing this restriction leads to different flavors of our meta-result:

▶ Remark 3.4 (★). Let Π be a graph class such that Π-Recognition is FPT or in XP with respect to some parameter κ. Then, Π-Scaling is in XP with respect to $k + \kappa$.

4 A Fixed-Parameter Tractable Algorithm for Scaling To Cluster

We now consider Π = cluster graphs and show that Scaling To Cluster is FPT parameterized by the budget k. This improves the XP-algorithm implied by Theorem 3.3 since cluster graphs can be recognized efficiently due to their characterization as being P_3-free.

Our algorithm follows the overall strategy of the general XP-approach behind Theorem 3.3, but uses structural properties of cluster graphs to obtain an FPT-running time. We provide in Figure 5 an overview of the algorithm. Let us fix an instance $\mathcal{I} = (\mathcal{S}, r_{\min}, r_{\max}, k)$ of Scaling To Cluster. Consider a hypothetical solution r. The algorithm has two main phases. In the first phase, we determine the disks in $\mathcal{T}(r)$. For each such disk $p \in \mathcal{T}(r)$, we also fix two, in some sense "extremal", unscaled disks that allow us to deduce the membership of p in clusters from $G(\mathcal{S}, r)$: We determine (a) the furthest *unscaled* disk $\mathsf{far}(p)$ which is in the *same* cluster as p in $G(\mathcal{S}, r)$, and (b) the closest *unscaled* disk $\mathsf{clo}(p)$ which is in a *different* cluster than p in $G(\mathcal{S}, r)$; see also Figure 5b–d. With this, we can almost completely fix the neighborhood of p in $G(\mathcal{S}, r)$ via FPT-many branches thanks to the structure of cluster graphs and the geometric aspect of the problem. In Phase 2, we resolve the (non-) edges between the at most k disks in $\mathcal{T}(r)$. Naïve branching would result in a overall running time of $2^{\mathcal{O}(k^2)} \cdot n^{\mathcal{O}(1)}$. We argue that for cluster graphs, this step can be done in $2^{\mathcal{O}(k \cdot \log(k))} \cdot n^{\mathcal{O}(1)}$ time. Finally, we employ the LP from Section 3 to check if suitable radii for the scaled disks exist. We now give an overview of the phases; see the full version [7] for details.

Intuition of Phase 1. In this phase, we aim to determine a set $\mathcal{T}$ of scaled disks and, for each disk $p \in \mathcal{T}$, the (a) furthest *unscaled* disk $\mathsf{far}(p)$ which is in the same cluster as p in the solution and the (b) closest *unscaled* disk $\mathsf{clo}(p)$ which is not in the same cluster as p in the solution. Initially, we compute in polynomial time a maximal vertex-disjoint P_3 packing $\mathcal{P}$ of $G(\mathcal{S}, \mathbb{1})$, that is, $\mathcal{P}$ is a set of vertex-disjoint P_3s of $G(\mathcal{S}, \mathbb{1})$ such that no further vertex-disjoint P_3 of $G(\mathcal{S}, \mathbb{1})$ can be added to $\mathcal{P}$. By P we denote the set of points in $\mathcal{S}$ whose corresponding disks are contained in at least one $P_3 \in \mathcal{P}$. Since every P_3 requires at least one scaling operation, we can safely output no if $|\mathcal{P}| \geq k + 1$. Moreover, $G' := G((\mathcal{S} \setminus P), \mathbb{1})$ is a cluster graph. Next, we create a complete edge-colored auxiliary graph H on the entire vertex set $\mathcal{S}$. The colors of H model the connectivity after some scaling operations and represent edges (blue), non-edges (red), and (still) undefined vertex-pairs (green) between two scaled disks that we resolve in Phase 2; see also Figure 5. More precisely, let $\mathcal{T}_i$ denote the set of the first i disks we identified for scaling (hence $\mathcal{T}_0 = \emptyset$). The graph H_i represents the connectivity of the disk graph resulting from $G(\mathcal{S}, \mathbb{1})$ if $\mathcal{T}_i$ is scaled accordingly; recall that we only compute the precise radii of the scaled disks at the end and now we only determine the final cluster graph H, which will represent the abstract intersection graph. The graph H_i is helpful to identify new P_3s after we have already scaled the first i disks $\mathcal{T}_i$. In particular, whenever H_i contains a *fully-defined* P_3 X, i.e., two edges (blue edges in H_i), one non-edge (red edge in H_i), and no undefined vertex-pair in $V(X)$ (green edge in H_i), we branch to determine which of the involved disks we (additionally) need to scale (and thus add to $\mathcal{T}_{i+1}$). Existing P_3s containing undefined (i.e., green) vertex-pairs will be treated at the end.

We determine $\mathsf{far}(p)$ and $\mathsf{clo}(p)$ via branching but exploit the structure of cluster graphs and the geometric aspect of the problem to bound the number of branches in k. To obtain $\mathsf{far}(p)$, let W be the k closest disks to p. Due to our budget k, at least one disk of W is unscaled. Let $\mathcal{C}$ be the set of at most k clusters of G' that contain the disks in W. If the cluster in the solution which contains p also contains disks of a cluster C of G' that "merges" into it, we must have $C \in \mathcal{C}$. There are three possibilities for the disk $\mathsf{far}(p)$: (a) $\mathsf{far}(p)$ does not exist, (b) $\mathsf{far}(p)$ is an unscaled disk from the P_3 packing $\mathcal{P}$, or (c) $\mathsf{far}(p)$ is among the k furthest disks in one of the clusters of $\mathcal{C}$ due to the budget k. This way, we can bound the overall number of choices for $\mathsf{far}(p)$ in $\mathcal{O}(k^2)$. Next, after we determined $\mathsf{far}(p)$, we branch on the k closest disks to p which are further away than $\mathsf{far}(p)$ from p to determine $\mathsf{clo}(p)$

