Optimal parameterized algorithms for planar facility location problems using Voronoi diagrams*

Dániel Marx† Michał Pilipczuk‡

Abstract

We study a general family of facility location problems defined on planar graphs and on the 2-dimensional plane. In these problems, a subset of k objects has to be selected, satisfying certain packing (disjointness) and covering constraints. Our main result is showing that, for each of these problems, the $n^{\mathcal{O}(k)}$ time brute force algorithm of selecting k objects can be improved to $n^{\mathcal{O}(\sqrt{k})}$ time. The algorithm is based on an idea that was introduced recently in the design of geometric QPTASs, but was not yet used for exact algorithms and for planar graphs. We focus on the Voronoi diagram of a hypothetical solution of k objects, guess a balanced separator cycle of this Voronoi diagram to obtain a set that separates the solution in a balanced way, and then recurse on the resulting subproblems.

The following list is an exemplary selection of concrete consequences of our main result. We can solve each of the following problems in time $n^{\mathcal{O}(\sqrt{k})}$, where n is the total size of the input:

- d-Scattered Set: find k vertices in an edge-weighted planar graph that pairwise are at distance at least d from each other (d is part of the input).
- d-Dominating Set (or (k,d)-Center): find k vertices in an edge-weighted planar graph such that every vertex of the graph is at distance at most d from at least one selected vertex (d is part of the input).
- Given a set $\mathcal{D}$ of connected vertex sets in a planar graph G, find k disjoint vertex sets in $\mathcal{D}$.
- Given a set $\mathcal{D}$ of disks in the plane (of possibly different radii), find k disjoint disks in $\mathcal{D}$.
- Given a set $\mathcal{D}$ of simple polygons in the plane, find k disjoint polygons in $\mathcal{D}$.
- Given a set $\mathcal{D}$ of disks in the plane (of possibly different radii) and a set $\mathcal{P}$ of points, find k disks in $\mathcal{D}$ that together cover the maximum number of points in $\mathcal{P}$.
- Given a set $\mathcal{D}$ of axis-parallel squares in the plane (of possibly different sizes) and a set $\mathcal{P}$ of points, find k squares in $\mathcal{D}$ that together cover the maximum number of points in $\mathcal{P}$.

It is known from previous work that, assuming the Exponential Time Hypothesis (ETH), there is no $f(k)n^{o(\sqrt{k})}$ time algorithm for any computable function f for any of these problems. Furthermore, we give evidence that packing problems have $n^{\mathcal{O}(\sqrt{k})}$ time algorithms for a much more general class of objects than covering problems have. For example, we show that, assuming ETH, the problem where a set $\mathcal{D}$ of axis-parallel rectangles and a set $\mathcal{P}$ of points are given, and the task is to select k rectangles that together cover the entire point set, does not admit an $f(k)n^{o(k)}$ time algorithm for any computable function f.

*The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreements n. 280152 (D. Marx) and n. 267959 (M. Pilipczuk, while he was affiliated with the University of Bergen, Norway). The further work of M. Pilipczuk on this research at the University of Warsaw is supported by Polish National Science Centre grant DEC-2013/11/D/ST6/03073; at this time, M. Pilipczuk held also a post-doc position at Warsaw Center of Mathematics and Computer Science. The work of D. Marx is also supported by Hungarian Scientific Research Fund (OTKA) grant NK105645.

†Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI) `dmarx@cs.bme.hu`.

‡Institute of Informatics, University of Warsaw, Poland, `michal.pilipczuk@mimuw.edu.pl`

1 Introduction

Parameterized problems often become easier when restricted to planar graphs: usually significantly better running times can be achieved and sometimes even problems that are W[1]-hard on general graphs can become fixed-parameter tractable on planar graphs. In most cases, the improved running time involves a square root dependence on the parameter: for example, it is often of the form $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ or $n^{\mathcal{O}(\sqrt{k})}$. The appearance of the square root can be usually traced back to the fact that a planar graph with n vertices has treewidth $\mathcal{O}(\sqrt{n})$. Indeed, the theory of bidimensionality gives a quick explanation why problems such as Independent Set, Longest Path, Feedback Vertex Set, Dominating Set, or even distance-r versions of Independent Set and Dominating Set (for fixed r) have algorithms with running time $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ [11, 12, 13, 14, 15, 16, 18, 19, 24, 47]. In all these problems, there is a relation between the size of the largest grid minor and the size of the optimum solution, which allows us to bound the treewidth of the graph in terms of the parameter of the problem. More recently, subexponential parameterized algorithms have been explored also for problems where there is no such straightforward parameter-treewidth bound: for example, for Partial Vertex Cover and Dominating Set [22], k-Internal Out-Branching and k-Leaf Out-Branching [17], Multiway Cut [33], Subset TSP [34], Strongly Connected Steiner Subgraph [9], Steiner Tree [42, 43]. For some of these problems, it is easy to see that they are fixed-parameter tractable on planar graphs, and the challenge is to make the dependence on k subexponential, e.g., to obtain $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ time (or perhaps $2^{\mathcal{O}(\sqrt{k}\log k)} \cdot n^{\mathcal{O}(1)}$ time) algorithms. Others are W[1]-hard on planar graphs, and then the challenge is to improve the known $n^{\mathcal{O}(k)}$ time algorithm to $n^{\mathcal{O}(\sqrt{k})}$ time. For all these problems, there are matching lower bounds showing that, assuming the Exponential Time Hypothesis (ETH) of Impagliazzo, Paturi, and Zane [30, 35], there are no $2^{o(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ time algorithms (for FPT problems) or $n^{o(\sqrt{k})}$ time algorithms (for W[1]-hard problems).

A similar "square root phenomenon" has been observed in the case of geometric problems: it is usual to see a square root in the exponent of the running time of algorithms for NP-hard problems defined in the 2-dimensional Euclidean plane. For example, TSP and Steiner Tree on n points can be solved in time $2^{\mathcal{O}(\sqrt{n}\log n)}$ [46]. Most relevant to our paper is the fact that Independent Set for unit disks (given a set of n unit disks, select k of them that are pairwise disjoint) and the discrete k-center problem (given a set of n points and a set of n unit disks, select k disks whose union covers every point) can be solved in time $n^{\mathcal{O}(\sqrt{k})}$ by geometric separation theorems and shifting arguments [3, 4, 5, 29, 40], improving on the trivial $n^{\mathcal{O}(k)}$ time brute force algorithm. However, all of these algorithms are crucially based on a notion of area and rely on the property that all the disks have the same size (at least approximately). Therefore, it seems unlikely that these techniques can be generalized to the case when the disks can have very different radii or to planar-graph versions of the problem, where the notion of area is meaningless. Using similar techniques, one can obtain approximation schemes for these and related geometric problems, again with the limitation that the objects need to have (roughly) the same area.

Very recently, a new and powerful technique emerged from a line of quasi-polynomial time approximation schemes (QPTAS) for geometric problems [1, 2, 28, 41]. As described explicitly by Har-Peled [28], the main idea is to reason about the Voronoi diagram of the k objects in the solution. In particular, we are trying to guess a separator consisting of $\mathcal{O}(\sqrt{k})$ segments that corresponds to a balanced separator of the Voronoi diagram. In this paper, we show how this basic idea and its extensions can be implemented to obtain $n^{\mathcal{O}(\sqrt{k})}$ time exact algorithms for a wide family of geometric packing and covering problems in a uniform way. In fact, we show that the algorithms can be made to work in the much more general context of planar graph problems.

Algorithmic results. We study a general family of facility location problems for planar graphs, where a set of k objects has to be selected, subject to certain independence and covering constraints. Two archetypal problems from this family are (1) selecting k vertices of an edge-weighted planar graph that are at distance at least d from each other (d-Scattered Set) and (2) selecting k vertices of an edge-weighted planar graph such that every vertex of the graph is at distance at most d from some selected vertex (d-Dominating Set); for both problems, d is a real value being part of the input. We show that, under very general conditions, the trivial $n^{\mathcal{O}(k)}$ time brute force algorithm can be improved to $n^{\mathcal{O}(\sqrt{k})}$ time for problems in this family. Our result is not just a simple consequence of bidimensionality and bounding the treewidth of the input graph. Instead, we focus on the Voronoi diagram of a hypothetical solution, which can be considered as a planar graph with $\mathcal{O}(k)$ vertices. It is known that such a planar graph has a balanced separator cycle of length $\mathcal{O}(\sqrt{k})$, which can be translated into a separator that breaks the instance in way suitable for using recursion on the resulting subproblems: each subproblem contains at most a constant fraction of the k objects of the solution. Of course, we do not know the Voronoi diagram of the solution and its balanced separator cycle. However, we argue that only $n^{\mathcal{O}(\sqrt{k})}$ separator cycles can be potential candidates. Thus, by guessing one of these cycles, we define and solve $n^{\mathcal{O}(\sqrt{k})}$ subproblems. The reason why this scheme yields an $n^{\mathcal{O}(\sqrt{k})}$ time algorithm is the fact that recurrence relations of the form $f(k) = n^{\mathcal{O}(\sqrt{k})} f(k/2)$ resolves to $f(k) = n^{\mathcal{O}(\sqrt{k})}$.

In Section 3.1, we define a general facility location problem called Disjoint Network Coverage, which contains numerous concrete problems of interest as special cases. We defer to Section 3.1 the formal definition of the problem and the exact statement of the running time we can achieve. In this introduction, we discuss specific algorithmic results following from the general result.

Informally, the input of Disjoint Network Coverage consists of an edge-weighted planar graph G, a set $\mathcal{D}$ of objects (which are connected sets of vertices in G) and a set $\mathcal{C}$ of clients (which are vertices of G). The task is to select a set of exactly k pairwise-disjoint[1] objects that maximizes the total number (or total prize) of the covered clients. We need to define what we mean by saying that an object covers a client: the input contains a radius for each object in $\mathcal{D}$ and a sensitivity for each client in $\mathcal{C}$, and the client is considered to be covered by an object if the sum of the radius and the sensitivity is at least the distance between the object and the client. In the special case when both the radius and the sensitivity are 0, this is equivalent to saying that the client is inside the object; when the radius is r and the sensitivity is 0, then this is equivalent to saying that the client is at distance at most r from the object. The objects and the clients may be equipped with weights and we may want to maximize/minimize the weight of the selected objects or the weight of covered clients.

The first special case of the problem is when there are no clients at all: then the task is to select k objects that are pairwise disjoint. Our algorithm solves this problem in complete generality: the only condition is that each object is a connected vertex set (i.e. it induces a connected subgraph of G).

Theorem 1.1 (packing connected sets). *Let G be a planar graph, $\mathcal{D}$ be a family of connected vertex sets of G, and k be an integer. In time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$, we can find a set of k pairwise disjoint objects in $\mathcal{D}$, if such a set exists.*

We can also solve the weighted version, where we want to select k members of $\mathcal{D}$ maximizing the total weight. As a special case, if we define the open ball $B_{v,d}$ to be the set of vertices at distance less d from v and let $\mathcal{D} = \{B_{v,d/2} \mid v \in V(G)\}$, then Theorem 1.1 gives us an $n^{\mathcal{O}(\sqrt{k})}$ time algorithm

[1] More preceisely, if the objects have different radii, then instead of requiring them to be pairwise disjoint, we require a technical condition called "normality," which we define in Section 3.1.

for d-Scattered Set, which asks for k vertices that are at distance at least d from each other (with d being part of the input). For unweighted graphs and fixed values of d, this problem is actually fixed-parameter tractable and can be solved in time $2^{\mathcal{O}(d \log d \cdot \sqrt{k})} \cdot n^{\mathcal{O}(1)}$ using a simple application of bidimensionality [47].

If each object in $\mathcal{D}$ is a single vertex and $r(\cdot)$ assigns a radius to each object (potentially different radii for different objects), then we get a natural covering problem. Thus, the following theorem is also a corollary of our general result.

Theorem 1.2 (covering vertices with centers of different radii). *Let G be a planar graph, let $D, C \subseteq V(G)$ be two subset of vertices, let $r\colon D \to \mathbb{Z}^+$ be a function, and k be an integer. In time $|D|^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$, we can find a set $S \subseteq D$ of k vertices that maximizes the number of vertices covered in C, where a vertex $u \in C$ is covered by $v \in S$ if the distance of u and v is at most $r(v)$.*

If $D = C = V(G)$, $r(v) = d$ for every $v \in V(G)$, and we are looking for a solution fully covering C, then we obtain as a special case d-Dominating Set (also called (k, d)-Center), that is, the problem of finding a set S of vertices such that every other vertex is at distance at most d from S. Theorem 1.2 gives an $n^{\mathcal{O}(\sqrt{k})}$ time algorithm for this problem (with d being part of the input). Again, for fixed d, bidimensionality theory gives a $2^{\mathcal{O}(d \log d \cdot \sqrt{k})} \cdot n^{\mathcal{O}(1)}$ time algorithm [12].

Theorem 1.2 can be interpreted as covering the vertices in C by very specific objects: we want to maximize the number of vertices of C in the union of k objects, where each object is a ball of radius $r(v)$ around a center v. If we require that the selected objects of the solution are pairwise disjoint, then we can generalize this problem to arbitrary objects.

Theorem 1.3 (covering vertices with independent objects). *Let G be a planar graph, let $\mathcal{D}$ be a set of connected vertex sets in G, let $C \subseteq V(G)$ be a set of vertices, and let k be an integer. In time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$, we can find a set S of at most k pairwise disjoint objects in $\mathcal{D}$ that maximizes the number of vertices of C in the union of the vertex sets in S.*

Our algorithmic results are also applicable to geometric problems. By simple reductions (see Section 3.2), packing and covering geometric problems can be reduced to problems on planar graphs (although one has to handle rounding and precision issues carefully). In particular, given a set of disks (of possibly different radii), the problem of selecting k pairwise disjoint disks can be reduced to the problem of selecting disjoint connected vertex sets in a planar graph, which can be solved using Theorem 1.1. Alternatively, the algorithmic techniques behind our main results (Voronoi diagrams, balanced separators, recursion) can be expressed directly in the geometric setting, yielding a somewhat simpler algorithm that does not have to handle some of the degeneracies appearing in the more general setting of planar graphs. In Section 2, we discuss some of these geometric algorithms in a self-contained way, which may help the reader to understand the main ideas of the more technical planar graph algorithm.

Theorem 1.4 (packing disks). *Given a set $\mathcal{D}$ of disks (of possibly different radii) in the plane, in time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})}$ we can find a set of k pairwise disjoint disks, if such a set exists.*

This is a strong generalization of the results of Alber and Fiala [5], which gives an $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})}$ time algorithm only if the ratio of the radii of the smallest and largest disks can bounded by a constant (in particular, if all the disks are unit disks).

As Theorem 1.1 works for arbitrary connected sets of vertices, we can prove the analog of Theorem 1.4 for most reasonable sets of connected geometric objects. We do not want dwell on exactly what kind of geometric objects we can handle (e.g., whether the objects can have holes, how the boundaries are described etc.), hence we state the result only for simple (that is, non-self-crossing) polygons.

Theorem 1.5 (packing simple polygons). *Given a set $\mathcal{D}$ of simple polygons in the plane, in time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ we can find a set of k polygons in $\mathcal{D}$ with pairwise disjoint closed interiors, if such a set exists. Here n is the total number of vertices of the polygons in $\mathcal{D}$.*

Geometric covering problems can be also reduced to planar problems. The problem of covering the maximum number of points by selecting k disks from a given set $\mathcal{D}$ of disks can be reduced to a problem on planar graphs and then Theorem 1.2 can be invoked. This reduction relies on the fact that covering by a disk of radius r can be expressed as being at distance at most r from the center of the disk, which is precisely what Theorem 1.2 is about.

Theorem 1.6 (covering points with disks). *Given a set $\mathcal{C}$ of points and a set $\mathcal{D}$ of disks (of possibly different radii) in the plane, in time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot |\mathcal{C}|^{\mathcal{O}(1)}$ we can find a set of k disks in $\mathcal{D}$ maximizing the total number of points they cover in $\mathcal{C}$.*

The problem of covering points with axis-parallel squares (of different sizes) can be handled similarly. Observe that an axis-parallel square with side length s covers a point p if and only if p is at distance at most $s/2$ from the center of the square *in the ℓ_∞ metric.* This allows us to reduce the geometric problem to the planar problem solved by Theorem 1.2.

Theorem 1.7 (covering points with squares). *Given a set $\mathcal{C}$ of points and a set $\mathcal{D}$ of axis-parallel squares (of possibly different size) in the plane, in time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot |\mathcal{C}|^{\mathcal{O}(1)}$ we can find a set of k squares in $\mathcal{D}$ maximizing the total number of points they cover in $\mathcal{C}$.*

Hardness results. There are several lower bounds suggesting that our main algorithmic result is, in many aspects, best possible—both in terms of the form of the running time and the generality of the problem being solved. The Disjoint Network Coverage problem we define in Section 3.1 gives a very general family of problems, including many artificial problems. Therefore, it is not very enlightening to show that the running time we obtain for Disjoint Network Coverage cannot be improved. What we really want to know is whether our algorithm gives the best possible running time in concrete special cases of interest, such as those in Theorems 1.1–1.7.

There have been investigations of the parameterized complexity of various geometric packing and covering problems, giving tight ETH-based lower bounds in many cases [36, 37, 38]. Many of these reductions can be described conveniently using the Grid Tiling problem as the source of reductions. Reductions using Grid Tiling involve a quadratic blow up in the parameter and therefore they give lower bounds with a square root in the exponent. For example, assuming ETH, the problem of finding k disjoint objects from a set of unit disks, or a set of axis-parallel unit squares, or a set of unit segments (of arbitrary directions) cannot be solved in time $f(k)n^{o(\sqrt{k})}$ for any computable function f [37, 38]. This shows the optimality of Theorems 1.4 and 1.5. In Dominating Set problem for unit disks, the task is to select k of the disks such that every disk is either selected or intersected by a selected disk; assuming ETH, there is no $f(k)n^{o(\sqrt{k})}$ time algorithm for this problem for any computable function f [37]. Observe that if $\mathcal{D}$ is a set of unit disks (that is, disks of radius 1) and C set of centers of these disks, then a subset of $S \subseteq \mathcal{D}$ is a dominating set if and only if replacing every disk in S with a disk of radius 2 covers every point in C. Therefore, covering points with disks (of the same radius) is more general than Dominating Set for unit disks, hence the optimality of Theorem 1.6 follows from the lower bounds for Dominating Set for unit disks. In a similar way, the optimality of Theorem 1.7 follows from the lower bounds on Dominating Set for axis-parallel unit squares [37].

As we shall see in Section 3.2, there are reductions from the geometric problems to the planar problems. Besides the algorithmic consequences, these reductions allow us to transfer lower bounds for geometric problems to the corresponding planar problems. In particular, it follows that, assuming ETH, the running time in Theorem 1.1 cannot be improved to $f(k)n^{o(\sqrt{k})}$ (even if $|\mathcal{D}| = n^{\mathcal{O}(1)}$). With a direct reduction from Grid Tiling, one can also show that there is no $f(k)n^{o(\sqrt{k})}$ time algorithm for d-Scattered Set and d-Dominating Set on planar graphs, with d being part of the input (these reductions will appear elsewhere).

Comparing packing results Theorems 1.1 and 1.5 with covering results Theorems 1.2, 1.6, and 1.7, one can observe that our algorithm solves packing problems in much wider generality than covering problems. It seems that we can handle arbitrary objects in packing problems, while it is essential for covering problems that each object is a "ball," that is, it is defined as a set of points that are at most at a certain distance from a center. (Theorem 1.3 seems to be an exception, as it is a covering problem with arbitrary objects, but notice that we require independent objects in the solution, so it is actually a packing problem as well.) We present a set of hardness results suggesting that this apparent difference between packing and covering problems is not a shortcoming of our algorithm, but it is inherent to the problem: there are very natural geometric covering problems where the square root phenomenon does not occur.

Our strongest lower bound is not based not on ETH, but on the variant called Strong Exponential Time Hypothesis (SETH), which can be informally stated as n-variable CNF-SAT not having algorithms with running time $(2-\epsilon)^n$ for any $\epsilon > 0$ (cf. [35]). Using a result of Pătraşcu and Williams [44] and a simple reduction from Dominating Set, we show that if the task is to cover all the vertices of a planar graph G by selecting k sets from a collection $\mathcal{D}$ of connected vertex sets, then is unlikely that one can do significantly better than trying all $|\mathcal{D}|^k$ possible sets of objects.

Theorem 1.8 (covering vertices with connected sets, lower bound). *Let G be a planar graph and let $\mathcal{D}$ be a set of connected vertex sets of G. Assuming SETH, there is no $f(k)\cdot(|\mathcal{D}|+|V(G)|)^{k-\epsilon}$ time algorithm for any computable function f and any $\epsilon > 0$ that decides if there are k sets in $\mathcal{D}$ whose union covers $|V(G)|$.*

A similar reduction gives a lower bound for covering points with convex polygons.

Theorem 1.9 (covering points with convex polygons, lower bound). *Let $\mathcal{D}$ be a set of convex polygons and let $\mathcal{P}$ be a set of points in the plane. Assuming SETH, there is no $f(k)\cdot(|\mathcal{D}|+|\mathcal{P}|)^{k-\epsilon}$ time algorithm for any computable function f and $\epsilon > 0$ that decides if there are k polygons in $\mathcal{D}$ that together cover $\mathcal{P}$.*

The convex polygons appearing in the hardness proof of Theorem 1.9 are relatively "fat" (i.e., the area of each polygon is at most a constant factor smaller than the smallest enclosing disk), and they have an unbounded number of vertices. Therefore, it may still be possible that the square root phenomenon occurs for simpler polygons and we can have $n^{\mathcal{O}(\sqrt{k})}$ time algorithms. We show that this is not the case: we give two lower bounds for axis-parallel rectangles. The first bound is for "thin" rectangles (of only two types), while the second bound is for rectangles that are "almost squares."

Theorem 1.10 (covering points with thin rectangles, lower bound). *Consider the problem of covering a set $\mathcal{P}$ of points by selecting k axis-parallel rectangles from a set $\mathcal{D}$. Assuming ETH, there is no algorithm for this problem with running time $f(k)\cdot(|\mathcal{P}|+|\mathcal{D}|)^{o(k)}$ for any computable function f, even if each rectangle in $\mathcal{D}$ is of size $1\times k$ or $k\times 1$.*

Theorem 1.11 (covering points with almost squares, lower bound). *Consider the problem of covering a set $\mathcal{P}$ of points by selecting k axis-parallel rectangles from a set $\mathcal{D}$. Assuming ETH,* *for every $\epsilon_0 > 0$, there is no algorithm for this problem with running time $f(k) \cdot (|\mathcal{P}| + |\mathcal{D}|)^{o(k/\log k)}$ for any computable function f, even if each rectangle in $\mathcal{D}$ has both width and height in the range $[1-\epsilon_0, 1+\epsilon_0]$.*

Theorem 1.11 shows that even a minor deviation from the setting of Theorem 1.7 makes it unlikely that $n^{\mathcal{O}(\sqrt{k})}$ algorithms exist. Therefore, it seems that for covering problems the existence of the square root phenomenon depends not on the objects being simple, or fat, or almost the same size, but really on the fact that the objects are defined as balls in a metric.

Our techniques. The standard technique of bidimensionality does not seem to be applicable to our problems: it is not clear for any of the problems how the existence of an $\mathcal{O}(\sqrt{k})\times\mathcal{O}(\sqrt{k})$ grid minor helps in solving the problem, hence we cannot assume that the input graph has treewidth $\mathcal{O}(\sqrt{k})$. In more recent subexponential parameterized algorithms, we can observe a different algorithmic pattern: instead of trying to bound the treewidth of the *input graph,* we define a "skeleton graph" describing the structure of the *solution* and use planarity of the skeleton to bound its treewidth. Then the fact that this skeleton graph has treewidth $\mathcal{O}(\sqrt{k})$ (and, in particular, has balanced separators of size $\mathcal{O}(\sqrt{k})$) can be used to solve the problem. The right choice of the skeleton graph can be highly nonobvious: for example, for Multiway Cut [33], the skeleton graph is the union of the dual of the solution with a minimum Steiner tree, while for Subset TSP [34], the skeleton is the union of the solution with a locally optimal solution. In our problem, we again define a skeleton graph based on a solution and exploit that its treewidth is $\mathcal{O}(\sqrt{k})$. This time, the right choice for the skeleton seems to be the Voronoi diagram of the k objects forming the solution. Then we exploit the fact that this Voronoi diagram has separator cycles of length $\mathcal{O}(\sqrt{k})$ to find a suitable way of separating the instance into subproblems.

Given a set $\mathcal{P}$ of points in the plane, the Voronoi region of a point $p \in \mathcal{P}$ consists of those points of the plane that are closer to p than any other member of $\mathcal{P}$. The boundaries of the Voronoi regions are segments that are equidistant to two points from $\mathcal{P}$, forming a diagram that can be considered a planar graph. We can define Voronoi regions of a graph and a set $\mathcal{D}$ of disjoint connected vertex sets in a similar way, by classifying vertices according to the closest object in $\mathcal{D}$. If the graph is planar, then we can use the edges on the boundary of the regions (in the dual graph) to construct a planar graph that is an analog of the Voronoi diagram. If $|\mathcal{D}| = k$, then this diagram has $\mathcal{O}(k)$ vertices and, by a well-known property of planar graphs, has treewidth $\mathcal{O}(\sqrt{k})$.

The separator cycle that we need is actually a noose: a closed curve that intersects the graph only in its vertices and visits each face at most once. It can be deduced from known results on sphere cut decompositions that a planar graph with k vertices has a noose that visits $\mathcal{O}(\sqrt{k})$ vertices and there are at most $\frac{2}{3}k$ faces strictly inside/outside the noose. The noose can be described by a cyclic sequence of $\mathcal{O}(\sqrt{k})$ vertices and faces. Such a sequence of vertices and faces of the Voronoi diagram can be translated into a sequence of shortest paths connecting points from $\mathcal{D}$ and the branch vertices of the Voronoi diagram, forming a closed cycle in the original graph, separating the inside and the outside. These separator cycles are the most important conceptual objects for our algorithm. The crucial observation is that, because of the properties of the Voronoi diagram, objects of the optimum solution inside the cycle cannot interact with outside world: they cannot intersect the cycle and, in covering problems, if a point is sufficiently close to an object inside the cycle, then it is sufficiently close also to an object on the boundary.

Of course, we do not know the Voronoi diagram of the k objects in the solution. But since the separator cycles are defined by a selection of $\mathcal{O}(\sqrt{k})$ objects/branch vertices, we can enumerate $n^{\mathcal{O}(\sqrt{k})}$ candidates for them. We branch into $n^{\mathcal{O}(\sqrt{k})}$ subproblems indexed by these candidates, and in each subproblem we assume that the selected candidate corresponds to a balanced noose in the Voronoi diagram of the solution. This assumption has certain consequences and we modify the instance accordingly. For example, we can deduce that certain members of $\mathcal{D}$ cannot be in the solution, and hence they can be removed from $\mathcal{D}$. After these modifications, we can observe that the problem falls apart into into subproblems: this is because there cannot be any interaction between the objects inside and outside the separator. Therefore, we can recursively solve these subproblems, where the parameter value is at most $\frac{2}{3}k$. Because of guessing the separator cycle, we eventually solve $n^{\mathcal{O}(\sqrt{k})}$ subproblems, each with parameter value at most $\frac{2}{3}k$, which results in the claimed $n^{\mathcal{O}(\sqrt{k})}$ running time following from the recursive formula.

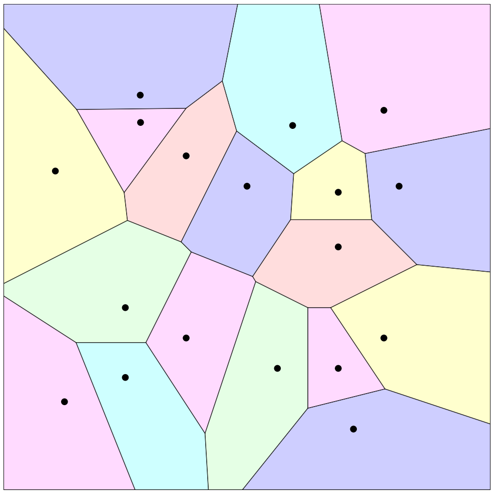
Lower bounds on how k has to appear in the exponent, such as the lower bounds in Theorems 1.9–1.11, can be obtained by parameterized reductions from a W[1]-hard problem. The strength of the lower bound depends on how the parameter k changes in the reduction. The reductions based on Grid Tiling involve a quadratic blowup and hence are able to rule out only algorithms of the form $f(k)n^{o(\sqrt{k})}$ (assuming ETH). The lower bounds in Theorems 1.9–1.11 are stronger: they show that even $n^{\mathcal{O}(\sqrt{k})}$ time algorithms are far from being possible. Therefore, they are very different from typical hardness proofs for planar and geometric problems based on Grid Tiling. Of particular interest is Theorem 1.11, where a tight reduction from Clique seems problematic, as it seems difficult to implement the $\mathcal{O}(k^2)$ pairwise interaction of the Clique problem with squares of almost the same size. Instead, we rely on a nontrivial hardness result for Partitioned Subgraph Isomorphism [39], which gives a strong lower bound even when the graph H to be found is sparse.

2 Geometric problems

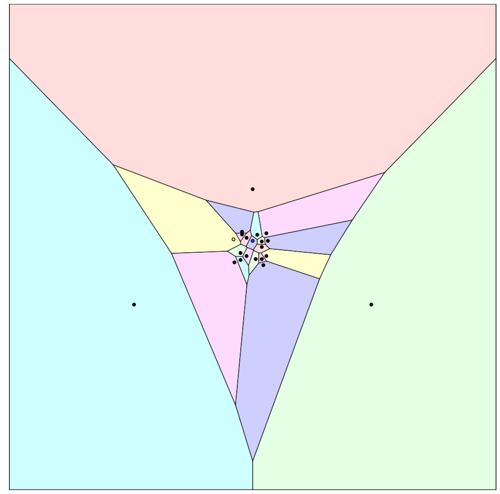
Our main algorithmic result is a technique for solving a general facility location problem on planar graphs in time $n^{\mathcal{O}(\sqrt{k})}$. With simple reductions, we can use these algorithms to solve 2-dimensional geometric problems (see Section 3.2). However, our main algorithmic ideas can be implemented also directly in the geometric setting, giving self-contained algorithms for geometric problems. These geometric algorithms avoid some of the technical complications that arise in the planar graph counterparts, such as the Voronoi diagram having bridges or shortest paths sharing subpaths. Unfortunately, a large part of the paper is devoted to the formal handling of these issues. Therefore, it could be instructive for the reader to see first a self-contained presentations of some of the geometric results.

Packing unit disks. We start with the Independent Set problem for unit disks: given a set $\mathcal{D}$ of closed disks of unit radius in the plane, the task is to select a set of k pairwise disjoint disks. This problem is known to be solvable in time $n^{\mathcal{O}(\sqrt{k})}$ [5, 40]. We present another $n^{\mathcal{O}(\sqrt{k})}$ algorithm for the problem, demonstrating how we can solve the problem recursively by focusing on the Voronoi diagram of a hypothetical solution. This idea appeared recently in the context of constructing quasi-polynomial time approximation schemes (QPTAS) for geometric problems (see, e.g., [28]), but it has not been used explicitely for exact algorithms.

While the algorithm we present in this section on its own does not deliver any new result yet, it is significantly different from the previous algorithms, which crucially use the notion of "area," in particular, by using the fact that a region of area $\mathcal{O}(1)$ can contain only $\mathcal{O}(1)$ independent unit disks. Our algorithm uses only the notion of distance, making it possible to translate it to the language of planar graphs, where the notion of area does not make sense. Furthermore, as we shall see, generalizations to disks of different radii and to covering problems are relatively easy for our algorithm. In what follows, it is oftem more convenient to think in terms of an equivalent formulation of the problem where instead of packing disks from $\mathcal{D}$ we are packing their centers subject to a constraint that every pair of packed centers has to be at distance more than 2 from each other. We will switch between these two formulations implicitly.



(a) Voronoi diagram of 17 points



(b) Voronoi diagram of 17 points with 3 guard points added. Observe that the only infinite regions are those of the guard points.

Figure 1: Voronoi diagrams

Let $\mathcal{P}$ be a set of points in the plane. The *Voronoi region* of $p \in \mathcal{P}$ is the set of those points x in the plane that are "closest" to p in the sense that the distance of x and $\mathcal{P}$ is exactly the distance of x and p (see Figure 1(a)). The Voronoi region of p can be obtained as the intersection of half-planes: for every $p' \in \mathcal{P}$ different from p, the Voronoi region is contained in the half-plane of points whose distance from p is not greater than the distance from p'. This implies that every Voronoi region is convex.

Even though defining Voronoi diagrams and working with them is much simpler in the plane than for their analogs in planar graphs (see Section 4.3), there is a technical difficulty specific to the plane. The issue is that the Voronoi region of a point $p \in \mathcal{P}$ can be infinite and consequently the Voronoi diagram consists of finite segments and infinite rays. Therefore, it is not clear what we mean by the graph of the Voronoi diagram. While this complication does not give any conceptual difficulty in the algorithm, we need to address it formally.

First, we introduce three new "guard" points into $\mathcal{P}$, at distance more than r from the other points and from each other. We introduce these three guards in such a way that every original point is inside the triangle formed by them (see Figure 1(b)). It is easy to see now that the Voronoi region of every original point is finite. Therefore, the only infinite regions are the regions of the three guards and it follows that finite segments of the Voronoi diagram form a 2-connected planar graph and there are three infinite rays in the infinite face. In the case of the packing problem, if we introduce three new disks corresponding to the three guards, then these three disks can be always selected into every solution. Thus instead of trying to find k independent disks in the original set, we can equivalently try to find $k+3$ disks in the new set. As increasing k by 3 does not change the asymptotic running time $n^{\mathcal{O}(\sqrt{k})}$ we are aiming for, in the following we assume that the set $\mathcal{P}$ of center points has this form, that is, contains three guard points.

If a vertex of the Voronoi diagram has degree more than 3, then this means that there are four points appearing on a common circle. In order to simplify the presentation, we may introduce small perturbation to the coordinates to ensure that this does not happen for any four points. Moreover, we may identify the "endpoints" of the three infinite rays into a new point at infinity. Therefore, in the following we assume that the Voronoi diagram is actually a 2-connected 3-regular planar graph. Let us emphasize again that these technicalities appear only in the geometric setting and will not present a problem when we are proving the main result for planar graphs.

We are now ready to explain the main combinatorial idea behind the algorithm for finding k independent unit disks. Consider now a hypothetical solution consisting of k independent disks and let us consider the Voronoi diagram of the centers of these k points (see Figure 2(a)). To emphasize that we consider the Voronoi diagram of the centers of the k disks in the solution and *not* the centers of the n disks in the input, we call this diagram the *solution Voronoi diagram.* As we discussed above, we may assume that the solution Voronoi diagram is a 2-connected 3-regular planar graph. There are various separator theorems in the literature showing that a k-vertex planar graph has balanced separators of size $\mathcal{O}(\sqrt{k})$. Certain technicalities appear in these theorems: for example, they may require the graph to be triangular or 2-connected, the separator may be a cycle in the primal graph or in the dual graph, etc. Therefore, in Section 4.7 we give a short proof showing that the known results on sphere cut decompositions imply a separator theorem of the form we need. A *noose* of a plane graph G is a closed curve δ on the sphere such that δ alternately travels through faces of G and vertices of G and every vertex and face of G is visited at most once. We show that every 3-regular planar graph G with k faces has a noose δ of length $\mathcal{O}(\sqrt{k})$ (that is, going through $\mathcal{O}(\sqrt{k})$ faces and vertices) that is *face balanced* in the sense that there are at most $\frac{2}{3}k$ faces of G strictly inside δ and at most $\frac{2}{3}k$ faces of G strictly outside δ.

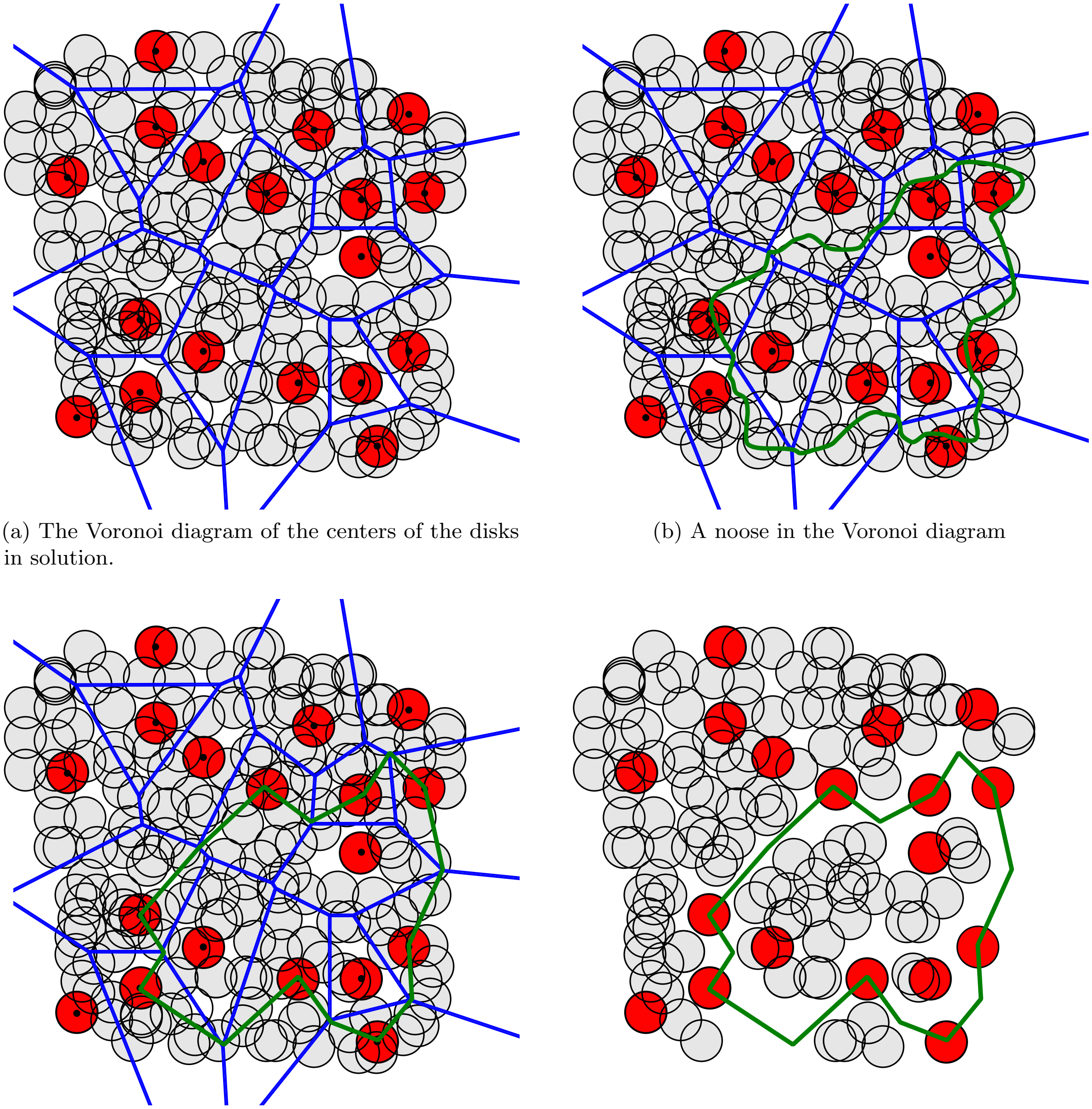
Consider a face-balanced noose δ of length $\mathcal{O}(\sqrt{k})$ as above (see the green curve in Figure 2(b)). Noose δ goes through $\mathcal{O}(\sqrt{k})$ faces of the solution Voronoi diagram, which corresponds to a set Q of $\mathcal{O}(\sqrt{k})$ disks of the solution. The noose can be turned into a polygon Γ with $\mathcal{O}(\sqrt{k})$ vertices the following way (see the green polygon in Figure 2(c)). Consider a subcurve of δ that is contained in the face corresponding to disk $p \in Q$ and its endpoints are vertices x and y of the solution Voronoi diagram. Then we can "straighten" this subcurve by replacing it with a straight line segment connecting x and the center of p, and a straight line segment connecting the center of p and y. Therefore, the vertices of the polygon Γ are center points of disks in Q and vertices of the solution Voronoi diagram. Observe that Γ intersects the Voronoi regions of the points in Q only. This follows from the convexity of the Voronoi regions: the segment between the center of $p \in Q$ and a point on the boundary of the region of p is fully contained in the region of p. In particular, this means that Γ does not intersect any disk other than those in Q.