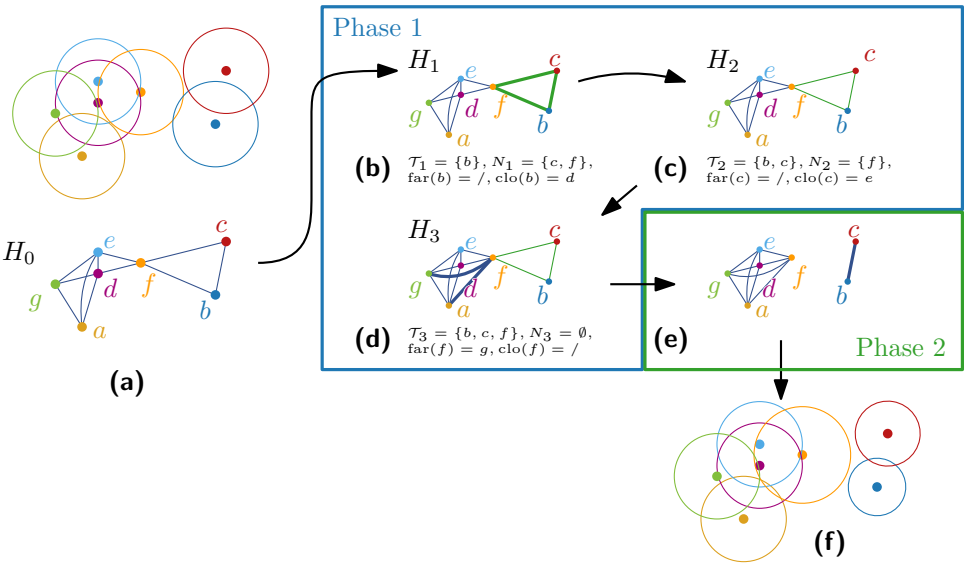


Figure 5 Overview of our FPT-algorithm for Scaling To Cluster with a small example. **(a)** An input instance and the corresponding edge-colored complete graph H_0 (we omit red edges for clarity). **(b)**–**(d)** In Phase 1, we iteratively identify disks to scale (the sets $\mathcal{T}_i$) and refine our edge-colored complete graph H_i (changes highlighted in bold). **(e)** In Phase 2, we branch to fix a compatible cluster graph H, and check using Proposition 3.2 if it can be realized. **(f)** The solution to **(a)**.

(we also consider the case that $\mathsf{clo}(p)$ does not exist). After fixing $\mathsf{far}(p)$ and $\mathsf{clo}(p)$, there may exist disks q with $\|\mathsf{far}(p) - p\|_2 < \|q - p\|_2 < \|\mathsf{clo}(p) - p\|_2$. Since $\mathsf{far}(p)$ and $\mathsf{clo}(p)$ is the furthest/closest *unscaled* disk which is in/not in the same cluster as p in the solution, each such disk q needs to be scaled as well; see Figure 5b–d. Note that due to our budget k, these can be at most k disks. We maintain a set N_i that keeps track of the disks that require scaling due to the above reasoning and update H accordingly. Whenever more than k scaling operations are required, i.e., $|\mathcal{T}_i \cup N_i| > k$, we abort the branch.

Intuition of Phase 2. Assume that the set $\mathcal{T}$ of scaled disks has been correctly determined. We now resolve the status of vertex-pairs where both endpoints are scaled disks, i.e., the undefined edges, in $2^{\mathcal{O}(k \cdot \log(k))} \cdot n^{\mathcal{O}(1)}$ time, and distinguish between scaled disks with $\mathsf{far}(p) \neq \emptyset$ and $\mathsf{far}(p) = \emptyset$. The latter implies that p is in a cluster with scaled disks only. All undefined vertex-pairs incident to disks with $\mathsf{far}(p) \neq \emptyset$ can be resolved in polynomial time. For the remaining scaled disks, we guess a clustering of them that also resolves all incident undefined vertex-pairs. Hence, the final auxiliary graph H contains no undefined vertex-pairs. If H does not correspond to a cluster graph, we can abort. Otherwise, we use the LP from Section 3.1 to check if suitable radii for the selected disks exist. Combining all, we obtain:

▶ **Theorem 4.1 (★).** Scaling To Cluster *can be solved in* $2^{\mathcal{O}(k \cdot \log(k))} \cdot n^{\mathcal{O}(1)}$ *time.*

5 Scaling to Cluster Graphs is NP-hard

We now justify the FPT-algorithm from Section 4 and show that Scaling To Cluster is NP-hard for every fixed rational $0 < r_{\min} \leq r_{\max}$ except $r_{\min} = r_{\max} = 1$. We provide two different reductions, depending on whether $r_{\min} < 1$ or $r_{\min} \geq 1$, that use the same gadget.

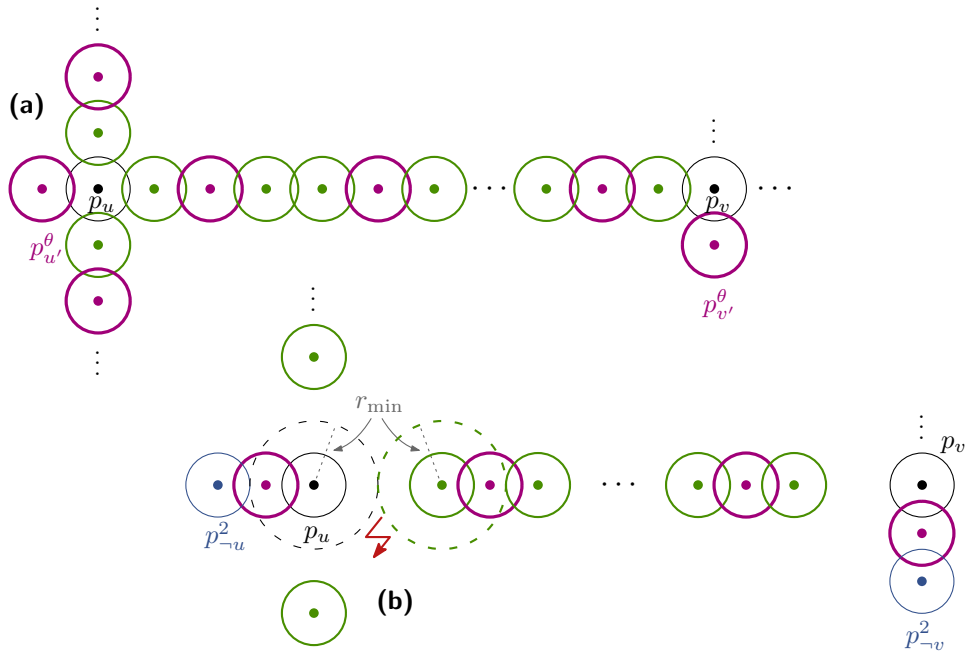


Figure 6 The edge uv for **(a)** $r_{\min} < 1$ and **(b)** $r_{\min} > 1$. Black, blue, green, and purple disks are 1-, 2-, η^2-, and θ-heavy, respectively. In (b), if we enlarge p_u and the left endpoint of a heavy P_3, then two θ-heavy disks are in the same connected component but not adjacent as $\theta > k$.

Heavy P_3s. For a point $p \in \mathbb{R}^2$ and an integer $\delta > 1$, we say that p is *δ-heavy*, denoted as p^δ, if it represents δ co-located *copies* $p^\delta(1), p^\delta(2), \ldots, p^\delta(\delta)$. We define $\mathcal{S}[p] \coloneqq \{p^\delta(i) : i \in [\delta]\}$ and also call the corresponding disks δ-heavy. A δ-heavy disk induces a δ-clique in the disk graph regardless of the radius assignment r, and we can assume that all copies receive equal radii in r, in particular, all are scaled or not scaled.[3] Our main gadget is a *δ-θ-heavy P_3* with $\theta > k \geq \delta$, or simply *heavy P_3* if δ and θ are clear from the context. It consists of three disks p_1, p_2, and p_3 with $\|p_1 - p_2\|_2 = \|p_2 - p_3\|_2 = \xi$, for some $1 < \xi \leq 2$, one of which is a θ-heavy disk, often p_2, and the remaining two are δ-heavy. This induces $\delta^2 \cdot \theta$ distinct P_3s:

▶ **Observation 5.1.** *Let $\mathcal{I}$ be an instance of* Scaling To Cluster*, let P be a δ-θ-heavy P_3 in $\mathcal{I}$, and r a solution. We have $|\mathcal{T}(r)| \geq \delta$ and $\mathcal{S}[p] \subseteq \mathcal{T}(r)$ for a δ-heavy disk $p \in V(P)$.*

We now provide an overview of the constructions; details are in the full version [7].