The main idea is to use this polygon Γ to separate the problem into two subproblems. Of course, we do not know the solution Voronoi diagram and hence we have no way of computing from it the balanced noose δ and the polygon Γ. However, we can efficiently list $n^{\mathcal{O}(\sqrt{k})}$ candidate polygons. By definition, every vertex of the polygon Γ is either the center of a disk in $\mathcal{D}$ or a vertex of the solution Voronoi diagram. Every vertex of the solution Voronoi diagram is equidistant from the the centers of three disks in $\mathcal{D}$ and for any three such center points (in general position) there is a unique point in the plane equidistant from them. Thus every vertex of the polygon Γ is either a center of a disk in $\mathcal{D}$ or can be described by a triple of disks in $\mathcal{D}$. This means that Γ can be described by an $\mathcal{O}(\sqrt{k})$-tuple of disks from $\mathcal{D}$. That is, by branching into $n^{\mathcal{O}(\sqrt{k})}$ directions, we may assume that we have correctly guessed the subset Q of the solution and the polygon Γ.

What do we do with the set Q and the polygon Γ? First, if Q is indeed part of the solution, then we may remove these disks from $\mathcal{D}$ and decrease the target number of disks to be found by $|Q|$. Second, we perform the following cleaning steps:

(1) Remove any disk that intersects a disk in Q.
(2) Remove any disk that intersects Γ.

Indeed, if Q is part of the solution, then no other disk intersecting Q can be part of the solution.



(a) The Voronoi diagram of the centers of the disks in solution.

(b) A noose in the Voronoi diagram

(c) The polygon corresponding to the noose.

(d) Removing every disk intersecting the polygon breaks the problem into two independent parts.

Figure 2: Using a noose in the Voronoi diagram for divide and conquer.

Moreover, we have observed above that in the solution the polygon Γ is contained in the Voronoi regions of the points in Q and hence no disk other than the disks in Q intersects Γ, justifying the removal of such disks. We say that these removed disks are *banned* by (Q, Γ).

After these cleaning steps, the instance falls apart into two independent parts: each remaining disk is either strictly inside Γ or strictly outside Γ (see Figure 2(d)). Moreover, recall that the noose δ was face balanced and hence there are at most $\frac{2}{3}k$ faces of the solution Voronoi diagram inside/outside δ. This implies that the solution contains at most $\frac{2}{3}k$ center points inside/outside Γ. Therefore, in the two recursive calls, we need to look for at most that many independent disks. For

$k' := 1, \ldots, \lfloor \frac{2}{3}k \rfloor$, we recursively try to find exactly k' independent disks from the input restricted to the inside/outside Γ, resulting in $2 \cdot \frac{2}{3}k$ recursive calls.[2] Taking into account the $n^{\mathcal{O}(\sqrt{k})}$ guesses for Q and Γ, the number of subproblems we need to solve is $2 \cdot \frac{2}{3}k \cdot n^{\mathcal{O}(\sqrt{k})} = n^{\mathcal{O}(\sqrt{k})}$ (as $k \leq n$, otherwise there is no solution) and the parameter value is at most $\frac{2}{3}k$ in each subproblem. Therefore, if we denote by $T(n,k)$ the time needed to solve the problem with at most n points and parameter value at most k, we arrive to the recursion

$$T(n,k) = n^{\mathcal{O}(\sqrt{k})} \cdot T(n, (2/3)k).$$

Solving the recursion gives

$$\begin{aligned} T(n,k) &= n^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(\sqrt{\frac{2}{3}k})} \cdot n^{\mathcal{O}(\sqrt{(\frac{2}{3})^2 k})} \cdot n^{\mathcal{O}(\sqrt{(\frac{2}{3})^3 k})} \cdots \\ &= n^{\mathcal{O}((1+(\frac{2}{3})^{\frac{1}{2}}+(\frac{2}{3})^{\frac{2}{2}}+(\frac{2}{3})^{\frac{3}{2}}+\ldots)\sqrt{k})} = n^{\mathcal{O}(\sqrt{k})}, \end{aligned}$$

as the coefficient of $\sqrt{k}$ in the exponent is a constant (being the sum of a geometric series with ratio $\sqrt{2/3}$). Therefore, the total running time for finding k independent disks is $n^{\mathcal{O}(\sqrt{k})}$. This proves the first result: packing unit disks in the plane in time $n^{\mathcal{O}(\sqrt{k})}$. Let us repeat that this result was known before [5, 40], but as we shall see, our algorithm based on Voronoi diagrams can be generalized to objects of different size, planar graphs, and covering problems.

Covering points by unit disks. Let us now consider the following problem: given a set $\mathcal{D}$ of unit disks and a set $\mathcal{C}$ of client points, we need to select k disks from $\mathcal{D}$ that together cover every point in $\mathcal{C}$. We show that this problem can be solved in time $n^{\mathcal{O}(\sqrt{k})}$ using an approach based on finding separators in the Voronoi diagram.

Similarly to the way we handled the packing of unit disks, we can consider the Voronoi diagram of the center points in the solution. Note, however, that this time the disks in the solution are not necessarily disjoint, but this does not change the fact that their center points (which can be assumed to be distinct) define a Voronoi diagram. Therefore, it will be convenient to switch to an equivalent formulation of the problem described in terms of the centers of the disks: $\mathcal{D}$ is a set of points and we say that a selected point in $\mathcal{D}$ covers a point in $\mathcal{C}$ if their distance is at most 1.

Similarly to the case of packing, we can try $n^{\mathcal{O}(\sqrt{k})}$ possibilities to guess a set $Q \subseteq \mathcal{D}$ of center points and a polygon Γ that corresponds to a face-balanced noose. The question is how to use Γ to split the problem into two independent subproblems. The cleaning steps (1) and (2) above for the packing problem are no longer applicable: the solution may contain disks intersecting the disks with centers in Q and the solution may contain further disks intersecting the polygon Γ. What we do instead is the following. First, if we assume that Q is part of the solution, then any point in $\mathcal{C}$ covered by some point in Q can be removed, as it is already covered. Second, we know that in the solution Voronoi diagram every point of Γ belongs to the Voronoi region of some point in Q, hence we can remove any point from $\mathcal{D}$ that is inconsistent with this assumption. That is, if there is a $p \in \mathcal{D}$ and $v \in \Gamma$ such that p is closer to v than to every point in Q, then p can be safely removed from $\mathcal{D}$; we say in this case that (Q, Γ) *bans* p. For a $p \in \mathcal{D}$ and for each segment of Γ, it is not difficult to check if the segment contains such a point v (we omit the details). Thus we have now the following two cleaning steps:

(1) Remove every point from $\mathcal{C}$ that is covered by Q.

[2] Doing a recursive call for each k' may seem unnecessarily complicated at this point: what we really need is a single recursive call returning the maximum number of independent disks, or $\frac{2}{3}k$ independent disks, whichever is smaller. However, we prefer to present the algorithm in a way similar to how the more general problems will be solved later on.

(2) Remove every point from $\mathcal{D}$ that is closer to a point of Γ than every point in Q.

Let $\mathcal{D}_{\text{in}}$ and $\mathcal{D}_{\text{out}}$ be the remaining points in $\mathcal{D}$ strictly inside/outside Γ and let $\mathcal{C}_{\text{in}}$ and $\mathcal{C}_{\text{out}}$ be the remaining points in $\mathcal{C}$ strictly inside/outside Γ. We know that the solution contains at most $\frac{2}{3}k$ center points inside/outside Γ. Therefore, for $k' = 1, \ldots, \lfloor \frac{2}{3}k \rfloor$, we solve two subproblems, with point sets $(\mathcal{D}_{\text{in}}, \mathcal{C}_{\text{in}})$ and $(\mathcal{D}_{\text{out}}, \mathcal{C}_{\text{out}})$.

If there is a set of k_{in} of center points in $\mathcal{D}_{\text{in}}$ covering $\mathcal{C}_{\text{in}}$ and there is a set of k_{out} center points in $\mathcal{D}_{\text{out}}$ covering $\mathcal{C}_{\text{out}}$, then, together with Q, they form a solution of $|Q| + k_{\text{in}} + k_{\text{out}}$ center points. By solving the defined subproblems optimally, we know the minimum value of k_{in} and k_{out} required to cover $\mathcal{C}_{\text{in}}$ and $\mathcal{C}_{\text{out}}$, and hence we can determine the smallest solution that can be put together this way. But is it true that we can always put together an optimum solution this way? The problem is that, in principle, the solution may contain a center point $p \in \mathcal{D}_{\text{out}}$ that covers some point $q \in \mathcal{C}_{\text{in}}$ that is not covered by any center point in $\mathcal{D}_{\text{in}}$. In this case, in the optimum solution the number of center points selected from $\mathcal{D}_{\text{in}}$ can be strictly less than what is needed to cover $\mathcal{C}_{\text{in}}$ and hence the way we are putting together a solution cannot result in an optimum solution.

Fortunately, we can show that this problem never arises, for the following reason. Suppose that there is such a $p \in \mathcal{D}_{\text{out}}$ and $q \in \mathcal{C}_{\text{in}}$. As p is outside Γ and q is inside Γ, the segment connecting p and q has to intersect Γ at some point $v \in \Gamma$, which means $\text{dist}(p, q) = \text{dist}(p, v) + \text{dist}(v, q)$. By cleaning step (2), there has to be a $p' \in Q$ such that $\text{dist}(p', v) \leq \text{dist}(p, v)$, otherwise p would be banned and we would have removed it from $\mathcal{D}$. This means that $\text{dist}(p, q) = \text{dist}(p, v) + \text{dist}(v, q) \geq \text{dist}(p', v) + \text{dist}(v, q) \geq \text{dist}(p', q)$. Therefore, if p covers q, then so does $p' \in Q$. But in this case we would have removed q from $\mathcal{C}$ in the first cleaning step. Thus we can indeed obtain an optimum solution the way we proposed, by solving optimally the defined subproblems.

As in the case of packing, we have $2 \cdot \frac{2}{3}k \cdot n^{\mathcal{O}(\sqrt{k})} = n^{\mathcal{O}(\sqrt{k})}$ subproblems, with parameter value at most $\frac{2}{3}k$. Therefore, the same recursion applies to the running time, resulting in an $n^{\mathcal{O}(\sqrt{k})}$ time algorithm.

Packing in planar graphs. How can we translate the geometric ideas explained above to the context of planar graphs? Let G be an edge-weighted planar graph and let $\mathcal{F}$ be a set of disjoint "objects," where each object is a connected set of vertices in G. Then we can define the analog of the Voronoi regions in a straightforward way: for every $p \in \mathcal{F}$, let M_p contain every vertex v to which p is the closest object in $\mathcal{F}$, that is, $\text{dist}(v, \mathcal{F}) = \text{dist}(v, p)$. By a perturbation of the edge weights, we may assume that there are no ties: vertex v cannot be at exactly the same distance from two objects $p_1, p_2 \in \mathcal{F}$. It follows that the sets M_p form a partition of $V(G)$ (here we use that the objects in $\mathcal{F}$ are disjoint). It is easy to verify that region M_p has the following convexity property: if $v \in M_p$ and P is a shortest path between v and p, then every vertex of P is in M_p.

While Voronoi regions are easy to define in graphs, the proper definition of Voronoi diagrams is far from obvious and it is also nontrivial how a noose δ in the Voronoi diagram defines the analog of the polygon Γ. We leave the discussion of these issues to Section 4, here we only define in an abstract way what our goal is and state in Lemma 2.1 below (a simplified version of) the main technical tool that is at the core of the algorithm. Note that the statement of Lemma 2.1 involves only the notion of Voronoi regions, hence there are no technical issues in interpreting and using it. However, in the proof we have to define the analog of the Voronoi diagram for planar graphs and address issues such that this diagram is not 2-connected etc. We defer the required technical definitions to Section 4.

Let us consider first the packing problem: given an edge-weighted graph G, a set $\mathcal{D}$ of d objects (connected subsets of vertices), and an integer k, the task is to find a subset $\mathcal{F} \subseteq \mathcal{D}$ of k pairwise disjoint objects. Looking at the algorithm for packing unit disks described above, what we need is a suitable *guarded separator*, which is a pair (Q, Γ) consisting of a set $Q \subseteq \mathcal{D}$ of $\mathcal{O}(\sqrt{k})$ objects and a subset $\Gamma \subseteq V(G)$ of vertices. If there is a hypothetical solution $\mathcal{F} \subseteq \mathcal{D}$ consisting of k disjoint objects, then we would like to have a guarded separator (Q, Γ) satisfying the following three properties: (1) Q is subset of the solution, (2) Γ is fully contained in the Voronoi regions of the objects in Q, and and (3) Γ separates the objects in $\mathcal{F}$ in a balanced way. Our main technical result is that it is possible to enumerate a set of $d^{\mathcal{O}(\sqrt{k})}$ guarded separators such that for every solution $\mathcal{F}$, one of the enumerated guarded separators satisfies these three properties. We state here a simplified version that is suitable for packing problems.

Lemma 2.1. *Let G be an n-vertex edge-weighted planar graph, $\mathcal{D}$ a set of d connected subsets of $V(G)$, and k an integer. We can enumerate (in time polynomial in the size of the output) a set $\mathcal{N}$ of $d^{\mathcal{O}(\sqrt{k})}$ pairs (Q, Γ) with $Q \subseteq \mathcal{D}$, $|Q| = \mathcal{O}(\sqrt{k})$, $\Gamma \subseteq V(G)$ such that the following holds. If $\mathcal{F} \subseteq \mathcal{D}$ is a set of k pairwise disjoint objects, then there is a pair $(Q, \Gamma) \in \mathcal{N}$ such that*

1. *$Q \subseteq \mathcal{F}$,*
2. *if $(M_p)_{p \in \mathcal{F}}$ are the Voronoi regions of $\mathcal{F}$, then $\Gamma \subseteq \bigcup_{p \in Q} M_p$,*
3. *for every connected component C of $G - \Gamma$, there are at most $\frac{2}{3}k$ objects of $\mathcal{F}$ that are fully contained in C.*

The proof goes along the same lines as the argument for the geometric setting. After carefully defining the analog of the Voronoi diagram, we can use the planar separator result to obtain a noose δ. The same way as this noose was turned into a polygon in the geometric algorithm, we "straighten" the noose into a closed walk in the graph connecting using shortest paths $\mathcal{O}(\sqrt{k})$ objects and $\mathcal{O}(\sqrt{k})$ vertices of the Voronoi diagram. The vertices of this walk separate the objects that are inside/outside the noose, hence it has the required properties. Thus by trying all sets of $\mathcal{O}(\sqrt{k})$ objects and $\mathcal{O}(\sqrt{k})$ vertices of the Voronoi diagram, we can enumerate a suitable set $\mathcal{N}$. A technical difficulty in the proof is that the definition of the vertices of the Voronoi diagram is nontrivial. Moreover, to achieve the bound $d^{\mathcal{O}(\sqrt{k})}$ instead of $n^{\mathcal{O}(\sqrt{k})}$, we need a nontrivial way of finding a set of $d^{\mathcal{O}(1)}$ candidate vertices; unlike in the geometric setting, enumerating vertices equidistant from three objects is not sufficient.

Armed with the set $\mathcal{N}$ given by Lemma 2.1, the packing problem can be solved in a way analogous to how we handled unit disks. We guess a pair $(Q, \Gamma) \in \mathcal{N}$ that satisfies the three properties of Lemma 2.1. From the first property, we know that Q is part of the solution, hence Q can be removed from $\mathcal{D}$ and the target number k of objects to be found can be decreased by $|Q|$. From the second property, we know that in the solution $\mathcal{F}$ the set Γ is fully contained in the Voronoi regions of the objects in Q. Thus we can remove any object from $\mathcal{D}$ intersecting Γ. That is, we have to solve the problem on the graph $G - \Gamma$. We can solve the problem independently on the components of $G - \Gamma$ and the third property of Lemma 2.1 implies that each component contains at most $\frac{2}{3}k$ objects of the solution. Thus for each component C of $G - \Gamma$ containing at least one object of $\mathcal{D}$ and for $k' = 1, \ldots, \lfloor \frac{2}{3}k \rfloor$, we recursively solve the problem on C with parameter k', that is, we try to find k' disjoint objects in C. Assuming that (Q, Γ) indeed satisfied the properties of Lemma 2.1 for the solution $\mathcal{F}$, the solutions of these subproblems allow us to put together a solution for the original problem. As at most d components of $G - \Gamma$ can contain objects from $\mathcal{D}$, we recursively solve at most $d \cdot \frac{2}{3}k$ subproblems for a given (Q, Γ). Therefore, the total number of subproblems we need to solve is at most $d \cdot \frac{2}{3}k \cdot |\mathcal{N}| = d \cdot \frac{2}{3}k \cdot d^{\mathcal{O}(\sqrt{k})} = d^{\mathcal{O}(\sqrt{k})}$ (assuming $k \leq d$). If we denote by $T(n, d, k)$ the running time for $|V(G)| \leq n$ and $|\mathcal{D}| \leq d$, then we arrive to the recursion $T(n, d, k) = d^{\mathcal{O}(\sqrt{k})} \cdot T(n, d, (2/3)k) + n^{\mathcal{O}(1)}$, which gives $T(n, k) = d^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$.

Covering in planar graphs. Let us consider now the following graph-theoretic analog of covering points by unit disks: given an edge-weighted planar graph G, two sets of vertices $\mathcal{D}$ and $\mathcal{C}$, and integers k and r, the task is to find a set $\mathcal{F} \subseteq \mathcal{D}$ of k vertices that covers every vertex in $\mathcal{C}$. Here we say that $p \in \mathcal{D}$ covers $q \in \mathcal{C}$ if $\text{dist}(p,q) \leq r$, that is, we can imagine p to represent a ball of radius r in the graph with center at p. Note that, unlike in the case of packing, $\mathcal{D}$ is a set of vertices, not a set of connected sets.

Let $\mathcal{F}$ be a hypothetical solution. We can construct the set $\mathcal{N}$ given by Lemma 2.1 and guess a guarded separator (Q,Γ) satisfying the three properties. As we assume that Q is part of the solution (second property of Lemma 2.1), we decrease the target number k of vertices to select by $|Q|$ and we can remove from $\mathcal{C}$ every vertex that is covered by some vertex in Q; let $\mathcal{C}'$ be the remaining set of vertices. Moreover, by the third property of Lemma 2.1, we can assume that in the solution $\mathcal{F}$, the set Γ is fully contained in the Voronoi regions of the vertices in Q. This means that if there is a $p \in \mathcal{D} \setminus Q$ and $v \in \Gamma$ such that $\text{dist}(p,v) < \text{dist}(p,Q)$, then p can be removed from $\mathcal{D}$, since it is surely not part of the solution. As in the case of covering with disks, we say that (Q,Γ) *bans* p. Let $\mathcal{D}'$ be the remaining set of vertices. For every component C of $G-S$ and $k'=1,\dots,\lfloor\frac{2}{3}k\rfloor$, we recursively solve the problem restricted to C, that is, with the restrictions $\mathcal{D}'[C]$ and $\mathcal{C}'[C]$ of the vertex sets. It is very important to point out that now (unlike how we presented the packing problem above) we *do not* change the graph G in each call: we use the same graph G with the restricted sets $\mathcal{D}'[C]$ and $\mathcal{C}'[C]$. The reason is that restricting to the graph $G[C]$ could potentially change the distances between two vertices $x,y \in C$, as it is possible that the shortest $x-y$ path leaves C.

If k_C is the minimum number of vertices in $\mathcal{D}'[C]$ that can cover $\mathcal{C}'[C]$, then we know that there are $|Q| + \sum k_C$ vertices in $\mathcal{D}$ that cover every vertex in $\mathcal{C}$. We argue that if there is a solution, we can obtain a solution this way. Analogously to the case of covering vertices with disks, we have to show that in the solution $\mathcal{F}$, every vertex in $\mathcal{C}'[C]$ is covered by some vertex in $\mathcal{D}'[C]$: it is not possible that there are two distinct components C_1, C_2 of $G-\Gamma$ and some $q \in \mathcal{C}'[C_1]$ is covered only by $p \in \mathcal{D}'[C_2]$. Suppose that this is the case, and let P be a shortest $p-q$ path (which has length at most r). As p and q are in two different components of $G-\Gamma$, the path P has to intersect Γ at some vertex $v \in \Gamma$ and we have $\text{dist}(p,q) = \text{dist}(p,v) + \text{dist}(v,q)$. We know that there has to be a $p' \in Q$ such that $\text{dist}(p',v) \leq \text{dist}(p,v)$, otherwise p would be banned and it would not be in $\mathcal{D}'$. Then the same calculation as in the geometric case shows that $p' \in Q$ also covers q, which means that we removed q and it cannot be in $\mathcal{C}'$. Thus we can indeed obtain a solution by solving the subinstances restricted to the components of $G-\Gamma$. As in the case of packing, we have at most $d \cdot \frac{2}{3}k \cdot d^{\mathcal{O}(\sqrt{k})}$ subproblems and the running time $d^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ follows the same way.