Hardness for Shrinking Disks ($0 < r_{\min} < 1$). The reduction is inspired by ideas from Fomin et al. [17]. We fix a rational value $0 < r_{\min} < 1$; the instance that we construct will later be scaled by a factor depending on $r_{\min}$. Let $\mathcal{I}_{\text{VC}} = (H, \kappa)$ be an instance of Vertex Cover where H is a planar cubic graph with $|V(H)| = \eta$; Vertex Cover remains NP-hard in this case [21]. We compute in polynomial time a planar rectilinear embedding Γ of H of polynomial area [30] and scale it by an integer $\gamma > 1$ that depends on $r_{\min}$. Similar to Fomin et al. [17], we represent edges of H by evenly spaced disks such that the resulting unit disk graph encodes a suitable subdivision of H; see Figure 6a. Note that we can slightly compress the distance between adjacent disks to accommodate an additional disk when

[3] For this it also suffices to place all copies in a sufficiently small region around p to avoid exact co-location.

required. Every vertex $v \in V(H)$ is represented by a disk p_v, and v will be part of a vertex cover C of H if and only if $r(p_v) \neq 1$. To ensure that every edge $uv \in E(H)$ is covered, we turn the disks along uv into a series of interleaving η^2-θ-heavy P_3s. Since H is cubic, we can place a further θ-heavy disk $p^\theta_{v'}$ next to p_v, for every $v \in V(H)$, that does not interfere with the edges incident to v; see Figure 6a. This makes p_v part of four θ-heavy P_3s. If we do not scale p_v, these P_3s, together with our budget k specified below, force the disks p_u for all vertices $u \in N_H(v)$ to be scaled (otherwise a P_3 remains along uv). We set $k = k_{\text{fix}} + \kappa$, where k_{fix} is the number of enforced scaling operations to remove all heavy P_3s along edges of H, and $\theta = k + 1$. For the value of $r_{\max}$, we observe that removing a P_3 can only be done via shrinking some disks, since initially multiple θ-heavy disks are in the same connected component; recall $\theta > k$. Thus, we can set any fixed rational $r_{\max} \geq r_{\min}$ in the construction.

▶ **Proposition 5.2 (★).** Scaling To Cluster *is* NP*-hard for all fixed rational* $0 < r_{\min} < 1$.

Hardness for Enlarging Disks ($1 \leq r_{\min}$). We fix a rational value $1 < r_{\min}$ (the case $r_{\min} = 1$ is handled in the end). Let $\mathcal{I}_{\text{IS}} = (H, \kappa)$ be an instance of Independent Set, where H is a planar cubic graph with $|V(H)| = \eta$; also Independent Set remains NP-hard in this case [21]. We start with an appropriately scaled planar rectilinear embedding Γ^γ of H, where the scaling factor $\gamma > 1$ depends on $r_{\min}$. To represent an edge $uv \in E(H)$, we use a series of disjoint η^2-θ-heavy P_3s. The distance between endpoints of each heavy P_3 is $2\xi = \min(r_{\min}, 1.5) + 1$ and the spacing d between consecutive heavy P_3s along uv satisfies $1 + r_{\min} < d \leq 2r_{\min}$. We also maintain a distance of $2r_{\min}$ to the start- and endpoint of uv; see Figure 6b. By Observation 5.1, we must scale all copies of one endpoint of each heavy P_3. Since $d \leq 2r_{\min}$, we must consistently scale either the left or the right endpoint of each heavy P_3 along uv. Otherwise, two θ-heavy disks belong to the same cluster but do not intersect; see the dashed disks in Figure 6b. As $\theta > k$, this cannot lead to a solution. We represent each vertex $v \in V(H)$ by an extra disk p_v. Since H is cubic, there is a free "direction" next to v in Γ^γ. There, we place an additional θ-heavy disk and a 2-heavy disk $p^2_{\neg v}$ that together with p_v form a heavy P_3; see Figure 6b. To remove this P_3, we have $p_v \in \mathcal{T}(r)$ or $\mathcal{S}[p_{\neg v}] \subseteq \mathcal{T}(r)$, encoding the choice of having v in an independent set C of H, i.e., $v \in C$ if and only if $p_v \in \mathcal{T}(r)$. Scaling p_u and p_v for an edge $uv \in E(H)$ results in "merging" two disjoint θ-heavy disks into one connected component, which cannot lead to a solution.

To complete the construction, we set $k = k_{\text{fix}} + \kappa + 2(\eta - \kappa) = k_{\text{fix}} + 2\eta - \kappa$ and $\theta = k + 1$. The value of k_{fix} denotes the number of scaling operations required to remove all heavy P_3s along the edges of H. Moreover, the term $2(\eta - \kappa)$ accounts for the $\eta - \kappa$ vertices $v \notin C$ for which we have to scale (both copies of) $p^2_{\neg v}$. Observe that enlarging a disk beyond $r_{\min}$ is never helpful, as we eventually connect two θ-heavy disks which cannot lead to a solution; recall $\theta > k$. Therefore, we can use any fixed rational $r_{\max} \geq r_{\min}$ in the above construction.

▶ **Proposition 5.3 (★).** Scaling To Cluster *is* NP*-hard for all fixed rational* $1 < r_{\min}$.

Combining Both Constructions. Propositions 5.2 and 5.3 together imply NP-hardness of Scaling To Cluster for every fixed rational $r_{\min}$ with $0 < r_{\min} < 1$ or $1 < r_{\min}$. We give a simplified example in Figure 7. For $r_{\min} = 1$ and $r_{\max} > 1$, we can modify the P_3s' placement in the construction $r_{\min} > 1$. For $r_{\min} = r_{\max} = 1$, we simply need to check if $G(\mathcal{S}, \mathbb{1})$ is a cluster graph, which can be done in polynomial time due to their characterization.

▶ **Theorem 5.4.** Scaling To Cluster *is* NP*-hard for every fixed rationals* $0 < r_{\min} \leq r_{\max}$ *except for the case* $r_{\min} = r_{\max} = 1$ *where it can be solved in polynomial time.*

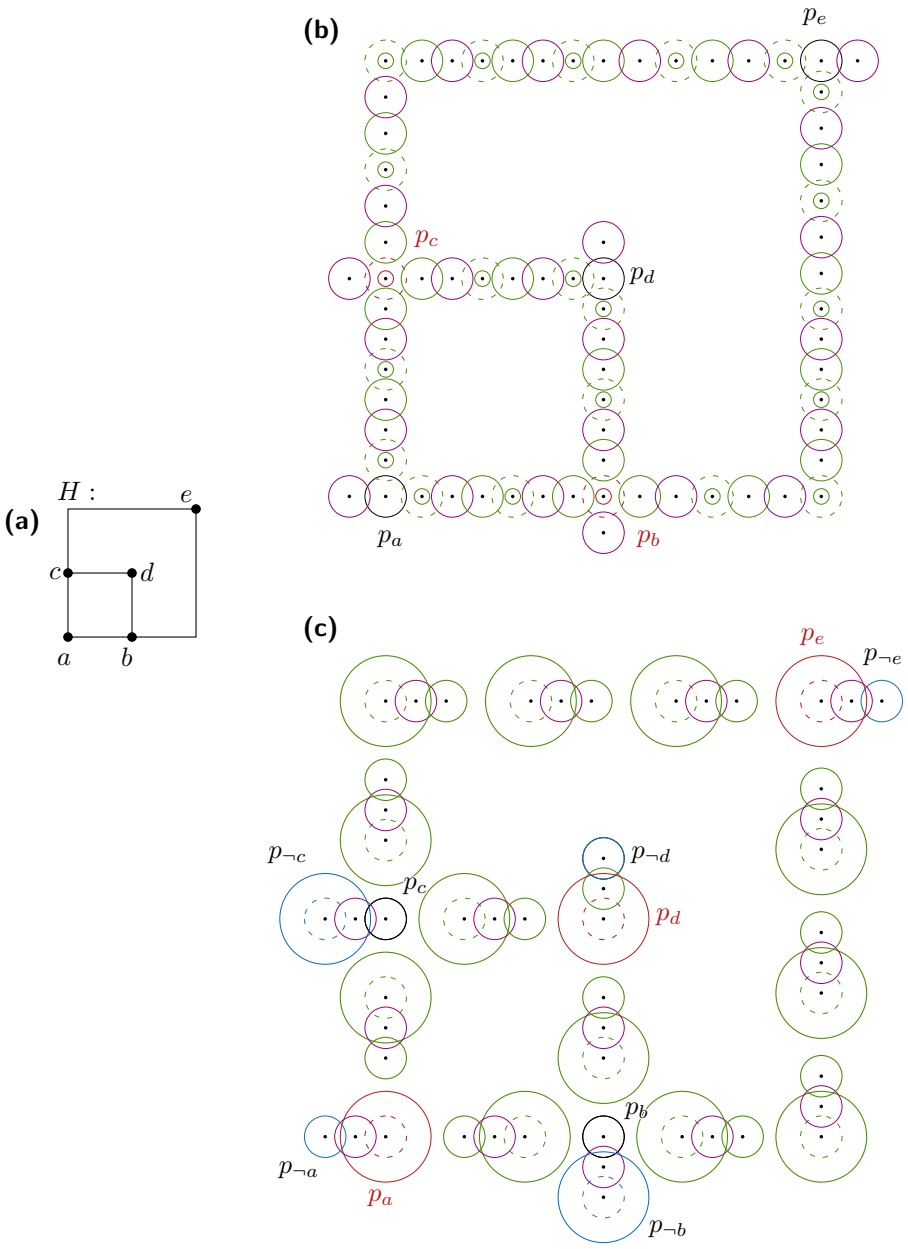


Figure 7 **(a)** A graph H and the corresponding instances of Scaling To Cluster for **(b)** $r_{\min} < 1$ and **(c)** $r_{\min} > 1$. If a disk representing a vertex is red, then the vertex belongs to a solution to the instance **(b)** $(H, 2)$ of Vertex Cover and **(c)** $(H, 3)$ of Independent Set. Note that we can always slightly shift some disks to make space along an edge of H if required. We simplified H and the constructed instances to ease readability. Colors have the same meaning as in Figure 6.

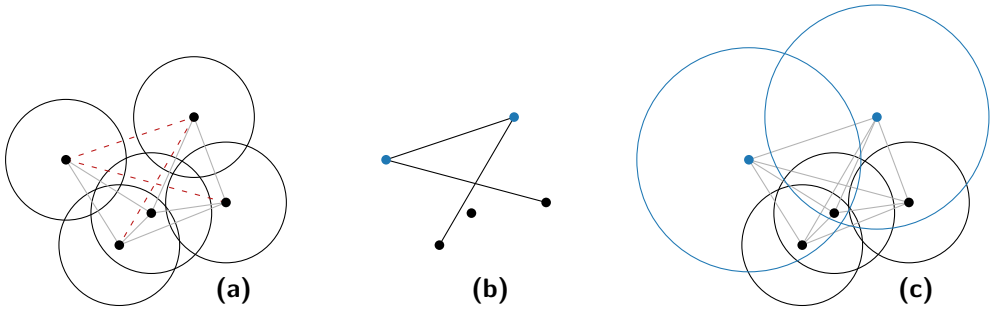


Figure 8 To turn **(a)** into a complete graph, we must scale at least one endpoint of every dashed non-edge. **(b)** A vertex cover (blue) of the complement graph **(c)** directly translates into a solution.

6 Scaling To Complete Graphs in Polynomial Time

We now turn our attention to complete graphs. Let us fix an instance $\mathcal{I} = (\mathcal{S}, r_{\min}, r_{\max}, k)$ of Scaling To Complete and consider a hypothetical solution r. Since additional edges can only be created by enlarging disks, we can assume without loss of generality that $r_{\max} > 1$ (otherwise, it suffices to test if $G(\mathcal{S}, \mathbb{1})$ is already complete) and that $r(p) = r_{\max}$ for all $p \in \mathcal{T}(r)$. Observe that for any pair $p, q \in \mathcal{S}$ with $pq \notin E(G(\mathcal{S}, \mathbb{1}))$, at least one of p and q must be in $\mathcal{T}(r)$. We now partition the set $E' = \binom{\mathcal{S}}{2}$ of all point pairs based on their distance:

- $E'_0 := \{pq \in E' : \|p - q\|_2 \leq 2\}$, i.e., pairs that already form an edge in $G(\mathcal{S}, \mathbb{1})$;
- $E'_1 := \{pq \in E' : 2 < \|p - q\|_2 \leq 1 + r_{\max}\}$, i.e., pairs where scaling one disk suffices;
- $E'_2 := \{pq \in E' : 1 + r_{\max} < \|p - q\|_2 \leq 2r_{\max}\}$, i.e., pairs where we must scale both disks;
- $E'_\perp := \{pq \in E' : 2r_{\max} < \|p - q\|_2\}$, i.e., pairs that can never intersect.