Covering in planar graphs (maximization version). Let us consider now the variant of the previous problem where we want to select k vertices from $\mathcal{D}$ that cover the maximum number of vertices in $\mathcal{C}$. We proceed the same way as before: we guess a guarded separator (Q,Γ), remove from $\mathcal{C}$ the set of vertices covered by Q (let m be the number of these vertices), remove from $\mathcal{D}$ the set of vertices banned by (Q,Γ), and recursively solve the problem for every component C of $G-\Gamma$ and every $k'=1,\dots,\lfloor\frac{2}{3}k\rfloor$. Having solved these subproblems, we have at our hand the values $m(C,k')$, the maximum number of vertices in $\mathcal{C}'[C]$ that can be covered by k' vertices from $\mathcal{D}'[C]$. How can we compute from these values the maximum number of vertices in $\mathcal{C}$ that can be covered by k vertices from $\mathcal{D}$? We need to solve a knapsack-type problem: for each component C, we have to select a solution containing a certain number $0 \leq k_C \leq \frac{2}{3}k$ vertices of $\mathcal{D}$ such that the sum of the k_C's is at most $k-|Q|$ and the sum of the values $m(C,k_C)$ is maximum possible. Given the values $m(C,k')$, this maximum can be computed by a standard polynomial-time dynamic programming algorithm. Thus instead of just deciding if k vertices from $\mathcal{D}$ are sufficient to cover every vertex in $\mathcal{C}$, we can also find a set of k vertices covering the maximum number of vertices from $\mathcal{C}$. Let us point out that in this problem it is really essential that we solve the subinstances for each value of

k' instead of just finding a minimum/maximum cardinality solution: as the optimum solution may not cover all vertices in a component C, computing the minimum number of vertices required to cover all vertices in C is clearly not sufficient.

Covering in planar graphs (nonuniform radius). A natural generalization of the covering problem is when every vertex $p \in \mathcal{D}$ is given a radius $r(p) \geq 0$ and now we say that p covers a vertex $q \in \mathcal{C}$ if $\text{dist}(p, q) \leq r(p)$. That is, now the vertices in $\mathcal{D}$ represent balls with possibly different radii.

There are two ways in which we can handle this more general problem. The first is a simple graph-theoretic trick. Let R be the maximum of all the $r(p)$'s. For every $p \in \mathcal{D}$, let us attach a path of length $R - r(p)$ to p, let us replace p in $\mathcal{D}$ with the other end p' of this path. Observe that now a vertex $q \in \mathcal{C}$ is at distance at most $r(p)$ from p if and only if it is at distance at most R from p'. Therefore, by solving the resulting covering problem with uniform radius R, we can solve the original problem as well. Note that, interestingly, this trick uses the flexibility of the problem being stated in terms of graphs and there is no analogous geometric trick to solve the problem of covering points with disks of nonuniform radius in the plane: by attaching the paths, we are distorting the distance metric in a specific way, whose geometric meaning is difficult to interpret in the 2-dimensional plane.

The second way is somewhat more complicated, but it seems to be the robust mathematical solution of the issue and it has also a geometric interpretation. The issue of nonuniform radius can be handled by working with the *additively weighted* version of the Voronoi diagram, that is, instead of defining the Voronoi regions of $\mathcal{P}$ by comparing the distances $\text{dist}(p, v)$ for $p \in \mathcal{P}$, we compare the weighted distances $\text{dist}(p, v) - r(p)$. It can be verified that the main arguments of the algorithm described above go through. For example, it remains true that Voronoi regions have the convexity property that if v is in the region of p, then the shortest path from v to p is fully contained in the Voronoi region of p. Given a guarded separator (Q, Γ), the crucial property allowing us to separate the problem to the components of $G - \Gamma$ was that if $p \in \mathcal{D}$ and $q \in \mathcal{C}$ are in two different components and p covers q, then some $p' \in Q$ also covers q. This property also remains true: we can redo the same calculation to compare $\text{dist}(p, q) - r(p)$ and $\text{dist}(p', q) - r(p')$.

Combining covering and packing. Comparing the algorithms for packing objects in planar graphs and for covering vertices by balls of radius r in planar graphs, we can observe that the set $\mathcal{D}$ played very different roles in the two algorithms: in the packing algorithm $\mathcal{D}$ contained the actual objects, while in the covering algorithm $\mathcal{D}$ contained only the centers of the balls. The reason for this is fundamental: in order to define the Voronoi regions of the solution $\mathcal{F} \subseteq \mathcal{D}$, we need $\mathcal{F}$ to be a set of disjoint objects (which is not true for the balls in the solution of the covering problem, but of course true for the centers of these balls). This means that we cannot generalize the covering algorithm to objects different from metric balls simply by putting objects into $\mathcal{D}$ more general than single vertices. However, we have no trouble obtaining a common generalization if we require that the objects selected from $\mathcal{D}$ are disjoint. That is, in the *independent covering* problem, we are given a set $\mathcal{D}$ of connected objects in a planar graph G, a set $\mathcal{C}$ of vertices, and integers k and r, the task is to select a *pairwise disjoint* subset $\mathcal{F} \subseteq \mathcal{D}$ of size exactly k that covers the maximum number of vertices in $\mathcal{C}$ (where, as usual, an object $p \in \mathcal{D}$ covers a vertex $q \in \mathcal{C}$ if $\text{dist}(p, q) \leq r$). Exactly the same algorithm as above goes through: as the solution $\mathcal{F}$ is disjoint, it is possible to define the Voronoi regions.

In the next section, we define the Disjoint Network Coverage problem that generalizes all our applications. The main algorithmic result is expressed as an algorithm for this problem (Theorem 3.1). To handle covering with nonuniform radius, the problem formulation allows different radius $r(p)$ for each $p \in \mathcal{D}$. However, if we allow nonuniform radius, then the requirement stating that we have to select disjoint objects should be replaced by a technical condition that we have to select a *normal* family of objects (see Section 3.1). Essentially, this condidition states that in the solution $\mathcal{F}$, every vertex of every object $p \in \mathcal{F}$ should be in the weighted Voronoi region of p. In other words, if we allow arbitrary radii, it could in principle happen that one object p has so large radius compared to some other object p' that the weighted Voronoi region of p actually contains some vertices of p', thus "eating" the object p' itself and making it not contained in its own Voronoi region. The normality condition states that, in addition to being disjoint, the solution $\mathcal{F}$ does not contain such pathological situations. Observe that when the objects are single vertices only, then this problem is not very dangerous: if p "dominates" p' in the way described above, then every client covered by p' is also covered by p, and we can discard p' from $\mathcal{D}$ because taking p instead is always more profitable. Unfortunately, if the objects are not just single vertices, or they are equipped with nonuniform costs, then this technicality makes the arising independent covering problems with nonuniform radii somewhat unnatural.

3 The general problem

In this section we introduce the generic problem Disjoint Network Coverage. The main result of this paper is an algorithm for this problem, expressed in Theorem 3.1. Before we give the algorithm for Disjoint Network Coverage, in Section 3.2 we shall see how the concrete results mentioned in the introduction (i.e., Theorem 1.1-1.7) follow from Theorem 3.1 by simple reductions.

3.1 Problem definition

Suppose we are given an undirected graph G embedded on a sphere Σ, together with a positive edge weight function $\mathbf{w}\colon E(G) \to \mathbb{R}^+$. We are given a family of *objects* $\mathcal{D}$ and a family of *clients* $\mathcal{C}$.

Every object $p \in \mathcal{D}$ has three attributes. It has its *location* $\mathbf{loc}(p)$, which is a nonempty subset of vertices of G such that $G[\mathbf{loc}(p)]$ is a connected graph. Moreover, it has its *cost* $\lambda(p)$, which is a real number. Note that costs may be negative. Finally, it has its *radius* $r(p)$, which is just a nonnegative real value denoting the strength of domination imposed by p. Note that locations of objects can intersect, and even there can be multiple objects with exactly the same location.

Every client $q \in \mathcal{C}$ has three attributes. It has its *placement* $\mathbf{pla}(q)$, which is just a vertex of G where the client resides. It has also its *sensitivity* $s(q)$, which is a real value that denotes how sensitive the client is to domination from objects. Finally, it has also *prize* $\pi(q)$, which is a real value denoting the prize gained by dominating the client. Note that there can be multiple clients placed in the same vertex, and the prizes may be negative.

We say that a subfamily $\mathcal{F} \subseteq \mathcal{D}$ is *normal* if locations of objects from $\mathcal{F}$ are disjoint, and moreover $\mathrm{dist}(\mathbf{loc}(p_1), \mathbf{loc}(p_2)) > r(p_1) - r(p_2)$ for all pairs (p_1, p_2) of different objects in $\mathcal{F}$; here $\mathrm{dist}(X, Y)$ denotes the distance between two vertex sets in G w.r.t. edge weights $\mathbf{w}$. As we require the same inequality for the pair (p_2, p_1) as well, it follows that actually $\mathrm{dist}(\mathbf{loc}(p_1), \mathbf{loc}(p_2)) > |r(p_1) - r(p_2)|$ is true. In particular, this implies disjointness of locations of objects from $\mathcal{F}$, but if all radii are equal, then normality boils down to just disjointness of locations. Note also that a subfamily of a normal family is also normal.

We say that a client q is *covered* by an object p if the following holds: $\mathrm{dist}(\mathbf{pla}(q), \mathbf{loc}(p)) \leq s(q) + r(p)$. In other words, a client gets covered by an object if the distance from the object does not exceed the sum of its sensitivity and the object's domination radius.

We are finally ready to define Disjoint Network Coverage. As input we get an edge-weighted graph G embedded on a sphere, families of objects $\mathcal{D}$ and clients $\mathcal{C}$ (described using locations, costs, radii, placements, sensitivities, and prizes), and a nonnegative integer k. For a subfamily $\mathcal{Z} \subseteq \mathcal{D}$, we define its *revenue*, denoted $\Pi(\mathcal{Z})$, as the total sum of prizes of clients covered by at least one object from $\mathcal{Z}$ minus the total sum of costs of objects from $\mathcal{Z}$. In the Disjoint Network Coverage problem, the task is to find a subfamily $\mathcal{Z} \subseteq \mathcal{D}$ such that the following holds:

(i) Family $\mathcal{Z}$ is normal and has cardinality *exactly* k.

(ii) Subject to the previous constraint, family $\mathcal{Z}$ maximizes the revenue $\Pi(\mathcal{Z})$.

It can happen that there is no subfamily $\mathcal{Z}$ satisfying property (i). In this case, value $-\infty$ should be reported by the algorithm. For an instance $(G, \mathcal{D}, \mathcal{C}, k)$ of Disjoint Network Coverage, we denote $d = |\mathcal{D}|$, $c = |\mathcal{C}|$, and $n = |V(G)|$. Observe that we can assume that $k \leq d$ and that the input graph G is simple: loops can be safely removed, and it is safe to only keep the edge with the smallest weight from any pack of parallel edges.

Note that if we do not provide any clients in the problem and we set all the radii to be equal to 0, then we arrive exactly at the problem of packing disjoint objects in the graph. However, by introducing also clients we can ask for (partial) domination-type constraints in the graph, as well as define prize-collecting objectives. In this manner, Disjoint Network Coverage generalizes both packing problems as well as covering problems. The caveat is, however, that we need to require that the objects that cover clients are pairwise disjoint, and moreover they have to form a normal family.

The main result of this paper is the following theorem.

Theorem 3.1 (Main result). Disjoint Network Coverage *can be solved in time* $d^{\mathcal{O}(\sqrt{k})} \cdot (cn)^{\mathcal{O}(1)}$.

3.2 Applications of Theorem 3.1

In this section we provide formal argumentation of how the concrete results mentioned in the introduction follow from Theorem 3.1. While for problems on planar graphs the reductions are straightforward, for geometric problems some non-trivial technicalities arise. We remark that we are proving the exact statements given in the introduction, but the generality of the Disjoint Network Coverage problem would equally easily allow us to solve more general variants with different costs, prizes, sensitivities of clients, etc.

Applications to planar graphs: Theorems 1.1-1.3.

Restated Theorem 1.1 (packing connected sets). *Let G be a planar graph, $\mathcal{D}$ be a family of connected vertex sets of G, and k be an integer. In time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$, we can find a set of k pairwise disjoint objects in $\mathcal{D}$, if such a set exists.*

Theorem 1.1 follows by taking $\mathcal{D}$ to be the set of objects (identified with their locations), assigning each of them radius and cost equal to 0, and considering no clients (putting $\mathcal{C} = \emptyset$). As distances play no role in the problem, we can simply put unit weights.

Restated Theorem 1.2 (covering vertices with centers of different radii). *Let G be a planar graph, let $D, C \subseteq V(G)$ be two subset of vertices, let $r\colon D \to \mathbb{Z}^+$ be a function, and k be an integer. In time $|D|^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$, we can find a set $S \subseteq D$ of k vertices that maximizes the number of vertices covered in C, where a vertex $u \in C$ is covered by $v \in S$ if the distance of u and v is at most $r(v)$.*

For Theorem 1.2, for every $v \in D$ we construct an object p_v with $\mathbf{loc}(p_v) = \{v\}$, $\lambda(v) = 0$ and $r(p_v) = r(v)$. Then, for every $u \in C$ we construct a client q_u with $\mathbf{pla}(q_u) = u$, $s(q_u) = 0$ and $\pi(q_u) = 1$. Let $\mathcal{D}, \mathcal{C}$ be the sets of constructed objects and clients, respectively. We claim that the optimum value for the input instance is equal to the maximum among optimum revenues for instances $(G, \mathcal{D}, \mathcal{C}, k')$ of Disjoint Network Coverage, for $0 \le k' \le k$. On one hand, any solution to any such instance $(G, \mathcal{D}, \mathcal{C}, k')$ trivially yields a solution to the input instance with the same revenue. On the other hand, if $S \subseteq D$ is a solution to the input instance, then observe that without loss of generality we may assume that S does not contain any two distinct vertices v_1, v_2 such that $\mathrm{dist}(v_1, v_2) \le r(v_1) - r(v_2)$. This is because when $\mathrm{dist}(v_1, v_2) \le r(v_1) - r(v_2)$, then any client covered by v_1 is also covered by v_2, and v_1 can be safely removed from S. Then a solution S with this property corresponds to a subfamily $\mathcal{Z} \subseteq \mathcal{D}$ that is normal and has size $|S| \le k$, i.e., it is a solution for $(G, \mathcal{D}, \mathcal{C}, |S|)$ with revenue equal to the number of vertices of C covered by S. We conclude that to solve the input instance it suffices to solve all the instances $(G, \mathcal{D}, \mathcal{C}, k')$ of Disjoint Network Coverage for $0 \le k' \le k$, using the algorithm of Theorem 3.1.

Restated Theorem 1.3 (covering vertices with independent objects). *Let G be a planar graph, let $\mathcal{D}$ be a set of connected vertex sets in G, let $C \subseteq V(G)$ be a set of vertices, and let k be an integer. In time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$, we can find a set S of at most k pairwise disjoint objects in $\mathcal{D}$ that maximizes the number of vertices of C in the union of the vertex sets in S.*

Theorem 1.3 follows by taking $\mathcal{D}$ to be the set of objects (identified with their locations), assigning each of them radius and cost equal to 0, and for every $u \in C$ introducing a client q_u with $\mathbf{pla}(q_u) = u$, $s(q_u) = 0$ and $\pi(q_u) = 1$. As distances play no role in the problem, we can simply put unit weights.

Applications to geometric problems: Theorems 1.4-1.7. In all the geometric reductions that follow, we perform arithmetic operations only of the following types:

- Computing new points on the plane with coordinates given as constant-size rational expressions (i.e., yielded by the four basic arithmetic operations) over the input coordinates and radii.
- Computing Euclidean distances between the obtained points.

Note that finding intersection of two segments between pairs of points given on the input gives a point that is compliant with this requirement.

Thus, if the input coordinates and radii are s-bit integers, then the coordinates of the points obtained in the reductions can be stored as rationals represented using $\mathrm{poly}(s)$ bits. Then the distances can be safely stored using $\mathrm{poly}(s, n)$-bit precision in order to avoid rounding errors when adding at most n of them. Similarly, if the input coordinates and radii are given as floating point numbers with s bits before the point and s bits after the point, then we can scale them up to get $2s$-bit integers, and perform the same analysis.

Restated Theorem 1.4 (packing disks). *Given a set $\mathcal{D}$ of disks (of possibly different radii) in the plane, in time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})}$ we can find a set of k pairwise disjoint disks, if such a set exists.*

Proof. We provide a reduction to the problem considered in Theorem 1.1. Let $I = (\mathcal{D}, k)$ be the input instance. Based on the disk set $\mathcal{D}$, we define a planar graph G as follows. First, construct a set of points X (see Figure 3) by taking (a) all the centers of disks from $\mathcal{D}$, and (b) for any two disks A_1, A_2 that intersect, an arbitrarily chosen point in their intersection (for example, a point on the segment between the centers for which distances to the centers are in the same ratio as radii of the disks). Start the construction of G by putting $V(G) = X$. Then, for each pair of points $x_1, x_2 \in X$, draw a segment between x_1 and x_2 on the plane, provided that no other vertex of X lies on this segment. These segments form so far the edge set of G, but they may cross. To make G planar, for every crossing of two segments introduce a new vertex at the intersection point, and

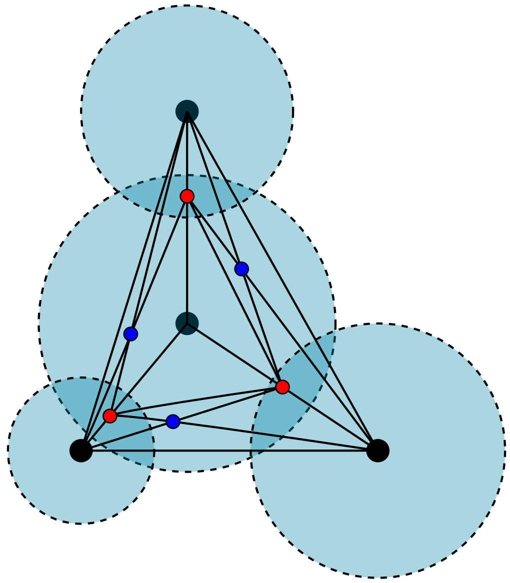


Figure 3: Proof of Theorem 1.4. The black dots are the centers of the disks, the red dots are the points chosen in the intersections of pairs of disks, and the blue dots were introduced at the intersections of the segments to make the graph planar.

connect all the old and new points according to the segments. That is, every former edge between two points $x_1, x_2 \in X$ becomes a path from x_1 to x_2 traversing consecutive crossing points, and the embedding of this path is exactly the segment between x_1 and x_2. To make G edge-weighted, to every edge $uv \in E(G)$ assign weight equal to the Euclidean length of the segment between u and v. Observe that G is connected and $|V(G)| \leq \mathcal{O}(|\mathcal{D}|^4)$.

Now, for every disk $A \in \mathcal{D}$ with radius r_A construct a vertex set $X(A)$ as follows: $X(A)$ comprises all the vertices of G that are at distance at most r_A from the center of A (which belongs to $V(G)$) in the graph G. Obviously $G[X(A)]$ is connected, because it is defined as a ball in the graph G. Moreover, by the triangle inequality it follows that every vertex of $X(A)$ is actually embedded into the disk A. Note, however, that $X(A)$ may not contain some intersection points that are actually embedded into the disk A, but in G they are at distance larger than r_A from the center of A. Nonetheless, $X(A)$ contains all the elements of X that are embedded into the disk A. This is because by the construction of G, for any $x \in X \cap A$ we have that x is connected to the center of A via a path consisting of parallel segments, which in particular has length equal to the Euclidean distance from x to this center.

Let $\mathcal{D}_0 = \{X(A) \mid A \in \mathcal{D}\}$ be the family of constructed vertex subsets, and let $I' = (G, \mathcal{D}_0, k)$ be the constructed instance of the problem considered in Theorem 1.1. We now claim that the input instance I has a solution if and only if the constructed instance I' has a solution.