Let $\mathcal{S}[E'_i] := \bigcup_{pq \in E'_i} \{p, q\}$ for $i = 0, 1, 2$. Furthermore, define $\mathcal{T}' := \mathcal{S}[E'_2]$. We observe:

▶ **Observation 6.1.** *It holds $\mathcal{T}' \subseteq \mathcal{T}(r)$. Also, $\mathcal{I}$ is a no-instance if $E'_\perp \neq \emptyset$ or $|\mathcal{T}'| > k$.*

While scaling all disks in $\mathcal{T}'$ is necessary, it is not yet sufficient. In particular, for $pq \in E'_1$ with $p, q \notin \mathcal{T}'$, the disks do not form an edge in $G(\mathcal{S}, r)$ unless we also scale p or q. Let $\mathcal{X} := \mathcal{S}[E'_1] \setminus \mathcal{T}'$ and $n_\mathcal{X} := |\mathcal{X}|$. Consider the unit disk graph $G(\mathcal{X}, \mathbb{1})$. Every non-edge $pq \notin E(G(\mathcal{X}, \mathbb{1}))$ corresponds to one such pair $pq \in E'_1$ where we must still scale p or q. Hence, we must select (at least) one endpoint from every non-edge in $G(\mathcal{X}, \mathbb{1})$, which corresponds to a vertex cover (of size at most $k' := k - |\mathcal{T}'|$) in the complement graph $\overline{G(\mathcal{X}, \mathbb{1})}$. Computing this vertex cover is equivalent to finding a clique (of size $n_\mathcal{X} - k'$) in $G(\mathcal{X}, \mathbb{1})$.[4]

▶ **Lemma 6.2 (★).** *$\mathcal{I}$ is a yes-instance if and only if $G(\mathcal{X}, \mathbb{1})$ has a clique of size $n_\mathcal{X} - k'$.*

Using Lemma 6.2, we can find the remaining k' disks to scale via a sufficiently large clique in $G(\mathcal{X}, \mathbb{1})$. Computing a maximum clique in a unit disk graph with given geometric representation takes $\mathcal{O}(n^{2.5} \log n)$ time [14]. All other steps take $\mathcal{O}(n^2)$ time.

▶ **Theorem 6.3.** Scaling To Complete *can be solved in $\mathcal{O}(n^{2.5} \log n)$ time.*

[4] Although slower, it would also suffice to find a clique of size $n - k'$ in the disk graph $G(\mathcal{S}, r')$ with $r'(p) = r_{\max}$ for $p \in \mathcal{T}'$ and $r'(p) = 1$ otherwise, which we can do in $\mathcal{O}(n^{6.5})$ time [23, 27].

7 Scaling to Connected Graphs is W[1]-hard

We now consider connected graphs and show that Scaling To Connected is W[1]-hard when parameterized by the number k of allowed scaling operations. This rules out an FPT-algorithm, and, in particular, generalizes the W[1]-hardness by Fomin et al. [17]: Their hardness holds in the variant where only disks from a given set $\mathcal{A} \subseteq \mathcal{S}$ can be enlarged, i.e., $\mathcal{T}(r) \subseteq \mathcal{A}$ holds. In contrast, our reduction allows any disk to be scaled, even to a fixed radius $r_{\max}$, which prevents a straightforward adaptation of Fomin et al.'s proof and instead requires a new construction. We reduce from the variant of Grid Tiling introduced below.

Let Grid Tiling With $\oplus$ be defined as follows. Given two integers $\eta, \kappa \geq 1$, and a family $\mathcal{X}$ of κ^2 sets $X_{i,j} \subseteq [\eta]^2$, choose a tuple $x_{i,j} = (a, b) \in X_{i,j}$ for each $i, j \in [\kappa]$ such that if $x_{i,j} = (a, b)$ and $x_{i+1,j} = (a', b')$ then $a \oplus a'$, and if $x_{i,j+1} = (a'', b'')$ then $b \oplus b''$ for all (possible) $i, j \in [\kappa]$, i.e., the selected tuples fulfill the relation $\oplus$ on the first and second value, respectively. We call $X_{i,j}$ the *tile* of the ith *column* and jth *row*; the deviation of the usual indexation facilitates the following description and is common for geometric reductions [5]. Two tiles $X_{i,j}$ and $X_{i',j'}$ are *adjacent* if and only if either $|i' - i| = 1$ or $|j' - j| = 1$ (but not both). Textbook-variants use $\oplus \in \{=, \leq\}$ and are known to be W[1]-hard parameterized by κ [5]. One can also extend the hardness to the case where we use $>$ for $\oplus$.

▶ **Lemma 7.1 (★).** Grid Tiling With $>$ *parameterized by* κ *is* W[1]*-hard.*

7.1 An Overview of the Construction

Let $\mathcal{I}_> = (\eta, \kappa, \mathcal{X})$ be an instance of Grid Tiling With $>$. The constructed instance $\mathcal{I}$ of Scaling To Connected closely resembles the (imagined) layout of $\mathcal{I}_>$, i.e., contains a $\kappa \times \kappa$-grid of tiles, each containing a subset of the $\eta \times \eta$ grid. To facilitate the following presentation, we rescale without loss of generality the plane such that one unit equals to 0.1. Our instance is composed of several gadgets of three different kinds. The first gadget represents a tile $X_{i,j} \in \mathcal{X}$ and contains a disk $p_{i,j}^{a,b}$ for every $(a, b) \in X_{i,j}$. Scaling the disk $p_{i,j}^{a,b}$ corresponds to choosing the tuple $x_{i,j} = (a, b)$. The second gadget consists of a series of disks to ensure that the scaled disks, i.e., selected tuples, fulfill the $>$-constraints along the rows and columns of $\mathcal{I}_>$. Finally, the third gadget is a set of dummy disks that enforce that we scale one disk per tile. To preserve the equivalence between solutions, we ensure that our gadgets fulfill certain properties that we use in Section 7.2 to establish correctness.

Encoding the Tiles. Let $X_{i,j} \in \mathcal{X}$ be a tile. We introduce one disk $p_{i,j}^{a,b}$ for every tuple $(a, b) \in X_{i,j}$ and set $x(p_{i,j}^{a,b}) = i(2\eta + 2\gamma) + a$ and $y(p_{i,j}^{a,b}) = j(2\eta + 2\gamma) + b$, where γ is a *displacement-factor* that we specify in the end. Observe that we arrange the disks for each tile along an $\eta \times \eta$-integer grid and the tiles along a $\kappa \times \kappa$-grid. We visualize this base layout in Figure 9 and provide further details in the full version [7]. Let $\mathcal{S}[X_{i,j}]$ denote the disks representing the tuples of the tile $X_{i,j}$, $i, j \in [\kappa]$. We observe that $G(\mathcal{S}[X_{i,j}], \mathbb{1})$ is edge-less; recall that one unit equals 0.1. To guarantee that our intended equivalence between the solutions is well-defined, we ensure that our final instance $\mathcal{I}$ has the following property.

▶ **Property 7.2.** *Let r be a solution to $\mathcal{I}$. We have $|\mathcal{S}[X_{i,j}] \cap \mathcal{T}(r)| = 1$ for every $i, j \in [\kappa]$.*

Moreover, to represent the (non-)adherence to the $>$-constraints, we ensure that two scaled disks $p_{i,j}^{a,b}$ and $p_{i',j'}^{a',b'}$ from different tiles intersect if and only if the corresponding tiles $X_{i,j}$ and $X_{i',j'}$ are adjacent and $a > a'$ (if $i' = i + 1$) or $b > b'$ (if $j' = j + 1$) holds:

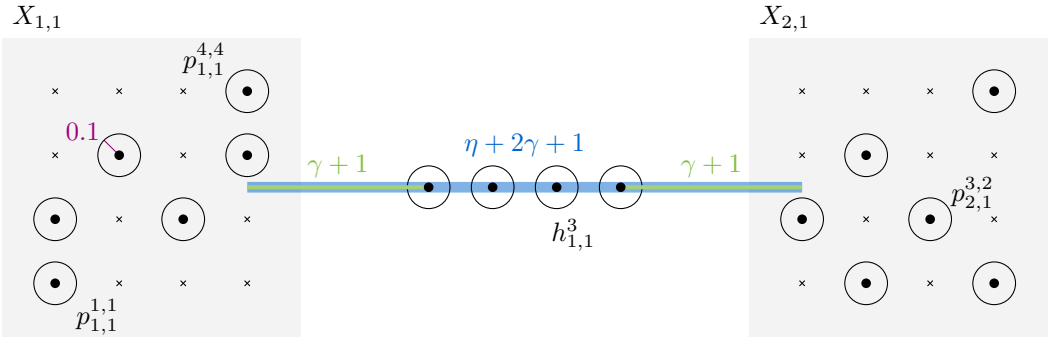


Figure 9 Gadgets encoding the tiles and constraints. The disks represent the elements of $X_{1,1}$ and $X_{2,1}$, and crosses indicate disk positions for elements not present in the respective tiles.

▶ **Property 7.3.** *Let $p_{i,j}^{a,b}, p_{i',j'}^{a',b'} \in \mathcal{S}$ be disks and r a solution to $\mathcal{I}$ with $r(p_{i,j}^{a,b}) = r_{\max}$. It holds:*

1. *if $i' = i$, $j' = j$, then the disks intersect;*
2. *if $i' = i+1$, $j' = j$, then the disks intersect if and only if $p_{i',j'}^{a',b'} \in \mathcal{T}(r)$ and $a > a'$;*
3. *if $i' = i-1$, $j' = j$, then the disks intersect if and only if $p_{i',j'}^{a',b'} \in \mathcal{T}(r)$ and $a' > a$;*
4. *if $i' = i$, $j' = j+1$, then the disks intersect if and only if $p_{i',j'}^{a',b'} \in \mathcal{T}(r)$ and $b > b'$;*
5. *if $i' = i$, $j' = j-1$, then the disks intersect if and only if $p_{i',j'}^{a',b'} \in \mathcal{T}(r)$ and $b' > b$; and*
6. *if the tiles $X_{i,j}$ and $X_{i',j'}$ are not adjacent, then the disks do not intersect.*

Towards establishing Properties 7.2 and 7.3, observe that the distance between any pair $p_{i,j}^{a,b}, p_{i,j}^{a',b'} \in \mathcal{S}[X_{i,j}]$ is at most $\sqrt{2}\eta$. Hence, we ensure $r_{\max} \geq 2\eta$. The distance between same-valued tuples in adjacent tiles is $2\eta + 2\gamma$, and the corresponding disks should never intersect. Thus, we ensure $2r_{\max} < 2\eta + 2\gamma$. This alone does not guarantee that $\mathcal{I}$ admits only solutions r that correspond to ones of $\mathcal{I}_>$. In particular, $G(\mathcal{S}, r)$ could be connected although two scaled disks from adjacent tiles do not intersect. We now prevent this.

Ensuring the $>$-Constraints. Let $X_{i,j}$ and $X_{i+1,j}$, $i \in [\kappa - 1], j \in [\kappa]$ be two adjacent tiles. We introduce η disks $h_{i,j}^1, h_{i,j}^2, \ldots, h_{i,j}^\eta$ between the (disks for the) tiles $X_{i,j}$ and $X_{i+1,j}$. These disks are evenly spaced, maintaining a distance of at least $\gamma + 1$ to the disks in the two tiles; see Figure 9 and the full version [7] for a formal construction. We introduce analogous disks $v_{i,j}^1, \ldots, v_{i,j}^\eta$ between the adjacent tiles $X_{i,j}$ and $X_{i,j+1}$, $i \in [\kappa], j \in [\kappa - 1]$. Let $\mathcal{S}[>]$ be the above-introduced disks. Using the disks $\mathcal{S}[>]$, we aim to fulfill the following property.

▶ **Property 7.4.** *Let $p_{i,j}^{a,b}$ be a disk and let r be a solution to $\mathcal{I}$ such that $r(p_{i,j}^{a,b}) = r_{\max}$ and $\mathcal{S}[>] \cap \mathcal{T}(r) = \emptyset$. For every $q, t \in [\eta]$, it holds:*

1. *if $1 \leq i < \kappa$, then $p_{i,j}^{a,b}$ and $h_{i,j}^q$ intersect if and only if $q < a$;*
2. *if $1 < i \leq \kappa$ then $p_{i,j}^{a,b}$ and $h_{i-1,j}^q$ intersect if and only if $q > a$;*
3. *if $1 \leq j < \kappa$, then $p_{i,j}^{a,b}$ and $v_{i,j}^t$ intersect if and only if $t < b$; and*
4. *if $1 < j \leq \kappa$, then $p_{i,j}^{a,b}$ and $v_{i,j-1}^t$ intersect if and only if $t > b$.*

Assume for a moment $p_{i,j}^{a,b}, p_{i+1,j}^{a',b'} \in \mathcal{T}(r)$ with $a \leq a'$. Neither of $p_{i,j}^{a,b}$ and $p_{i+1,j}^{a',b'}$ intersect $h_{i,j}^q$ for $a \leq q \leq a'$ by Property 7.4. Combined with Property 7.2 and an appropriately-chosen budget k, these $h_{i,j}^q$ witness that $G(\mathcal{S}, r)$ is not connected. Observe that $x_{i,j} = (a, b)$ and $x_{i+1,j} = (a', b')$ with $a \leq a'$ can also not be part of a solution to $\mathcal{I}_>$. Fulfilling Property 7.4 heavily depends on our values for γ, i.e., the displacement-factor, and $r_{\max}$. In particular,

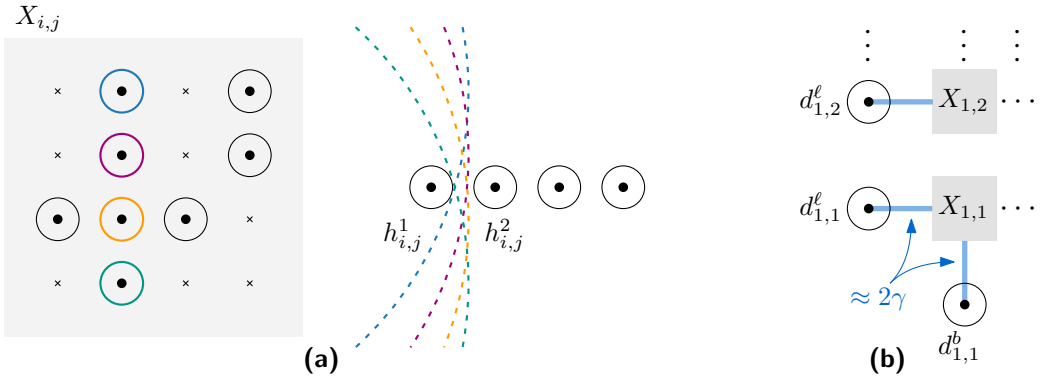


Figure 10 (a) We must ensure Property 7.4 no matter which disk $p^{a,b}_{i,j}$ we scale, indicated by color, i.e., independent of the value of a and b. **(b)** The placement of the dummy disks.