On one hand, if $\mathcal{S} \subseteq \mathcal{D}$ is a subfamily of k disjoint disks, then $\mathcal{S}_0 = \{X(A) \mid A \in \mathcal{S}\}$ is a subfamily of k vertex sets that are pairwise disjoint, and hence I' has some solution. On the other hand, suppose there exists some solution $\mathcal{S}_0$ to I', and let $\mathcal{S} \subseteq \mathcal{D}$ be the corresponding set of k disks. We claim that these disks are pairwise disjoint. Suppose that, on the contrary, there are some distinct disks $A_1, A_2 \in \mathcal{S}$ that intersect. Recall that for these disks we have added some point $x \in A_1 \cap A_2$ to X. As we have argued, by the construction of G this means that x belongs both to $X(A_1)$ and to $X(A_2)$, which is a contradiction with the fact the vertex sets in $\mathcal{S}_0$ are pairwise disjoint. $\square$

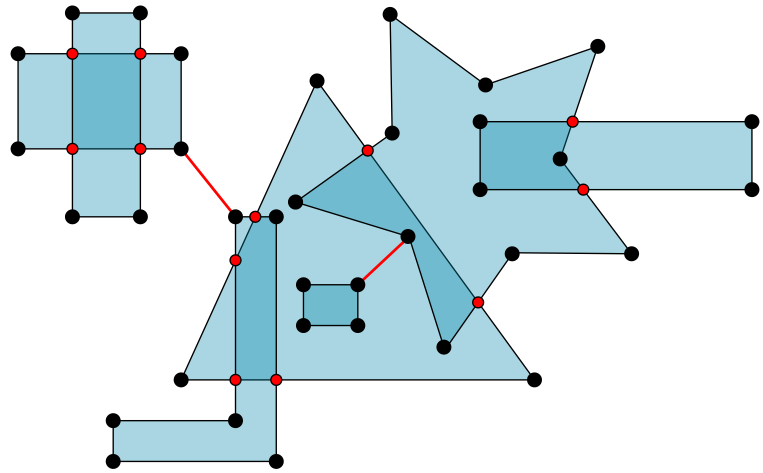


Figure 4: Proof of Theorem 1.5. The black dots are the vertices of the polygons and the red dots are the points introduced at the intersection of segments. The red segments were introduced to make the graph connected.

Restated Theorem 1.5 (packing simple polygons). *Given a set $\mathcal{D}$ of simple polygons in the plane, in time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ we can find a set of k polygons in $\mathcal{D}$ with pairwise disjoint closed interiors, if such a set exists. Here n is the total number of vertices of the polygons in $\mathcal{D}$.*

Proof. We give a reduction to the problem considered in Theorem 1.1. Let $I = (\mathcal{D}, k)$ be the input instance. Based on the set of polygons $\mathcal{D}$, define a planar graph G as follows. Start with taking $V(G)$ to be the set of all the vertices of all the polygons of $\mathcal{D}$, and then for every polygon $A \in \mathcal{D}$ draw all the sides of A as edges of G, embedded as respective segments (see Figure 4). Then perform a similar construction as in the proof of Theorem 1.4: in order to make G planar, for every crossing of two segments introduce a vertex on their intersection. The edge set of G is defined as the set of all the segments xy between two vertices $x, y \in V(G)$, being either the original vertices of the polygons or the introduced crossing points, such that xy is contained in some side of an original polygon and the interior of xy does not contain any other vertex of $V(G)$. Thus, this definition yields a planar embedding of G. Note that the definition is also correct when we have two sides xy and $x'y'$ of two different polygons that share a common subinterval (not being a single point); then vertices x, y count as crossing points subdividing $x'y'$ and vice versa.

Observe that G defined in this way is planar, but may not be connected. Hence, we make G connected by iteratively taking two connected components C_1, C_2 of G that are incident to a common face, and adding an edge within this face between two arbitrarily chosen vertices from C_1 and C_2. Observe that $|V(G)| = \mathcal{O}(n^2)$.

Now, for every polygon $A \in \mathcal{D}$ construct a vertex set $X(A)$ as follows: $X(A)$ comprises all the vertices of G that are embedded into the polygon A. We regard polygons as closed, so $X(A)$ in particular contains all the vertices of A. Note that the perimeter of A naturally induces a simple cycle in G, and $X(A)$ is exactly the set of points enclosed or on this cycle. Since G is connected, it also follows that $G[X(A)]$ is connected. Let $\mathcal{D}_0 = \{X(A) \mid A \in \mathcal{D}\}$ be the set of constructed vertex sets, and let $I' = (G, \mathcal{D}_0, k)$ be the constructed instance of the problem considered in Theorem 1.1. We now claim that the input instance I has a solution if and only if the constructed instance I' has a solution.

On one hand, if $\mathcal{S} \subseteq \mathcal{D}$ is a subfamily of k disjoint polygons, then $\mathcal{S}_0 = \{X(A) \mid A \in \mathcal{S}\}$ is a subfamily of k vertex sets that are pairwise disjoint, and hence I' has some solution. On the other hand, suppose there exists some solution $\mathcal{S}_0$ to I', and let $\mathcal{S} \subseteq \mathcal{D}$ be the corresponding set of polygons. We claim that these polygons are pairwise disjoint. Suppose that, on the contrary, there are some distinct polygons $A_1, A_2 \in \mathcal{S}$ that intersect. If the perimeters of A_1 and A_2 intersect, then each their intersection point x belongs to $V(G)$ and is contained in $X(A_1) \cap X(A_2)$, contradicting the fact that $X(A_1)$ and $X(A_2)$ are disjoint. Otherwise, either A_1 is entirely contained in A_2, or A_2 is entirely contained in A_1. In the former case every vertex of A_1 belongs to $V(G)$ and is contained in $X(A_1) \cap X(A_2)$, and in the latter case every vertex of A_2 has this property. In both cases we obtain a contradiction with the fact that $X(A_1)$ and $X(A_2)$ are disjoint. ☐

Restated Theorem 1.6 (covering points with disks). *Given a set $\mathcal{C}$ of points and a set $\mathcal{D}$ of disks (of possibly different radii) in the plane, in time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot |\mathcal{C}|^{\mathcal{O}(1)}$ we can find a set of k disks in $\mathcal{D}$ maximizing the total number of points they cover in $\mathcal{C}$.*

Restated Theorem 1.7 (covering points with squares). *Given a set $\mathcal{C}$ of points and a set $\mathcal{D}$ of axis-parallel squares (of possibly different size) in the plane, in time $|\mathcal{D}|^{\mathcal{O}(\sqrt{k})} \cdot |\mathcal{C}|^{\mathcal{O}(1)}$ we can find a set of k squares in $\mathcal{D}$ maximizing the total number of points they cover in $\mathcal{C}$.*

Proof. We prove both theorems at the same time by showing that the algorithm can be constructed whenever sets from $\mathcal{D}$ are defined as balls in some norm $||\cdot||$ on $\mathbb{R}^2$. More precisely, there exists a norm $||\cdot||$ on $\mathbb{R}^2$, such that every $A \in \mathcal{D}$ is defined as $A = \{x \in \mathbb{R}^2 \mid ||x - c(A)|| \leq r(A)\}$, where $c(A)$ is the center of ball A and $r(A)$ is its radius. Then Theorem 1.6 follows by taking $||\cdot||$ to be the ℓ_2-norm and Theorem 1.7 follows by taking $||\cdot||$ to be the ℓ_∞-norm.

We give a reduction to the problem considered in Theorem 1.2, and to this end we perform a similar construction as in the proof of Theorem 1.4. Let $I = (\mathcal{D}, \mathcal{C}, k)$ be the input instance. Let X be the set of all the centers of balls from $\mathcal{D}$ and all the points from $\mathcal{C}$. Start the construction of G by putting $V(G) = X$. Then, for each pair of points $x_1, x_2 \in X$ such that x_1 is a center of a ball from $\mathcal{D}$ and x_2 is a point from $\mathcal{C}$, draw a segment between x_1 and x_2 on the plane, provided that no other vertex of X lies on this segment. These segments form so far the edge set of G, but they may cross. To make G planar, for every crossing of two segments introduce a new vertex at the intersection point, and connect all the old and new points naturally according to the segments. To make G edge-weighted, observe that every edge e of G is embedded as a segment on the plane, so put $\mathbf{w}(e)$ to be the $||\cdot||$-length of this segment. Observe that G is connected and $|V(G)| = \mathcal{O}(|\mathcal{D}|^2|\mathcal{C}|^2)$.

Now, let us set $D = \{c(A) \mid A \in \mathcal{D}\}$ and $C = \mathcal{C}$. Construct an instance I' of the problem considered in Theorem 1.2 by taking graph G with subsets of vertices D and C, assigning $r(c(A)) = r(A)$ for each $c(A) \in D$, and putting budget k. We now claim that a vertex $x \in C$ is covered by some $c(A) \in D$ in the instance I' if and only if point $x \in \mathcal{C}$ belongs to $A \in \mathcal{D}$ in the input instance I. To prove this it suffices to show that $\mathrm{dist}_G(x, c(A)) = ||x - c(A)||$, since x is covered by $c(A)$ in I' if and only if $\mathrm{dist}_G(x, c(A)) \leq r(A)$, whereas x belongs to A in I if and only if $||x - c(A)|| \leq r(A)$. First, by the triangle inequality it follows that distances in G are lower bounded by distances w.r.t. norm $||\cdot||$ on the plane, so $\mathrm{dist}_G(x, c(A)) \geq ||x - c(A)||$. On the other hand, since $x, c(A) \in X$, then by the construction of G there exists a path in G that connects x and $c(A)$ and consists of parallel segments. In particular, this path has total length $||x - c(A)||$, which means that $\mathrm{dist}_G(x, c(A)) \leq ||x - c(A)||$. We conclude that indeed $\mathrm{dist}_G(x, c(A)) = ||x - c(A)||$.

Solutions to the input instance I correspond one-to-one to solutions of the constructed instance I', and the argumentation of the previous paragraph shows that this correspondence preserves the number of points covered by the solution. Hence, it suffices to run the algorithm of Theorem 1.2 on the instance I'. ☐

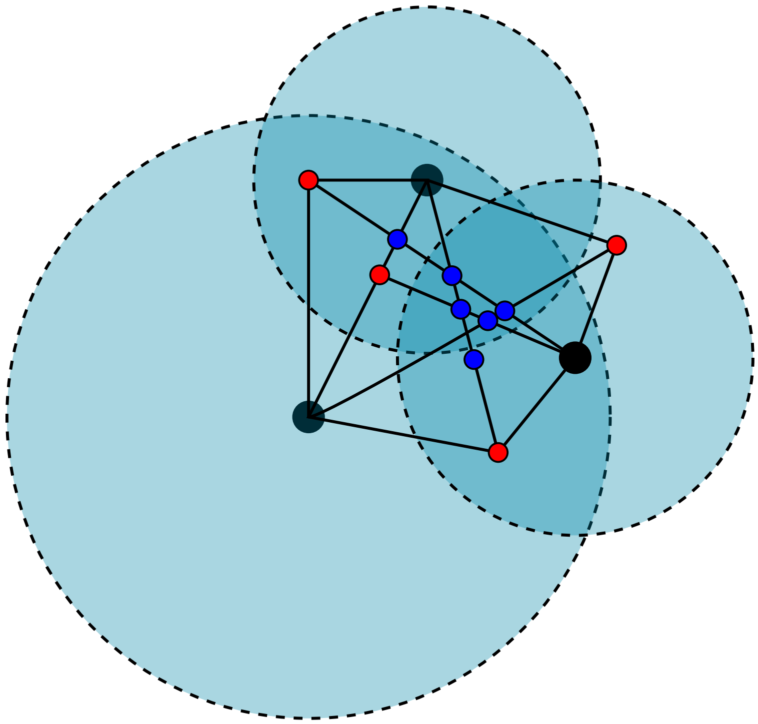


Figure 5: Proof of Theorem 1.6. The black dots are the centers of the disks in $\mathcal{D}$, the red dots are in the points in $\mathcal{C}$, and the blue dots were introduced at the intersections of the segments to make the graph planar.

4 The main algorithm: proof of Theorem 3.1

4.1 Notation and general definitions

We first establish notation and recall some general definitions that will be used throughout the proof.

For a graph G, by $V(G)$ we denote its vertex set and by $E(G)$ we denote its edge set. Graph G is *simple* if every edge connects two different vertices, and every pair of vertices is connected by at most one edge. On the other hand, G is a *multigraph* if we allow (a) *parallel edges*, that is, multiple edges connecting the same pair of vertices, and (b) *loops*, that is, edges that connect some vertex u with itself. A graph *induced* by a vertex subset $X \subseteq V(G)$, denoted $G[X]$, has X as the vertex set, and its edge set comprises all the edges of G whose both endpoints belong to X. A graph *spanned* by an edge subset $F \subseteq E(G)$, denoted $G[F]$, has F an the edge set, and its vertex set comprises all the vertices of G incident with at least one edge of F.

In this work we will be mostly working with edge-weighted graphs. That is, we assume that the input graph G is given together with a positive edge weight function $\mathbf{w}\colon E(G) \to \mathbb{R}^+$. The *distance* between two vertices $u, v \in V(G)$, denoted $\mathrm{dist}(u, v)$, is defined as the minimum total weight of a path connecting u and v. We extend this notation to subsets in the obvious manner: $\mathrm{dist}(u, Y)$ is the minimum weight of a path from u to any vertex of Y, and $\mathrm{dist}(X, Y)$ is the minimum weight of a path from any vertex of X to any vertex of Y. Note that the $\mathrm{dist}(\cdot, \cdot)$ function satisfies the triangle inequality: $\mathrm{dist}(u, v) + \mathrm{dist}(v, w) \geq \mathrm{dist}(u, w)$, and similar inequalities hold for sets as well.

Let Σ be the sphere $S^2 = \{(x, y, z) \in \mathbb{R}^3 \mid x^2 + y^2 + z^2 = 1\}$. A *curve* on Σ is a homeomorphic image of the interval $[0, 1]$, and a *closed curve* on Σ is a homeomorphic image of the circle

$S^1 = \{(x,y) \in \mathbb{R}^2 \mid x^2+y^2=1\}$. Note that, in particular, a curve and a closed curve do not have self-crossings, i.e., no point of Σ is visited twice. The Jordan curve theorem states that for any closed curve γ on Σ, $\Sigma \setminus \gamma$ consists of two connected sets homeomorphic to open disks. A *sphere embedding* of a graph G is a mapping that maps vertices of G to distinct points of Σ, and edges of G to curves connecting respective endpoints that do not intersect apart from their endpoints. In case $e \in E(G)$ is a loop, the corresponding curve connects a point with itself, i.e., it is a closed curve. Note that this definition is also valid for multigraphs. It is well known that a graph is planar, i.e., it can be embedded on the 2-dimensional plane in the same manner, if and only if it can be embedded on a sphere.

If G is a sphere-embedded (multi)graph, then a *face* of G is an inclusion-wise maximal connected set that does not contain any point of the embedding of G. Note that in particular every face is open. It is well known that if G is connected, then every face is homeomorphic to an open disk. Note, however, that if G has bridges or cutvertices, then there may be faces whose closures are not homeomorphic to closed disks. If the closure of each face is homeomorphic to a closed disk, then we say that the embedding is a *2-cell embedding*. The set of faces of a (sphere-embedded) graph G is denoted by $F(G)$. The well-known Euler formula states that for every connected multigraph G embedded on a sphere, it holds that $|V(G)|-|E(G)|+|F(G)|=2$.

Let $f \in F(G)$ be a face of a connected, sphere-embedded multigraph G. We say that a vertex v or an edge e is *incident* with f if it is contained in the closure of f. By the *boundary of f*, denoted ∂f, we mean a walk in G that visits consecutive vertices and edges incident with f in the order of their appearance around f. Note that if the closure of f is homeomorphic to a closed disk, then ∂f is a simple cycle in G. However, otherwise a vertex can be visited more than once on ∂f (this can happen if it is a cutvertex in G), and an edge can be traversed more than once on ∂f (this can happen if it is a bridge in G). In case v is a cutvertex of G, then there is a face f of G that appears multiple times around v in the planar embedding. Whenever some curve is leaving v to f (or entering v from f), we will distinguish these different appearances and call then *directions*. Note that every such direction corresponds to an appearance of v on ∂f.

A sphere-embedded graph is *triangulated*, if for every face f, walk ∂f is a triangle in G; in particular, the graph is simple and the embedding is a 2-cell embedding. It is well-known that for any simple graph G embedded on a sphere, one can add edges to the embedding so that the new supergraph is triangulated.

4.2 Simplifying assumptions

Throughout the proof we assume that $(G, \mathcal{D}, \mathcal{C}, k)$ is the input instance of Disjoint Network Coverage. Before we proceed to the proof, let us make a few simplifying assumptions about the input instance $(G, \mathcal{D}, \mathcal{C}, k)$. These assumptions can be made without loss of generality, and they will streamline further reasonings.

Firstly, we will assume that the graph G is connected. Indeed, for disconnected graph we can apply the algorithm for every connected component of G separately, for all the parameters $i = 0, 1, 2, \ldots, k$, and then merge the results using a simple dynamic programming algorithm.

Secondly, we will assume that all the values $\mathrm{dist}(u,v)$ and $\mathrm{dist}(u,v) - r(p)$ for $u, v \in V(G)$ and $p \in \mathcal{D}$ are pairwise different, and moreover that the shortest paths between pairs of vertices in G are unique. This can be easily obtained using the standard technique of breaking ties by adding very small, distinct values to the weights of edges as well as to the radii of objects. To do this, we need to use $(d+n)^{\mathcal{O}(1)}$ more bits of precision in the representation of floating point numbers. Since we never estimate precisely the exact running time of polynomial-time subroutines run on G, this does not influence the claimed asymptotic running time of the algorithm. For brevity, we will denote this assumption by (♣).

Thirdly, we fix some sphere embedding of G, and without loss of generality we assume that G is triangulated. This can be achieved by triangulating every face of G using edges of weight $+\infty$ (equivalently, a value much larger than all the radii and sensitivities). After this operation, every face of G is a triangle whose closure is homeomorphic to a closed disc (i.e., it is a 2-cell embedding). Note that thus G is 2-connected, so in particular it does not have any cut-vertices, bridges, or vertices of degree 1. We also assume that G has more than 3 vertices, since we can always add a new vertex within a triangular face connected with its vertices via edges of weight $+\infty$.

Finally, for every object $p \in \mathcal{D}$ we arbitrarily distinguish one vertex $\mathbf{cen}(p) \in \mathbf{loc}(p)$, called the *center* of object p. We also pick an arbitrary spanning tree $T(p)$ of $G[\mathbf{loc}(p)]$, rooted at $\mathbf{cen}(p)$. For concreteness, the reader may think of $T(p)$ as the tree of shortest paths from $\mathbf{cen}(p)$ in $G[\mathbf{loc}(p)]$, but we will never use this property.

4.3 Voronoi partitions

We want to define analogs of Voronoi regions and related notions in the context of planar graphs: given a set $\mathcal{F}$ of objects, we would like to partition the vertices of the graph according to which object in $\mathcal{F}$ is closest to a vertex. For this to make sense, we clearly need the assumption that the objects in $\mathcal{F}$ are pairwise disjoint, otherwise it is not clear how to classify vertices in the intersection of the objects. Moreover, to express the fact that different objects have different power of domination ($r(p)$ can be different for different objects), we need additively weighted Voronoi regions. That is, instead of comparing the distances to locations of objects from $\mathcal{F}$, we compare the values of distances to $\mathbf{loc}(p)$ minus $r(p)$, for $p \in \mathcal{F}$. Here we need the technical assumption that the family $\mathcal{F}$ is normal, which rules out degenerate situations by ensuring that every point of the object is in the Voronoi region of the object itself, and hence the Voronoi regions are nonempty and connected.

Formally, suppose $\mathcal{F} \subseteq \mathcal{D}$ is a normal subfamily of objects. Then we define the *Voronoi partition* of G with respect to $\mathcal{F}$, denoted $\mathbb{M}_{\mathcal{F}}$, as a partition of $V(G)$ into parts $\{M_p\}_{p\in\mathcal{F}}$ as follows: a vertex $v \in V(G)$ belongs to part M_{p_0} if and only if $\mathrm{dist}(v, \mathbf{loc}(p_0)) - r(p_0)$ is the smallest value among $\{\mathrm{dist}(v, \mathbf{loc}(p)) - r(p)\}_{p\in\mathcal{F}}$. Note that assumption (♣) implies there can be no ties between different objects. Set M_p shall be called the *Voronoi region* of p. For some Voronoi partition $\mathbb{M}_{\mathcal{F}}$ and $p \in \mathcal{F}$, the Voronoi region of p will be also denoted by $\mathbb{M}_{\mathcal{F}}(p)$. Whenever the family we are working with will be clear from the context, we shall drop the index $\mathcal{F}$. This remark holds for all the objects defined in the sequel that depend on the currently considered family $\mathcal{F}$.

The following simple lemma shows that the normality condition is a sanity check that the Voronoi regions are well-defined.

Lemma 4.1. *Suppose $\mathcal{F} \subseteq \mathcal{D}$ is a normal subfamily of objects and let $\mathbb{M} = \mathbb{M}_{\mathcal{F}}$ be the Voronoi partition w.r.t. $\mathcal{F}$. Then for every $p \in \mathcal{F}$, it holds that $\mathbf{loc}(p) \subseteq \mathbb{M}(p)$.*

Proof. For the sake of contradiction, suppose there are two distinct objects p_1 and p_2 such that some vertex $v \in \mathbf{loc}(p_1)$ belongs to $\mathbb{M}(p_2)$. Then in particular we have that

$$\mathrm{dist}(v, \mathbf{loc}(p_2)) - r(p_2) < \mathrm{dist}(v, \mathbf{loc}(p_1)) - r(p_1) = -r(p_1),$$

and hence

$$\mathrm{dist}(\mathbf{loc}(p_1), \mathbf{loc}(p_2)) \leq \mathrm{dist}(v, \mathbf{loc}(p_2)) < r(p_2) - r(p_1).$$

This is a contradiction with the definition of normality. □

The following simple claim shows that Voronoi regions behave as expected.

Lemma 4.2. *Suppose $\mathcal{F} \subseteq \mathcal{D}$ is a normal subfamily of objects and let $\mathbb{M} = \mathbb{M}_{\mathcal{F}}$ be the Voronoi partition w.r.t. $\mathcal{F}$. Then for every $p \in \mathcal{F}$ and every $v \in \mathbb{M}(p)$, the unique shortest path from v to $\mathbf{loc}(p)$ is entirely contained in $\mathbb{M}(p)$. In particular, $G[\mathbb{M}(p)]$ is connected.*

Proof. Let P be the shortest path connecting v with some $w \in \mathbf{loc}(p)$. Note that the uniqueness of P follows from (♣). For the sake of contradiction, suppose there exists $v' \in V(P)$ such that $\mathrm{dist}(v', \mathbf{loc}(p')) - r(p') < \mathrm{dist}(v', \mathbf{loc}(p)) - r(p)$, for some $p' \in \mathcal{F}$, $p' \neq p$. Since P is the shortest path, we have:

$$\begin{aligned}
\mathrm{dist}(v, \mathbf{loc}(p)) - r(p) &= \mathrm{dist}(v, w) - r(p) \\
&= \mathrm{dist}(v, v') + \mathrm{dist}(v', w) - r(p) \\
&= \mathrm{dist}(v, v') + \mathrm{dist}(v', \mathbf{loc}(p)) - r(p) \\
&> \mathrm{dist}(v, v') + \mathrm{dist}(v', \mathbf{loc}(p')) - r(p') \geq \mathrm{dist}(v, \mathbf{loc}(p')) - r(p').
\end{aligned}$$

This is a contradiction with $v \in \mathbb{M}(p)$. The second part of the claim follows from the first part, Lemma 4.1, and the fact that $G[\mathbf{loc}(p)]$ is connected. □

Lemma 4.2 gives rise to the following definition of a spanning tree of a Voronoi region. For a normal subfamily of objects $\mathcal{F} \subseteq \mathcal{D}$ and some $p \in \mathcal{F}$, we shall distinguish a spanning tree $\widehat{T}_{\mathcal{F}}(p)$ of $G[\mathbb{M}(p)]$ constructed as follows:

- start with $\widehat{T}_{\mathcal{F}}(p) = T(p)$;
- for every $v \in \mathbb{M}(p)$, add to $\widehat{T}_{\mathcal{F}}(p)$ the unique shortest path from v to $\mathbf{loc}(p)$.

By Lemma 4.2 and properties of shortest paths, $\widehat{T}_{\mathcal{F}}(p)$ defined in this manner is a spanning tree of $G[\mathbb{M}(p)]$. Note that we may view $\widehat{T}_{\mathcal{F}}(p)$ as rooted in $\mathbf{cen}(p)$, the center of p. However, $\widehat{T}_{\mathcal{F}}(p)$ is *not* necessarily the tree of shortest paths from $\mathbf{cen}(p)$ in $G[\mathbb{M}(p)]$; for example, it is possible that the shortest path between two vertices of $\mathbf{loc}(p)$ is not contained fully in $\mathbf{loc}(p)$.

The following statement is straightforward, but it is instructive to state it explicitly.

Lemma 4.3. *Let $\mathcal{F}' \subseteq \mathcal{F} \subseteq \mathcal{D}$ be two normal families of objects. Then for every $p \in \mathcal{F}'$ it holds that $\mathbb{M}_{\mathcal{F}'}(p) \supseteq \mathbb{M}_{\mathcal{F}}(p)$ and $\widehat{T}_{\mathcal{F}'}(p)$ is a supergraph of $\widehat{T}_{\mathcal{F}}(p)$.*

In other words, when we restrict the family of objects, then the Voronoi regions and their spanning trees can only grow.

4.4 Voronoi (pre-)diagrams

Let us fix some normal subfamily of objects $\mathcal{F}$, let $\ell = |\mathcal{F}|$, and let $\mathbb{M} = \mathbb{M}_{\mathcal{F}}$ be the corresponding Voronoi partition. Intuitively, $\mathbb{M}$ partitions G into regions, such that if we put an edge between adjacent regions, then the resulting graph is planar. We now formalize this concept by introducing *Voronoi prediagrams* and *Voronoi diagrams*. More precisely, Voronoi (pre-)diagrams correspond to the dual of the adjacency graph between the regions.

A *Voronoi prediagram* for $\mathcal{F}$ is a subgraph $\mathcal{H}_{\mathcal{F}}$ of G^*, the dual of G, defined as follows:

1. Start with the graph G^*.
2. For every $p \in \mathcal{F}$, remove all the dual edges corresponding to edges of $\widehat{T}(p)$.
3. Iteratively remove vertices of degree 1 in the obtained graph, up to the point when the minimum degree of the graph is 2.

In this definition we drop the weights of edges, i.e., $\mathcal{H}$ is an unweighted graph. See Figure 6 for an example. The following lemma formalizes the properties of the Voronoi prediagram.

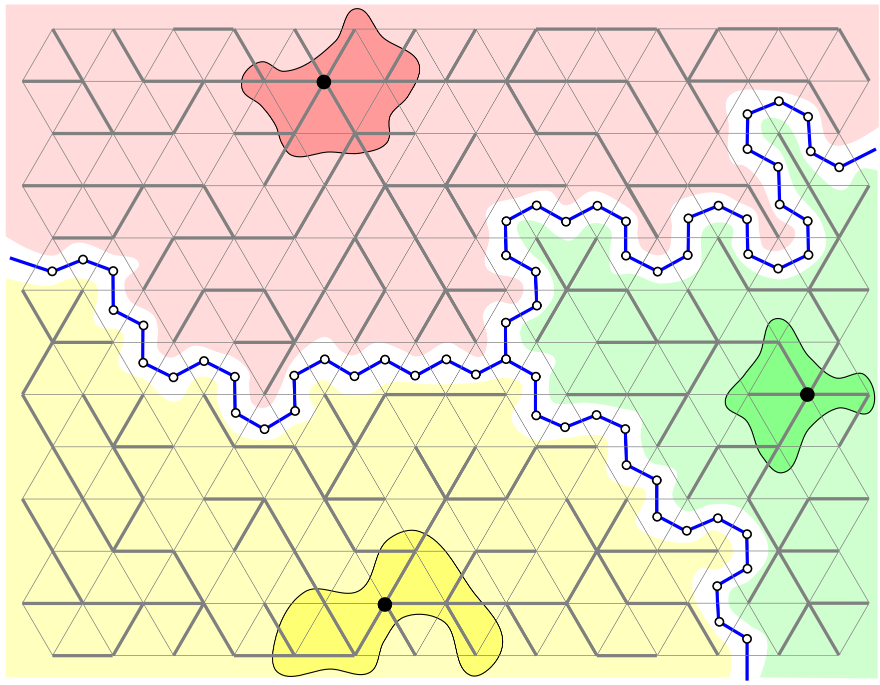


Figure 6: Construction of the prediagram in an area between three objects: red, yellow, and green. The Voronoi regions are depicted in light red, light yellow, and light green. Gray edges belong to the primal graph G, and thick gray edges belong to spanning trees $\widehat{T}_{\mathcal{F}}(p)$. The white vertices and thick blue edges belong to the prediagram $\mathcal{H}_{\mathcal{F}}$.

Lemma 4.4. *The prediagram $\mathcal{H}$ is a simple, connected graph embedded on the sphere, where every vertex is of degree 2 or 3. Moreover, $\mathcal{H}$ has exactly ℓ faces, and faces of $\mathcal{H}$ correspond to objects of $\mathcal{F}$ in the following sense: object p is associated with a face $f_p^* \in F(\mathcal{H})$ such that f_p^* contains all the vertices of $\mathbb{M}(p)$, and no other vertex.*

Proof. Observe that $\mathcal{H}$ is a subgraph of G^*, so it inherits a sphere embedding from G^*. The fact that $\mathcal{H}$ is simple follows from the fact that G^* is simple, since G is triangulated, simple, and has more than 3 vertices. Also, the fact that G is triangulated implies that G^* is a 3-regular graph. Hence $\mathcal{H}$, as a subgraph of G^*, has maximum degree at most 3. The fact that G^* has minimum degree at least 2 follows from the construction.

We now show that $\mathcal{H}$ is connected. Obviously G^* is connected, as a dual of a sphere-embedded graph. We now argue that during the step when the dual edges to the edges of some $\widehat{T}(p)$ are removed, we also cannot spoil the connectivity of $\mathcal{H}$. Let W be a walk in G^* that traverses the set of faces incident to vertices and edges of $\widehat{T}(p)$, but does not use any dual edge to an edge of $\widehat{T}(p)$; W is constructed by traversing $\widehat{T}(p)$ around, with respect to its planar embedding of G. All the edges used W are not removed when $\widehat{T}(p)$ is removed, and could not have been removed when some previous $\widehat{T}(p')$ was removed. Hence, walk W remains in the graph after $\widehat{T}(p)$ is removed. Since the removed dual edges of $\widehat{T}(p)$ always connect two faces visited by W, we infer that all the vertices of G^* that could have been possibly disconnected by removal of $\widehat{T}(p)$, are actually still connected by W. Therefore, after the second step of the construction of $\mathcal{H}$ the graph is still connected. Since removal of a degree-1 vertex does not spoil connectivity, we infer that at the end $\mathcal{H}$ is indeed connected.

We are left with proving the correspondence between objects of $\mathcal{F}$ and faces of $\mathcal{H}$. Since $\mathcal{H}$ is a subgraph of G^*, every face of $\mathcal{H}$ consists of a union of a nonempty set of original faces of G^*, which moreover were connected in G^*. Every face of G^* corresponds to vertex of G, which means that every face f^* of $\mathcal{H}$ contains a nonempty subset of vertices $X(f^*)$, which moreover is connected in G. Note now that for every $p \in \mathcal{F}$, removal of the dual edges of $\widehat{T}(p)$ merges all the faces of G^* corresponding to vertices of $\mathbb{M}(p)$, which means that vertices of every Voronoi region $\mathbb{M}(p)$ are contained in one face $f_p^* \in F(\mathcal{H})$. It remains to show that no face $f^* \in F(\mathcal{H})$ can accommodate more than one Voronoi region. If this was the case, then, since $X(f^*)$ is connected in G, there would be an edge $uv \in E(G)$, $u, v \in X(f^*)$, that would connect two vertices from different Voronoi regions, and whose dual was removed in the construction of $\mathcal{H}$. However, in the construction of $\mathcal{H}$, in the first step we removed only duals to the edges that connect two vertices of the same Voronoi region, and in the second step we removed only vertices of degree 1, which cannot merge two faces of the graph. □

Lemma 4.4 shows that the prediagram $\mathcal{H}$ is a suitable skeleton of the final diagram. More precisely, $\mathcal{H}$ consists of vertices of degree 3 that are connected using long paths consisting of vertices of degree 2. From now on we assume that $\ell > 2$. If $\ell = 1$ then $\mathcal{H}$ is empty, and if $\ell = 2$ then $\mathcal{H}$ is a cycle. Both these cases are degenerate in the argumentation to follow.

Given the prediagram $\mathcal{H} = \mathcal{H}_\mathcal{F}$, we define the *Voronoi diagram* $\widetilde{\mathcal{H}} = \widetilde{\mathcal{H}}_\mathcal{F}$ as follows. To obtain $\widetilde{\mathcal{H}}$ from $\mathcal{H}$, we iteratively take a vertex of degree 2 in the graph, remove it, and add a new edge between its neighbors. Thus, all the paths consisting of vertices of degree 2 in $\mathcal{H}$ are shortened to single edges. Note, that in this manner $\widetilde{\mathcal{H}}$ may become a multigraph:

- There may appear multiple edges between a pair of vertices u and v, in case there was more than one path of vertices of degree 2 connecting u and v in $\mathcal{H}$.
- There may appear a loop at some vertex u, in case there was a path of vertices of degree 2 traveling from u back to itself.

In the following, we assume that a loop at vertex u contributes with 2 to the degree of u. The following lemma formalizes the properties of the Voronoi diagram $\widetilde{\mathcal{H}}$.

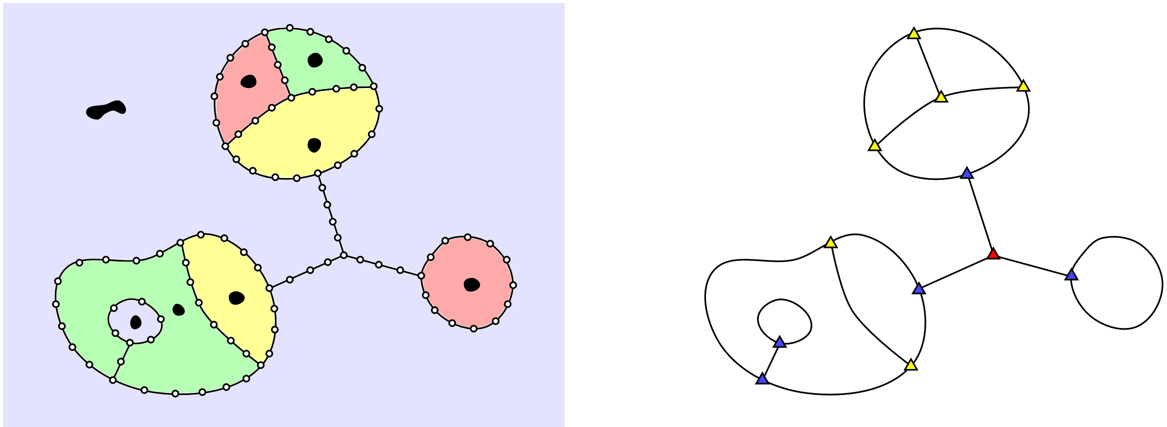
Lemma 4.5. *If $\ell > 2$, then the diagram $\widetilde{\mathcal{H}}$ is a connected 3-regular multigraph embedded on a sphere. Moreover, $\widetilde{\mathcal{H}}$ has exactly $2\ell - 4$ vertices, $3\ell - 6$ edges, and ℓ faces, and the set of faces of $\widetilde{\mathcal{H}}$ is exactly the same as the set of faces of $\mathcal{H}$.*

Proof. The diagram $\widetilde{\mathcal{H}}$ is constructed from $\mathcal{H}$ by taking every path P in $\mathcal{H}$ that begins and ends in vertices u, v of degree 3 but travels only through vertices of degree 2, and replacing it with a single edge uv (possibly $u = v$). The embedding of $\widetilde{\mathcal{H}}$ into the sphere is defined as follows: The vertices of $V(\widetilde{\mathcal{H}})$ are embedded exactly in the same manner as in the embedding of $\mathcal{H}$. Whenever a path P is replaced with an edge uv between the endpoints of P, then the embedding of uv is defined as the union of the embeddings of the edges of P. Thus, the faces of $\widetilde{\mathcal{H}}$ are exactly the same as faces of $\mathcal{H}$.

When constructing $\widetilde{\mathcal{H}}$, we removed all the vertices of degree 2, while all the vertices of degree 3 keep their degrees. Hence $\widetilde{\mathcal{H}}$ is 3-regular. Moreover, since $\mathcal{H}$ was connected (Lemma 4.4), then by the construction so is $\widetilde{\mathcal{H}}$.

The fact that $|F(\widetilde{\mathcal{H}})| = \ell$ follows from Lemma 4.4 and the fact that the faces of $\widetilde{\mathcal{H}}$ are exactly the same as faces of $\mathcal{H}$. The conclusion that $|V(\widetilde{\mathcal{H}})| = 2\ell - 4$ and $|E(\widetilde{\mathcal{H}})| = 3\ell - 6$ follows from the fact that $\widetilde{\mathcal{H}}$ is connected and 3-regular, and a simple application of the Euler's formula, which is valid also for connected multigraphs. □

In the following, the vertices of $V(\widetilde{\mathcal{H}})$ will be also call the *branching points* of diagram $\widetilde{\mathcal{H}}$. The following auxiliary lemma shows that loops of the diagram $\widetilde{\mathcal{H}}$ have very simple structure.



(a) Voronoi partition $\mathbb{M}$ and the prediagram $\mathcal{H}$ (b) Diagram $\widetilde{\mathcal{H}}$

Figure 7: Example of a Voronoi partition $\mathbb{M}_{\mathcal{F}}$ and the corresponding prediagram and diagram. Objects of $\mathcal{F}$ are denoted in black. Note that vertices in the prediagram $\mathcal{H}_{\mathcal{F}}$ are faces of the original graph G. Branching points of the diagram $\widetilde{\mathcal{H}}_{\mathcal{F}}$ are depicted in yellow, blue, and red, depending whether they correspond to type-1, -2, or -3 singular face according to definitions in Section 4.5.

Lemma 4.6. *Suppose H is a connected 3-regular multigraph embedded on a sphere. Then for every loop in H, one of the disks in which the loop separates the sphere is a face of H.*

Proof. Consider a loop e at vertex u in H. Since H is 3-regular and loop e contributes with 2 to the degree of u, then u is incident to one more edge e'. Then e' is embedded into one of the disks into which e partitions the sphere, and by the connectivity of H we infer that the whole rest of H is embedded into the same disk. The second disk contains therefore no point of the embedding of H, and hence is a face of H. □

4.5 Important faces

In the algorithm we consider a large family of long sequences of faces of G, in hope of capturing a sequence that forms a short balanced separator of the Voronoi diagram imposed by the solution, on which the recursion step can be employed. Since balanced separators of this diagram can have width as large as $\mathcal{O}(\sqrt{k})$, we need to take into consideration all the sequences of at most this length. Of course, we could consider every face of G as a possible candidate to be used in a separator, but then we would have $n^{\mathcal{O}(\sqrt{k})}$ candidate separators, which is too much for the running time promised in Theorem 3.1.

In this section we prove the following: one can enumerate in polynomial time a family of at most d^4 faces of G, called further *important faces*, such that the candidates for faces used in separators can be safely picked from this family. Thus, we arrive at $d^{\mathcal{O}(\sqrt{k})}$ candidates for separators, instead of the original $n^{\mathcal{O}(\sqrt{k})}$. Formally, this section is devoted to the proof of the following result.

Theorem 4.7. *There exists a family $\mathcal{E} \subseteq F(G)$ such that*

- $|\mathcal{E}| \leq d^4$*;*
- *for every normal subfamily of objects $\mathcal{F} \subseteq \mathcal{D}$, it holds that $V(\widetilde{\mathcal{H}}_{\mathcal{F}}) \subseteq \mathcal{E}$.*

Moreover, family $\mathcal{E}$ can be constructed in time $\mathcal{O}(d^4 \cdot n^{\mathcal{O}(1)})$.

Let us remark that Theorem 4.7 is needed *only* for the purpose of reducing the running time — if we just allowed all the faces of G to be used to build separators, then we would obtain an algorithm solving Disjoint Network Coverage in time $d^{\mathcal{O}(\sqrt{k})} \cdot c^{\mathcal{O}(1)} \cdot n^{\mathcal{O}(\sqrt{k})}$. In many of our applications in Section 3.2, it actually holds that $n = d^{\mathcal{O}(1)}$, and hence we would obtain the same asymptotic running time without the speed-up due to the results of this section. Therefore, the result on enumerating a small family of important faces can be considered as optional, and the reader interested only in a $d^{\mathcal{O}(\sqrt{k})} \cdot c^{\mathcal{O}(1)} \cdot n^{\mathcal{O}(\sqrt{k})}$ time algorithm is advised to skip this section.