we must select γ such that the property is fulfilled no matter which disks $p^{a,b}_{i,j}$ we choose to scale, i.e., we must compensate the possible value of a and b; see also Figure 10a. Since $a, b \in [\eta]$, we can set $\gamma = \eta^2$. Using geometry, we can compute the value for $r_{\max}$.

The Dummy Disks. The above-introduced additional disks make establishing Property 7.2 difficult. In particular, we must ensure that no disk $p \in \mathcal{S}[>]$ is scaled. We introduce 4κ *dummy disks* $d_{i,j}$, (at least) one for every tile $X_{i,j}$ with $i \in \{1, \kappa\}$ or $j \in \{1, \kappa\}$. They are placed around the overall $\kappa \times \kappa$-grid, maintaining a distance of (roughly) 2γ to the base layout; see Figure 10b and the full version [7]. Due to their placement, we have the following.

▶ **Property 7.5.** *Let r be a solution to $\mathcal{I}$, and let $d_{i,j}$ be a dummy disk. We have $d_{i,j}, p^{a,b}_{i,j} \in \mathcal{T}(r)$ for some tuple $(a, b) \in X_{i,j}$.*

Property 7.5 enforces us to scale each of the 4κ dummy disks. Together with our intended semantic (recall also Property 7.2), we thus set $k = 4\kappa + \kappa^2$. We can set $r_{\min} = 0.1$.

7.2 Combining the Gadgets: Correctness of the Reduction

Figure 11 shows a small example of our reduction. We now show that an above-constructed instance $\mathcal{I}$ fulfills our properties, which we later use to argue correctness of the reduction.

▶ **Lemma 7.6 (★).** *Our instance $\mathcal{I}$ of* Scaling To Connected *fulfills Properties 7.2–7.5.*

We use Lemma 7.6 in the (more involved) backward direction of the correctness argument of the reduction. In particular, Properties 7.2 and 7.5 ensures that a solution to $\mathcal{I}_>$ is well-defined, and Properties 7.3 and 7.4 ensure that the tuples corresponding to the scaled disks adhere to the $>$-constraints; note that we can assume without loss of generality that $r(p) = r_{\max}$ holds in a solution r for every disk $p \in \mathcal{T}(r)$ since connectivity is preserved under enlarging disks.

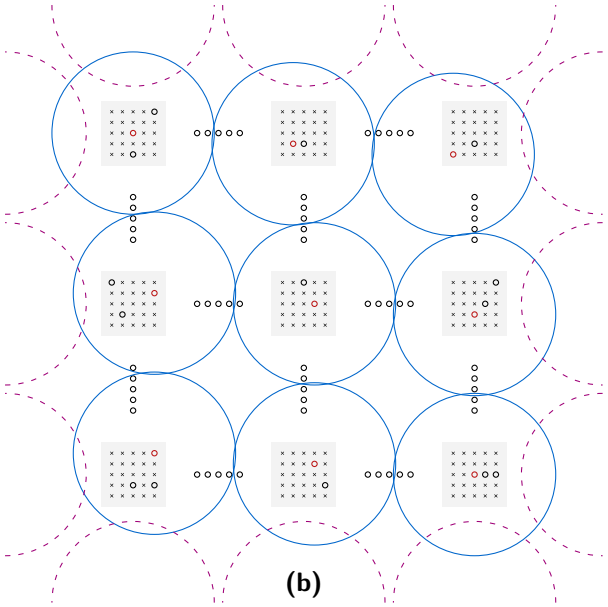
▶ **Theorem 7.7 (★).** Scaling To Connected *is* W[1]*-hard when parameterized by k.*

8 Generalizations and Concluding Remarks

We contributed to the study of geometric-aware modification operations for intersection graphs, focusing on disk-scaling in unit disk graphs where each modified disk can choose a radius from $[r_{\min}, r_{\max}]$. Our work opens up several promising avenues for future research:

$X_{1,3}$: $(5,5)$ $(3,1)$ $(3,3)$	$X_{2,3}$: $(2,2)$ $(3,2)$	$X_{3,3}$: $(1,1)$ $(3,2)$
$X_{1,2}$: $(1,5)$ $(5,4)$ $(1,2)$	$X_{2,2}$: $(3,5)$ $(4,3)$	$X_{3,2}$: $(5,5)$ $(4,2)$ $(3,2)$
$X_{1,1}$: $(5,5)$ $(3,2)$ $(5,2)$	$X_{2,1}$ $(4,4)$ $(5,2)$	$X_{3,1}$: $(5,3)$ $(4,3)$ $(3,3)$

(a)



(b)

Figure 11 (a) An instance $(5, 3, \mathcal{X})$ of Grid Tiling With $>$ and **(b)** the corresponding instance of Scaling To Connected. Small disks and crosses in the tiles have the same meaning as in Figure 9. We highlight a solution in red, the corresponding enlarged disks in blue, and the scaled dummy disks with purple dashed lines.

Fomin et al. [17] also studied an optimization variant where we scale (at most) k disks and minimize $\sum_{p\in\mathcal{S}} |1 - r(p)|$. Generalizing our algorithmic results to this *min*-variant is non-trivial and remains open. In particular, we cannot simply adapt the optimization function of the LP due to our (maximized) "slack-variable" ε (and only determining the disks to scale is not sufficient since disks might require different radii). It is also open whether Scaling To Cluster admits a polynomial kernel for k or is in NP. Finally, it would be interesting to combine scaling with other modification operations, such as disk dispersion [16].

References

1 Faisal N. Abu-Khzam, Judith Egan, Serge Gaspers, Alexis Shaw, and Peter Shaw. Cluster editing with vertex splitting. *Discrete Applied Mathematics*, 371:185–195, 2025. doi:10.1016/j.dam.2025.04.013.

2 Timothy M. Chan. Improved deterministic algorithms for linear programming in low dimensions. *ACM Transactions on Algorithms (TALG)*, 14(3):30:1–30:10, 2018. doi:10.1145/3155312.

3 Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165–177, 1990. doi:10.1016/0012-365X(90)90358-O.

4 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.

5 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

6 Mark de Berg, Hans L. Bodlaender, Sándor Kisfaludi-Bak, Dániel Marx, and Tom C. van der Zanden. A framework for exponential-time-hypothesis-tight algorithms and lower bounds in geometric intersection graphs. *SIAM Journal on Computing*, 49(6):1291–1331, 2020. doi:10.1137/20M1320870.