We now move on to the proof of Theorem 4.7, which spans the whole this section. However, before we proceed, we need some definitions and auxiliary lemmas. Let us fix some normal family of objects $\mathcal{F} \subseteq \mathcal{D}$, and let $\mathbb{M} = \mathbb{M}_{\mathcal{F}}$ be the corresponding Voronoi partition. For two distinct objects $p_1, p_2 \in \mathcal{F}$, we say that Voronoi regions $\mathbb{M}(p_1)$ and $\mathbb{M}(p_2)$ *meet* at face f if ∂f contains both a vertex of $\mathbb{M}(p_1)$ and a vertex of $\mathbb{M}(p_2)$.

We say that face f is a *type*-1 *singular face* for $\mathcal{F}$, if for a triple of distinct objects $p_1, p_2, p_3 \in \mathcal{F}$, the Voronoi regions $\mathbb{M}(p_1), \mathbb{M}(p_2), \mathbb{M}(p_3)$ pairwise meet at f. Then we say that triple (p_1, p_2, p_3) *certifies* that f is a type-1 singular face. The following lemma shows that there are generally not so many type-1 singular faces.

Lemma 4.8. *Suppose* $\{p_1, p_2, p_3\} = \mathcal{F} \subseteq \mathcal{D}$ *is a normal family of three objects. Then there are at most* 2 *faces of* G *that are type*-1 *singular faces for* $\mathcal{F}$*.*

Proof. Let $M_i = \mathbb{M}_{\mathcal{F}}(p_i)$ for $i = 1, 2, 3$. For the sake of contradiction, suppose that there are 3 different faces f_1, f_2, f_3, such that M_1, M_2, M_3 all pairwise meet at each of f_1, f_2, f_3. Construct a planar graph H as follows:

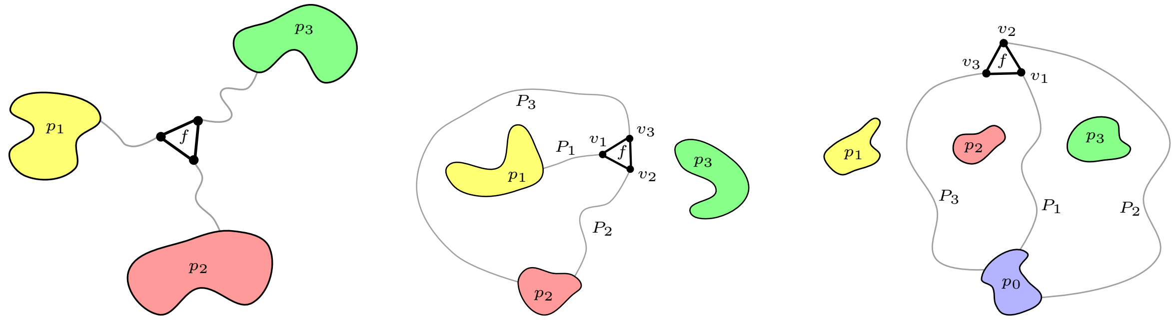
- For each $i = 1, 2, 3$, introduce a vertex w_i inside face f_i, and connect w_i to all the vertices of ∂f_i.
- For each $j = 1, 2, 3$, contract the whole subgraph $G[M_j]$ to a single vertex m_j. Note that this is possible since $G[M_j]$ is connected by Lemma 4.2.
- Remove all the vertices and edges of the graph apart from $\{w_i : i = 1, 2, 3\}$, $\{m_j : j = 1, 2, 3\}$, and the edges between them.

Observe that the obtained graph H is isomorphic to $K_{3,3}$. After performing the first step of the construction we still had a graph with a sphere embedding, since the new vertices and edges could be drawn inside faces f_i. Then, we applied only edge contractions, vertex deletions and edge deletions. Hence, we have found a $K_{3,3}$ as a minor of a planar graph, a contradiction. □

Now we define the second type of singular faces. We say that face f is a *type*-2 *singular face* for $\mathcal{F}$, if there exists a triple of distinct objects $p_1, p_2, p_3 \in \mathcal{F}$ satisfying the following. One vertex v_1 of ∂f belongs to $\mathbb{M}(p_1)$ and two vertices v_2, v_3 of ∂f belong to $\mathbb{M}(p_2)$. Moreover, if we take a closed walk W in G constructed by concatenating paths in $\widehat{T}(p_2)$ from $\mathbf{cen}(p_2)$ to v_2 and v_3, and the edge $v_2 v_3$, then this closed walk separates $\mathbf{loc}(p_1)$ from $\mathbf{loc}(p_3)$ on the sphere. Observe that this walk is not necessarily a path, but, as it is formed by two paths in the tree $\widehat{T}(p_2)$ closed by an edge, its removal partitions the sphere into two disks. The requirement is exactly that one of these disks contains $\mathbf{loc}(p_1)$ and the second contains $\mathbf{loc}(p_3)$.

Similarly as before, we will say that triple (p_1, p_2, p_3) *certifies* that f is a type-2 singular face. The following lemma shows that, again, there are not so many type-2 singular faces.

Lemma 4.9. *Suppose* $\{p_1, p_2, p_3\} = \mathcal{F} \subseteq \mathcal{D}$ *is a normal family of three objects. Then there is at most* 1 *face of* G *that is a type*-2 *singular face for* $\mathcal{F}$ *and is certified by the triple* (p_1, p_2, p_3)*.*



(a) Type 1 (b) Type 2 (c) Type 3

Figure 8: Singular faces of types 1, 2, and 3. Paths from a vertex to the nearest object have been depicted in gray.

Proof. Suppose there are two such faces f and f'. We adopt the notation from the definition of a type-2 singular face for f: $\{v_1\} = V(\partial f) \cap \mathbb{M}(p_1)$, $\{v_2, v_3\} = V(\partial f) \cap \mathbb{M}(p_2)$, P_t is the $\mathbf{cen}(p_2)$-v_t path inside $\widehat{T}(p_2)$, for $t = 2, 3$, and W is the closed walk formed by concatenation of P_2, P_3, and edge v_2v_3. Note that edge v_2v_3 does not belong to $\widehat{T}(p_2)$, since then P_2 would be a subpath of P_3 or vice versa, and W would degenerate to a path that would not separate any two nonempty disks. Let D_1 be the open disk of $\Sigma \setminus W$ that contains p_1, and D_3 be the one that contains p_3. Since $v_1 \in \mathbb{M}(p_1)$, $\mathbf{loc}(p_1)$ is contained in D_1, and $G[\mathbb{M}(p_1)]$ is connected, we infer that $v_1 \in D_1$. Moreover, if P_1 is the $\mathbf{cen}(p_1)$-v_1 path inside $\widehat{T}(p_1)$, then P_1 is entirely contained in D_1.

Let v'_t for $t = 1, 2, 3$, P'_t for $t = 2, 3$ and W' be the analogical objects defined for f'. Since f' contains a vertex of $\mathbb{M}(p_1)$, then we have that f' is contained in D_1. Since P'_2 and P'_3 are paths inside the tree $\widehat{T}(p_2)$, and W is defined also by paths inside the tree $\widehat{T}(p_2)$, we infer that both P'_2 and P'_3 are entirely contained in $D_1 \cup W$. Moreover, they do not contain edge v_2v_3 (since this edge does not belong to $\widehat{T}(p_2)$). As $f' \neq f$ and the other face incident to v_2v_3 lies outside D_1, we have also that $v_2v_3 \neq v'_2v'_3$. Consequently, we infer that the closed walk W' does not use the edge v_2v_3.

To obtain a contradiction, we now exhibit a curve γ on the sphere that connects $\mathbf{cen}(p_1)$ with $\mathbf{cen}(p_3)$ and does not touch W'; this will contradict the fact that W' separates $\mathbf{loc}(p_1)$ from $\mathbf{loc}(p_3)$. Construct γ by:

(i) starting by traveling from $\mathbf{cen}(p_1)$ to v_1 using path P_1 inside $\widehat{T}(p_1)$;
(ii) traveling inside the face f from v_1 to the middle of edge v_2v_3; and
(iii) finishing by traveling inside the disk D_3 from the middle of edge v_2v_3 to $\mathbf{cen}(p_3)$.

In part (i) of γ we do not touch W', because we only travel through vertices of $\mathbb{M}(p_1)$, whereas all the vertices on W' belong to $\mathbb{M}(p_2)$. Then, in part (ii) we do not touch W', since W' does not use the edge v_2v_3. Finally, in part (iii) we do not touch W' since W' is disjoint with D_3. Hence, γ is a curve from $\mathbf{cen}(p_1)$ to $\mathbf{cen}(p_3)$ that avoids W', a contradiction. $\square$

Finally, we define the third type of singular faces. We say that face f is a *type-*3 *singular face* for $\mathcal{F}$, if there exists a quadruple of distinct objects $p_0, p_1, p_2, p_3 \in \mathcal{F}$ satisfying the following. Let $\{v_1, v_2, v_3\} = V(\partial f)$. The first requirement is that $v_1, v_2, v_3 \in \mathbb{M}(p_0)$, but no edge of ∂f is contained in $\widehat{T}(p_0)$. Let P_t be the path inside $\widehat{T}(p_0)$ from $\mathbf{cen}(p_0)$ to v_t, for $t = 1, 2, 3$. Let W_t be the closed walk formed by concatenation of paths P_{t+1}, P_{t+2} and the edge $v_{t+1}v_{t+2}$, where the indices behave cyclically. Observe that removal of walks W_1, W_2, W_3 partitions sphere Σ into four open disks: disks D_t for $t = 1, 2, 3$ such that W_t contains the boundary of D_t, and the last disk being simply the face f. Then the second requirement is that $\mathbf{loc}(p_t)$ is entirely contained in D_t, for $t = 1, 2, 3$.

Similarly as before, we will say that a quadruple (p_0, p_1, p_2, p_3) *certifies* that f is a type-3 singular face. The following lemma shows that, again, there are not so many type-3 singular faces.

Lemma 4.10. *Suppose $\{p_0, p_1, p_2, p_3\} = \mathcal{F} \subseteq \mathcal{D}$ is a normal family of four objects. Then there is at most* 1 *face of G that is a type-*3 *singular face for $\mathcal{F}$ and is certified by the quadruple (p_0, p_1, p_2, p_3).*

Proof. Suppose there are two such faces f and f'. We adopt the notation from the definition of a type-3 singular face for f, and we will follow the same notation but with primes for the face f'. By assumption, no edge of f or f' belongs to $\widehat{T}(p_0)$.

Recall that removal of walks W_1, W_2, W_3 partitions Σ into four open disks: D_1, D_2, D_3 and f. Since $f' \neq f$, f' is contained in one of the disks D_1, D_2, D_3. Without loss of generality suppose f' is contained in D_1. Removal of W_1 from Σ partitions Σ into two disks: one of them is simply D_1, and the second is a disk $\widetilde{D_1}$ that contains D_2, D_3 and f. Observe that since W_1 is formed by edge v_2v_3 and two paths P_2, P_3, which are contained in the tree $\widehat{T}(p_0)$, then each of the paths P_1', P_2', P_3' has to be entirely contained either in $D_1 \cup W_1$ or in $\widetilde{D_1} \cup W_1$ — this is because paths P_1', P_2', P_3' are also contained in the tree $\widehat{T}(p_0)$. Since $\partial f' \subseteq D_1 \cup W_1$, we infer that all these paths are contained in $D_1 \cup W_1$. Since f' is also contained in D_1, we conclude that all the closed walks W_1', W_2', W_3' are contained in $D_1 \cup W_1$.

Now examine disk $\widetilde{D_1}$. Notice that this disk is disjoint with all the walks W_1', W_2', W_3' and does not contain f', which means that it is entirely contained in one of the disk D_1', D_2', or D_3'. However, $\widetilde{D_1}$ contains both $\mathbf{loc}(p_2)$ and $\mathbf{loc}(p_3)$, while every disk D_1', D_2', D_3' contains only one of them. This is a contradiction. $\square$

In Lemmas 4.8, 4.9, and 4.10 we focused only on families $\mathcal{F}$ of size 3, 3, and 4, respectively. The following lemma, which follows immediately from Lemma 4.3 and the definition of singular faces of types 1, 2, and 3, justifies why we can do it.

Lemma 4.11. *Suppose $\mathcal{F} \subseteq \mathcal{D}$ is a normal family of objects. Then for a face f of G, the following conditions hold:*

- *If f is a type-*1 *singular face for $\mathcal{F}$, certified by a triple (p_1, p_2, p_3), then it is a type-*1 *singular face for $\{p_1, p_2, p_3\}$, certified by the same triple (p_1, p_2, p_3).*
- *If f is a type-*2 *singular face for $\mathcal{F}$, certified by a triple (p_1, p_2, p_3), then it is a type-*2 *singular face for $\{p_1, p_2, p_3\}$, certified by the same triple (p_1, p_2, p_3).*
- *If f is a type-*3 *singular face for $\mathcal{F}$, certified by a quadruple (p_0, p_1, p_2, p_3), then it is a type-*3 *singular face for $\{p_0, p_1, p_2, p_3\}$, certified by the same quadruple (p_0, p_1, p_2, p_3).*

Now we show that for a normal family $\mathcal{F}$, every branching point of $\widetilde{\mathcal{H}}_{\mathcal{F}}$ is always a singular face, and moreover its type depends on the number of bridges it is incident to in $\widetilde{\mathcal{H}}_{\mathcal{F}}$. Note that since $\widetilde{\mathcal{H}}_{\mathcal{F}}$ is 3-regular, every its vertex is incident to 0, 1, or 3 bridges of $\widetilde{\mathcal{H}}_{\mathcal{F}}$.

Lemma 4.12. *Suppose $\mathcal{F} \subseteq \mathcal{D}$ is a normal subfamily of objects, and let $\widetilde{\mathcal{H}} = \widetilde{\mathcal{H}}_{\mathcal{F}}$ be the corresponding Voronoi diagram. Then for every branching point $f \in V(\widetilde{\mathcal{H}})$, the following holds:*

*(1) If f is not incident to any bridge in $\widetilde{\mathcal{H}}$, then f is a type-*1 *singular face for $\mathcal{F}$.*
*(2) If f is incident to exactly one bridge in $\widetilde{\mathcal{H}}$, then f is a type-*2 *singular face for $\mathcal{F}$.*
*(3) If f is incident to exactly three bridges in $\widetilde{\mathcal{H}}$, then f is a type-*3 *singular face for $\mathcal{F}$.*

In particular, every branching point of $\widetilde{\mathcal{H}}$ is a singular face for $\mathcal{F}$ of one of the three types.

Proof. Let $\tilde{e}_1^*, \tilde{e}_2^*, \tilde{e}_3^*$ be the three edges incident to f in $\widetilde{\mathcal{H}}$ (possibly two of these edges are equal, in case f is incident to a loop in $V(\widetilde{\mathcal{H}})$). Then, in the prediagram $\mathcal{H}$, f was incident to three edges e_1^*, e_2^*, e_3^*, which are the first edges of the paths contracted to $\tilde{e}_1^*, \tilde{e}_2^*, \tilde{e}_3^*$, respectively (these three edges are pairwise different, since $\mathcal{H}$ is simple). Edges e_1^*, e_2^*, e_3^* are dual to edges e_1, e_2, e_3, respectively, which form the boundary of the face f.

We perform a case study depending on how many of edges $\tilde{e}_1^*, \tilde{e}_2^*, \tilde{e}_3^*$ are bridges in $\widetilde{\mathcal{H}}$ (note that a loop is not considered a bridge). Observe that, for $t = 1, 2, 3$, edge $\tilde{e}_t^*$ is a bridge in $\widetilde{\mathcal{H}}$ if and only if e_t^* is a bridge in $\mathcal{H}$.

Suppose first that none of edges $\tilde{e}_1^*, \tilde{e}_2^*, \tilde{e}_3^*$ are bridges. Then f is incident to three different faces of $\widetilde{\mathcal{H}}$. Since faces of $\widetilde{\mathcal{H}}$ correspond one-to-one to objects of $\mathcal{F}$ (see Lemmas 4.4 and Lemmas 4.5), there are three distinct objects $p_1, p_2, p_3 \in \mathcal{F}$ such that f is incident to corresponding faces $f_{p_1}^*, f_{p_2}^*, f_{p_3}^*$ of $\widetilde{\mathcal{H}}$. This means that one of the vertices of f belongs to $\mathbb{M}(p_1)$, second to $\mathbb{M}(p_2)$, and third to $\mathbb{M}(p_3)$, and the triple (p_1, p_2, p_3) certifies that f is a type-1 singular face for $\mathcal{F}$. This proves (1).

Suppose second that $\tilde{e}_1^*$ is a bridge in $\widetilde{\mathcal{H}}$, while edges $\tilde{e}_2^*, \tilde{e}_3^*$ are not bridges. This means that there exists two distinct objects $p_1, p_2 \in \mathcal{F}$, with corresponding faces of $\widetilde{\mathcal{H}}$ being $f_{p_1}^*$ and $f_{p_2}^*$, such that the face incident to f between $\tilde{e}_2^*, \tilde{e}_3^*$ is $f_{p_1}^*$, while the face $f_{p_2}^*$ resides on both sides of the edge $\tilde{e}_1^*$. Let H_1 be the subgraph of $\widetilde{\mathcal{H}}$ induced by all the branching points that become disconnected from f by removing the bridge $\tilde{e}_1^*$. Since $\widetilde{\mathcal{H}}$ is 3-regular, H_1 is not a tree, so it has at least two faces. Only one face of H_1 contains face $f_{p_2}^*$, and all the other faces of H_1 are actually faces of $\widetilde{\mathcal{H}}$. Let then p_3 be any object of $\mathcal{F}$ whose corresponding face $f_{p_3}^*$ of $\widetilde{\mathcal{H}}$ is a face of H_1. Similarly as in the previous case, one vertex $v_1 \in V(\partial f)$ belongs to $\mathbb{M}(p_1)$ and two vertices of $v_2, v_3 \in V(\partial f)$ belong to $\mathbb{M}(p_2)$. Moreover these two vertices v_2, v_3 are connected by the primal edge e_1. Consider now a closed walk W obtained by concatenating: walk P_2 inside $\widehat{T}(p_2)$ from $\mathbf{cen}(p_2)$ to v_2, walk P_3 inside $\widehat{T}(p_2)$ from $\mathbf{cen}(p_2)$ to v_3, and edge $e_1 = v_2v_3$. Removal of walk W from the sphere separates the sphere into two open disks, one of which contains all the branching points of $V(H_1)$ — and in particular also $\mathbf{loc}(p_3)$ — and the second contains all other the branching points of $\widetilde{\mathcal{H}}$ — and in particular also $\mathbf{loc}(p_1)$. Hence, W separates $\mathbf{loc}(p_1)$ from $\mathbf{loc}(p_3)$. This means that triple (p_1, p_2, p_3) certifies that f is a type-2 singular face for $\mathcal{F}$. This proves (2).

Finally, suppose that all the edges $\tilde{e}_1^*, \tilde{e}_2^*, \tilde{e}_3^*$ are bridges in $\widetilde{\mathcal{H}}$. This means that there is one object $p_0 \in \mathcal{F}$ and corresponding face $f_{p_0}^*$ of $\widetilde{\mathcal{H}}$, such that $f_{p_0}^*$ is on both sides of each of the edges $\tilde{e}_1^*, \tilde{e}_2^*, \tilde{e}_3^*$. Let $\{v_1, v_2, v_3\} = V(\partial f)$, where $e_t = v_{t+1}v_{t+2}$ for $t = 1, 2, 3$. Hence, in particular we have that $v_1, v_2, v_3 \in \mathbb{M}(p_0)$. Similarly as in the previous case, let H_t be the subgraph of $\widetilde{\mathcal{H}}$ induced by all the branching points that become disconnected from f by removing the bridge $\tilde{e}_t^*$, for $t = 1, 2, 3$. Each H_t is not a forest, since $\widetilde{\mathcal{H}}$ is 3-regular. Moreover, each H_t has one face containing the original face $f_{p_0}^*$, whereas all the other faces of $\widetilde{\mathcal{H}}$ are partitioned into face sets $F(H_1)$, $F(H_2)$, $F(H_3)$. Hence, there exist three distinct objects $p_1, p_2, p_3 \in \mathcal{F}$, distinct from p_0, such that the corresponding faces $f_{p_1}^*$, $f_{p_2}^*$, $f_{p_3}^*$ of $\widetilde{\mathcal{H}}$ are actually faces of H_1, H_2, and H_3, respectively. We now define P_t to be the path inside $\widehat{T}(p_0)$ from $\mathbf{cen}(p_0)$ to v_t, for $t = 1, 2, 3$, and W_t to be the closed walk formed by concatenation of paths P_{t+1}, P_{t+2} and the edge $v_{t+1}v_{t+2}$. Similarly as in the previous point, we see that each walk W_t separates $\mathbf{loc}(p_t)$ from $\mathbf{loc}(p_{t+1})$ and $\mathbf{loc}(p_{t+2})$. This means that quadruple (p_0, p_1, p_2, p_3) certifies that f is a type-3 singular face for $\mathcal{F}$. This proves (3), and concludes the proof. $\square$

We are finally ready to prove Theorem 4.7, that is, present an enumeration algorithm for important faces. Family $\mathcal{E}$ will consist of a union of three families $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$. Families $\mathcal{E}_1$ and $\mathcal{E}_2$ are constructed by considering every triple objects $p_1, p_2, p_3 \in \mathcal{F}$ s.t. $\{p_1, p_2, p_3\}$ is normal, and including all the faces that are type-1, resp. type-2, singular faces for $\{p_1, p_2, p_3\}$, certified by

(p_1, p_2, p_3). Lemmas 4.8 and 4.9 guarantees that for every triple we have at most two such faces for type 1 and at most one such face for type 2, so in total we have that $|\mathcal{E}_1| \leq 2d(d-1)(d-2)$ and $|\mathcal{E}_2| \leq d(d-1)(d-2)$. Similarly, family $\mathcal{E}_3$ is constructed by considering every quadruple of objects p_0, p_1, p_2, p_3 s.t. $\{p_0, p_1, p_2, p_3\}$ is normal, and including all the faces that are type-3 singular faces for $\{p_0, p_1, p_2, p_3\}$, certified by (p_0, p_1, p_2, p_3). Lemma 4.10 guarantees that for every quadruple we have at most one such face, and thus $|\mathcal{E}_3| \leq d(d-1)(d-2)(d-3)$. Note that, for a given triple or quadruple of objects, it can be verified in polynomial time using the definition whether a given face is a type-1/type-2/type-3 singular face for this triple/quadruple. Hence, families $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$ can be constructed in time $\mathcal{O}(d^4 \cdot n^{\mathcal{O}(1)})$.