7 Thomas Depian and Frank Sommer. Revisiting graph modification via disk scaling: From one radius to interval-based radii, 2026. doi:10.48550/arXiv.2603.05358.

8 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate Texts in Mathematics*. Springer, 2012.

9 Nicolás Honorato Droguett, Kazuhiro Kurita, Tesshu Hanaka, and Hirotaka Ono. Algorithms for optimally shifting intervals under intersection graph models. In Bo Li, Minming Li, and Xiaoming Sun, editors, *Proc. 18th International Joint Conference on Frontiers in Algorithmics (IJTCS-FAW'24)*, volume 14752 of *Lecture Notes in Computer Science*, pages 66–78. Springer, 2024. doi:10.1007/978-981-97-7752-5_5.

10 Nicolás Honorato Droguett, Kazuhiro Kurita, Tesshu Hanaka, and Hirotaka Ono. On the complexity of minimising the moving distance for dispersing objects. In Pat Morin and Eunjin Oh, editors, *Proc. 19th International Symposium on Algorithms and Data Structures (WADS'25)*, volume 349 of *LIPIcs*, pages 36:1–36:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.WADS.2025.36.

11 Nicolás Honorato Droguett, Kazuhiro Kurita, Tesshu Hanaka, Hirotaka Ono, and Alexander Wolff. Further results on rendering geometric intersection graphs sparse by dispersion. In Emilio Di Giacomo and Debajyoti Mondal, editors, *Proc. 20th International Conference and Workshops on Algorithms and Computation (WALCOM'26)*, volume 16444 of *Lecture Notes in Computer Science*, pages 451–466. Springer, 2025. doi:10.1007/978-981-95-7127-7_30.

12 David Eppstein. Graph-theoretic solutions to computational geometry problems. In Christophe Paul and Michel Habib, editors, *Proc. 35th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'09)*, volume 5911 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2009. doi:10.1007/978-3-642-11409-0_1.

13 Jeff Erickson, Ivor van der Hoog, and Tillmann Miltzow. Smoothing the gap between NP and ER. *SIAM Journal on Computing*, 53(6):S20–102, 2024. doi:10.1137/20M1385287.

14 Jared Espenant, J. Mark Keil, and Debajyoti Mondal. Finding a maximum clique in a disk graph. In Erin W. Chambers and Joachim Gudmundsson, editors, *Proc. 39th International Symposium on Computational Geometry (SoCG'23)*, volume 258 of *LIPIcs*, pages 30:1–30:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.SOCG.2023.30.

15 Aleksei V. Fishkin. Disk graphs: A short survey. In Klaus Jansen and Roberto Solis-Oba, editors, *Proc. 1st Workshop on Approximation and Online Algorithms (WAOA'03)*, volume 2909 of *Lecture Notes in Computer Science*, pages 260–264. Springer, 2003. doi:10.1007/978-3-540-24592-6_23.

16 Fedor V. Fomin, Petr A. Golovach, Tanmay Inamdar, Saket Saurabh, and Meirav Zehavi. Kernelization for spreading points. In Inge Li Gørtz, Martin Farach-Colton, Simon J. Puglisi, and Grzegorz Herman, editors, *Proc. 31st European Symposium on Algorithms (ESA'23)*, volume 274 of *LIPIcs*, pages 48:1–48:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ESA.2023.48.

17 Fedor V. Fomin, Petr A. Golovach, Tanmay Inamdar, Saket Saurabh, and Meirav Zehavi. Parameterized geometric graph modification with disk scaling. In Raghu Meka, editor, *Proc. 16th Innovations in Theoretical Computer Science (ITCS'25)*, volume 325 of *LIPIcs*, pages 51:1–51:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.ITCS.2025.51.

18 Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. Finding, hitting and packing cycles in subexponential time on unit disk graphs. *Discrete & Computational Geometry*, 62(4):879–911, 2019. doi:10.1007/S00454-018-00054-X.

19 Fedor V. Fomin, Daniel Lokshtanov, and Saket Saurabh. Bidimensionality and geometric graphs. In Yuval Rabani, editor, *Proc. 23rd ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*, pages 1563–1575. SIAM, 2012. doi:10.1137/1.9781611973099.124.

20 Fedor V Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019. doi:10.1017/9781107415157.

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

22 Petr Hlinený and Jan Kratochvíl. Representing graphs by disks and balls (a survey of recognition-complexity results). *Discrete Mathematics*, 229(1-3):101–124, 2001. doi:10.1016/S0012-365X(00)00204-1.

23 John E. Hopcroft and Richard M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973. doi:10.1137/0202019.

24 M.L. Huson and A. Sen. Broadcast scheduling algorithms for radio networks. In *Proceedings of the Military Communications Conference (MILCOM '95)*, volume 2, pages 647–651. IEEE, 1995. doi:10.1109/MILCOM.1995.483546.

25 Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. A faster algorithm for solving general LPs. In Samir Khuller and Virginia Vassilevska Williams, editors, *Proc. 53rd Symposium on the Theory of Computing (STOC'21)*, pages 823–832. ACM, 2021. doi:10.1145/3406325.3451058.

26 Richard M. Karp. Reducibility Among Combinatorial Problems. In Raymond E. Miller and James W. Thatcher, editors, *Proc. Computer Science Conferences & Workshops*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972. doi:10.1007/978-1-4684-2001-2_9.

27 J. Mark Keil and Debajyoti Mondal. The maximum clique problem in a disk graph made easy. In Oswin Aichholzer and Haitao Wang, editors, *Proc. 41st International Symposium on Computational Geometry (SoCG'25)*, volume 332 of *LIPIcs*, pages 63:1–63:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.SOCG.2025.63.

28 Christian Komusiewicz. Tight running time lower bounds for vertex deletion problems. *ACM Transactions on Computation Theory (ToCT)*, 10(2):6:1–6:18, 2018. doi:10.1145/3186589.

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

30 Yanpei Liu, Aurora Morgana, and Bruno Simeone. A linear algorithm for 2-bend embeddings of planar graphs in the two-dimensional grid. *Discrete Applied Mathematics*, 81(1–3):69–91, 1998. doi:10.1016/s0166-218x(97)00076-0.

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

32 James Renegar. A polynomial-time algorithm, based on Newton's method, for linear programming. *Mathematical Programming*, 40–40(1–3):59–93, 1988. doi:10.1007/bf01580724.

33 Paolo Santi. Topology control in wireless ad hoc and sensor networks. *ACM Computing Surveys*, 37(2):164–194, 2005. doi:10.1145/1089733.1089736.

34 Jie Xue and Meirav Zehavi. Parameterized algorithms on geometric intersection graphs. *Computer Science Review*, 58:100796, 2025. doi:10.1016/J.COSREV.2025.100796.

35 Mihalis Yannakakis. Edge-deletion problems. *SIAM Journal on Computing*, 10(2):297–309, 1981. doi:10.1137/0210021.