We now set $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2 \cup \mathcal{E}_3$, so in particular we have the following:

$$|\mathcal{E}| \leq 3d(d-1)(d-2) + d(d-1)(d-2)(d-3) = d^2(d-1)(d-2) < d^4.$$

Let us consider any normal subfamily of objects $\mathcal{F} \subseteq \mathcal{D}$. By Lemma 4.11, we have that if $f \in F(G)$ is a type-t singular face for $\mathcal{F}$, then f has been included in respective family $\mathcal{E}_t$, for $t = 1, 2, 3$. On the other hand, Lemma 4.12 implies that every branching point of $\widetilde{\mathcal{H}}_{\mathcal{F}}$ is a singular face for $\mathcal{F}$ of one of the three types. We infer that every branching point of $\widetilde{\mathcal{H}}_{\mathcal{F}}$ belongs to $\mathcal{E}$. This concludes the proof of Theorem 4.7.

4.6 Curves, nooses, and sphere cut decompositions

Curves. Recall that for us a *curve* on a sphere is a homeomorphic image of an interval, and a *closed curve* is a homeomorphic image of a circle. A *directed (closed) curve* $\vec{\gamma}$ is simply a (closed) curve with a specified direction of traversal. The same curve but with the opposite direction of traversal will be denoted by $\vec{\gamma}^{-1}$. For a directed closed curve $\vec{\gamma}$, the *area enclosed* by $\vec{\gamma}$, denoted $\mathbf{enc}(\vec{\gamma})$, is the open disk of $\Sigma \setminus \vec{\gamma}$ that lies to the right of $\vec{\gamma}$ in the direction of its traversal (in this definition we fix the orientation of Σ). The second open disk, called the *area excluded* by $\vec{\gamma}$, will be denoted by $\mathbf{exc}(\vec{\gamma})$. Note that $\mathbf{enc}(\vec{\gamma}) = \mathbf{exc}(\vec{\gamma}^{-1})$.

Nooses and the radial graph. Let us consider a situation where some connected multigraph H is embedded on the sphere Σ. We say that a directed closed curve $\vec{\gamma}$ is a *noose*[3] if it intersects the embedding of G only in the vertices of H, and moreover visits every face of H at most once. The *length* of a noose $\vec{\gamma}$ is equal to $|V(H) \cap \vec{\gamma}|$, the number of vertices it traverses; we require that a noose traverses at least one vertex. We often say that a noose $\vec{\gamma}$ is *with respect to H* if the graph H is not clear from the context.

When considering nooses, we shall work on the *radial* graph of H, denoted $\mathbf{Rad}(H)$. The graph $\mathbf{Rad}(H)$ is a bipartite multigraph, where one partite set consists of $V(H)$, the vertices of H, and the second partite set is $F(H)$, the faces of H. For every face $f \in F(H)$ and vertex $v \in V(H)$ we put an edge between f and v for every occurrence of v on the boundary of f; note that if v appears multiple times on the boundary of f, it will be connected to f in $\mathbf{Rad}(H)$ via multiple edges. A sphere embedding of H naturally induces a sphere embedding of H as follows. For every face f we put the corresponding vertex of $\mathbf{Rad}(H)$ inside f; we call this vertex the *center* of f, and denote it by $\mathbf{cen}(f)$. Then we connect it to the vertices on the boundary of f radially, according to their order of traversal on ∂f. See Figure 9 for an example.

Now observe that every noose $\vec{\gamma}$ can be homeomorphically transformed to an equivalent noose (in terms of the order of vertices and faces visited) that is a simple cycle in the radial graph $\mathbf{Rad}(H)$

[3] In the classic literature nooses are not directed, but it will be convenient for us to work with directed ones. Also, often nooses are not required to visit every face at most once and nooses with this property are called *tight*. For us, all the considered nooses are tight.

with a chosen orientation. More precisely, whenever $\vec{\gamma}$ travels from a vertex u to a vertex v through a face f, then we replace this part of $\vec{\gamma}$ by going from u to $\mathbf{cen}(f)$ along an edge of $\mathbf{Rad}(H)$, and then from $\mathbf{cen}(f)$ to v again along an edge of $\mathbf{Rad}(H)$. Note that if v appears on the boundary of f multiple times, we may always pick the edge between $\mathbf{cen}(f)$ and v that corresponds to the direction from which v is accessed by $\vec{\gamma}$. The same holds for entering f from u. Hence, from now on whenever considering some embedded connected multigraph H, we will consider only nooses that are oriented cycles in $\mathbf{Rad}(H)$.

Given some noose $\vec{\gamma}$, we may define the *subgraph enclosed* by $\vec{\gamma}$ as the subgraph $\mathbf{enc}(\vec{\gamma}, H)$ of H consisting of all the vertices and edges of H that are embedded into $\mathbf{enc}(\vec{\gamma}) \cup \vec{\gamma}$. Similarly, the *subgraph excluded* by $\vec{\gamma}$ is the subgraph $\mathbf{exc}(\vec{\gamma}, H)$ of H that consists of all the vertices and edges of H that are embedded into $\mathbf{exc}(\vec{\gamma}) \cup \vec{\gamma}$. Note that $(E(\mathbf{enc}(\vec{\gamma}, H)), E(\mathbf{exc}(\vec{\gamma}, H)))$ is a partition of $E(H)$, and $V(\mathbf{enc}(\vec{\gamma}, H)) \cap V(\mathbf{exc}(\vec{\gamma}, H))$ consists of exactly the vertices traversed by $\vec{\gamma}$.

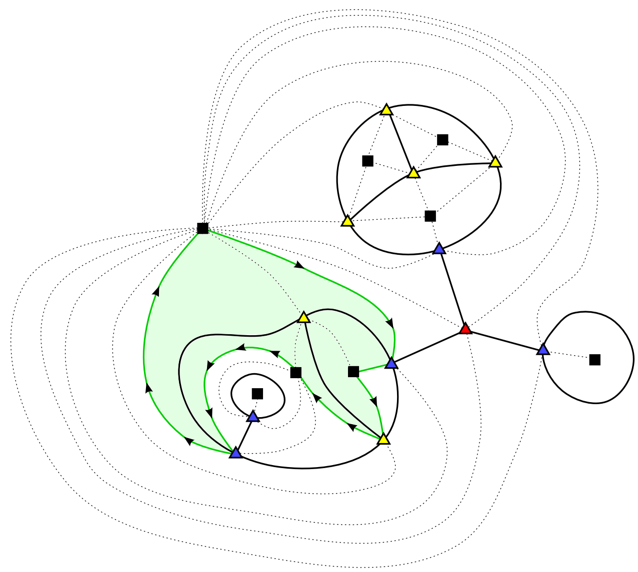


Figure 9: The Voronoi diagram from Figure 7 together with its radial graph (dotted edges) and an exemplary noose $\vec{\gamma}$. The area enclosed by $\vec{\gamma}$ is depicted in green.

Sphere cut decompositions. We now recall the main tool that will be used to justify the correctness of our algorithm, namely sphere cut decompositions. Intuitively speaking, every graph H embedded on a sphere has a hierarchical decomposition (formally, a branch decomposition), where every separator used to decompose the graph is formed by vertices traversed by some noose of length at most $\mathcal{O}(\sqrt{|V(H)|})$. In our case we will apply the sphere cut decomposition to the diagram $\widetilde{\mathcal{H}}_{\mathcal{Z}}$, where $\mathcal{Z}$ is the optimum solution to the problem. By Lemma 4.5 we have that $|V(\widetilde{\mathcal{H}}_{\mathcal{Z}})| = \mathcal{O}(k)$, which means that all the separators used in the (unknown) sphere cut decomposition of $\widetilde{\mathcal{H}}_{\mathcal{Z}}$ have length $\mathcal{O}(\sqrt{k})$. By dint of Theorem 4.7, we are able to construct a family of $d^{\mathcal{O}(\sqrt{k})}$ candidates for these separators. Then we run a recursive algorithm that at each step guesses a separator that splits the (unknown) solution in a balanced way, and then recursively solves simpler subinstances. Therefore, to see why the algorithm is correct, it is necessary to have a good understanding of noose separators in planar graphs, which brings us to the framework of sphere cut decompositions.

We now move to a formal introduction of sphere cut decompositions. Recall that for a multigraph G, a pair $\langle T, \eta \rangle$ is a *branch decomposition* of G if T is a tree with all the internal vertices of degree 3, and η is a bijection from $E(G)$ to the set of leaves of T. For an edge $e \in E(T)$, we define the *middle* *set* of e, denoted $\mathrm{mid}(e)$, as follows. Suppose that removal of e breaks T into two trees T_1, T_2, and let F_1, F_2 be the sets of edges of G mapped to the leaves of T contained in T_1 and T_2, respectively. Then (F_1, F_2) is a partition of $E(G)$. Set $\mathrm{mid}(e)$ consists of all the vertices of G that are incident both to an edge of F_1 and to an edge of F_2. The *width* of edge e is then defined as $|\mathrm{mid}(e)|$, and the *width* of decomposition $\langle T, \eta \rangle$ is the minimum width among the edges of T. The *branchwidth* of G, denoted $\mathbf{bw}(G)$, is the minimum possible width of a branch decomposition of G.

Suppose now that G is a connected multigraph without loops that is embedded on a sphere Σ. A *sphere cut decomposition* (*sc-decomposition*) of G is a triple $\langle T, \eta, \vec{\delta} \rangle$, where:

- $\langle T, \eta \rangle$ is a branch decomposition of G;
- $\vec{\delta}$ is a function that assigns to every edge $e \in E(T)$ a noose $\vec{\delta}(e)$ that traverses exactly the set of vertices $\mathrm{mid}(e)$. Moreover, if (F_1, F_2) is the partition of $E(G)$ as in the definition of $\mathrm{mid}(e)$, and G_1, G_2 are the subgraphs of G spanned by F_1, F_2, respectively, then one of G_1, G_2 is $\mathbf{enc}(\vec{\delta}(e), G)$ and the second is $\mathbf{exc}(\vec{\delta}(e), G)$.

Note that in the second condition, whether $\mathbf{enc}(\vec{\delta}(e), G)$ is G_1 or G_2 depends on the choice of direction of the noose $\vec{\delta}(e)$. Observe also that the given partition (F_1, F_2) of $E(G)$ uniquely defines the noose $\vec{\delta}(e)$ (treated as a simple cycle in $\mathbf{Rad}(G)$, without orientation), for the following reason: For every face f that is traversed by $\vec{\delta}(e)$, the intersections of ∂f with F_1 and F_2 have to be just two intervals of the walk ∂f, since face f is visited only once by $\vec{\delta}(e)$. Hence, by examining these intervals we can uniquely deduce between which vertices the noose travels within f, and from which directions these vertices are accessed.

We will regard sc-decompositions also as branch decompositions, with all the inherited notions for them.

As observed by Dorn et al. [19], the following theorem was implicitly proved by Seymour and Thomas in [45]. The improved running time (from original $\mathcal{O}(n^4)$ to $\mathcal{O}(n^3)$) is due to Gu and Tamaki [26].

Theorem 4.13 ([19, 26, 45]). *Suppose G is a bridgeless, connected multigraph G without loops, embedded on a sphere Σ. Then there exists an sc-decomposition $\langle T, \eta, \vec{\delta} \rangle$ such that $\langle T, \eta \rangle$ is a branch decomposition of G of optimum width. Moreover, such an sc-decomposition can be found in time $\mathcal{O}(n^3)$.*

Let us now deliberate on the proof of Theorem 4.13, as the statement differs in important details from the one provided by Dorn et al. [19] (see Theorem 1 in this work). The following argumentation is based on the sketch of the proof of Theorem 1 in [19].

The proof follows from (5.1) and the proof of (7.2) in [45]. In (5.1), it is proved that for every 2-connected multigraph G, there exists an optimum carving decomposition of G that uses only bond carvings (is a *bond carving decomposition*). Here, a carving decomposition is a similar concept to a branch decomposition, but this time leaves of decomposition T correspond one-to-one to vertices of G, and the width of an edge $e \in E(T)$ is defined as the number of edges traversing the corresponding partition (X_1, X_2) of $V(G)$. This partition (X_1, X_2) is called a *carving*, and the carving is a *bond carving* if both X_1 and X_2 induce connected graphs in G. Equivalently, the edges traversing the carving form an inclusion-wise minimal edge cut between some two vertices of G.

Note now that if G is embedded on a sphere Σ, then such a minimal cut corresponds to a simple cycle in the dual G^*, which traverses duals of the consecutive edges of the cut. This cycle divides the sphere into two open disks such that vertices of X_1 are embedded into one of them and vertices of X_2 are embedded into the second of them.

Carving decompositions are then translated to branch decompositions using the notion of a *medial graph* $\mathbf{Med}(G)$, which is just the dual of the radial graph. More precisely, vertices of the

$\mathbf{Med}(G)$ are the edges of the primal graph, and for every vertex $v \in V(G)$ we connect the edges incident to v into a cycle, in the order of their appearance around v in the sphere embedding. Thus, the medial graph is also embedded into the same sphere. In the proof of (7.2) in [45], Seymour and Thomas consider carving decompositions of the medial graph $\mathbf{Med}(G)$. Since the vertex set of $\mathbf{Med}(G)$ is exactly the edge set of G, carving decompositions of $\mathbf{Med}(G)$ correspond exactly to branch decompositions of G. Moreover, cycles in the dual of $\mathbf{Med}(G)$ (which is just $\mathbf{Rad}(G)$) correspond exactly to nooses in G. Hence, bond carving decompositions of $\mathbf{Med}(G)$ correspond to sc-decompositions of G. Seymour and Thomas prove that this correspondence translates an optimum bond carving decomposition of $\mathbf{Med}(G)$ into an optimum branch decomposition of G, where the width gets divided by two. In particular the carving width of $\mathbf{Med}(G)$ is twice larger than the branchwidth of G. Since this translation produces in fact an sc-decomposition, we conclude that there exists an sc-decomposition of G of optimum width.

In (9.1) Seymour and Thomas give an $\mathcal{O}(n^4)$ algorithm that constructs an optimum bond carving decomposition of an planar graph. By applying this algorithm to $\mathbf{Med}(G)$ and translating the output to an optimum sc-decomposition of G, we obtain the sought result. The running time of the algorithm has been improved from original $\mathcal{O}(n^4)$ to $\mathcal{O}(n^3)$ by Gu and Tamaki [26].

Observe now that in (5.1) of [45], it was assumed that the graph, whose carving decompositions are considered, needs to be 2-connected. Indeed, for every vertex $v \in V(G)$ we have that $(\{v\}, V(G)\setminus\{v\})$ is one of the carvings in every carving decomposition. If now v was a cutvertex, then this carving would not be a bond carving.

It is now easy to see that the medial graph $\mathbf{Med}(G)$ is 2-connected if and only if the primal graph G is connected and bridgeless. Therefore, the whole presented argumentation holds provided that G is connected and bridgeless. And indeed, if G contained some bridge b, then $(\{b\}, E(G)\setminus\{b\})$ would be one of the partitions (F_1, F_2) considered for some edge of the decomposition T. However, then the noose enclosing only the bridge b would need to visit the face on both sides of b twice, which contradicts the definition of a noose. Therefore, in Theorem 4.13 it is necessary and sufficient to assume that G is connected and bridgeless.

For the other assumptions, Dorn et al. [19] consider only simple graphs. However, all the arguments work in the same way in presence of multiple edges, since the original proof of Seymour and Thomas [45] works for multigraphs (as is noted in the first sentence of [45]). We need, however, to exclude loops, since in the presence of loops, understood as curves connecting a vertex to itself, there arise technical problems with the medial graph $\mathbf{Med}(G)$. Namely, for the natural generalization of this notion to multigraphs with loops, a loop can correspond to a cutvertex in $\mathbf{Med}(G)$. We remark that Seymour and Thomas allow loops in their work, but they understand them differently, as edges of arity 1, being de facto annotations at vertices. We also remark that in their version of Theorem 4.13, Dorn et al. [19] (incorrectly) assume only that the graph does not have vertices of degree 1, instead of excluding bridges. As we have seen, this assumption is necessary.

The second necessary ingredient is the well-known fact that planar graphs have branchwidth bounded by approximately the square root of the number of vertices.

Theorem 4.14 ([23]). *For a planar graph G on n vertices, it holds that* $\mathbf{bw}(G) \leq \sqrt{4.5n}$.

We note that Theorem 4.14 can be trivially extended to multigraphs. Thus, Theorems 4.13 and 4.14 together imply that every bridgeless, connected multigraph G without loops that is embedded on a sphere, admits an sc-decomposition of width at most $\sqrt{4.5|V(G)|}$. This corollary will be our main tool in the sequel.

4.7 Balanced nooses in plane graphs

Given a connected multigraph G embedded on a sphere, a noose $\vec{\gamma}$ w.r.t. G is *α-edge-balanced* if the numbers of edges enclosed and excluded by $\vec{\gamma}$ are bounded by $\alpha|E(G)|$, i.e. $|E(\mathbf{enc}(\vec{\gamma},G))| \leq \alpha|E(G)|$ and $|E(\mathbf{exc}(\vec{\gamma},G))| \leq \alpha|E(G)|$. The following lemma shows that the plane multigraphs we are interested in admit short edge-balanced nooses.

Theorem 4.15. *Let G be a connected 3-regular multigraph with n vertices, $m \geq 6$ edges, possibly with loops, and embedded on a sphere Σ. Then there exists a $\frac{2}{3}$-edge-balanced noose w.r.t. G that has length at most $\sqrt{4.5n}$.*

Proof. Let T^* be the tree of bridgeless components of G, where with every node $x \in V(T^*)$ we associate a subset $B_x \subseteq V(G)$ inducing a bridgeless component of G, and every edge $xy \in E(T^*)$ corresponds to a bridge b_{xy} connecting $G[B_x]$ with $G[B_y]$. For every edge $xy \in E(T^*)$, consider the connected components G_1 and G_2 of $G - b_{xy}$ and let us direct this edge from x to y if $|E(G_1)| > |E(G_2)|$, from y to x if $|E(G_1)| < |E(G_2)|$, and breaking the tie arbitrarily in case $|E(G_1)| = |E(G_2)|$. Thus, every edge of T^* becomes directed. Since T^* has exactly $|V(T^*)| - 1$ edges, we have that the sum of outdegrees in T^* is equal to $|V(T^*)| - 1$, and hence there exists a node $x_0 \in V(T^*)$ that has no outgoing directed edge.

Let $H = G[B_{x_0}]$ and observe that H is a bridgeless, connected multigraph embedded on Σ. Moreover, observe that H does not contain loops for the following reason: A loop in a 3-regular multigraph must be attached to a vertex having exactly one other incident edge that is a bridge. Hence, if H contained a loop then $B_{x_0} = V(H)$ would consist of one vertex and x_0 would be a leaf in T^*. Since the only edge incident to x_0 in T^* is directed towards x_0 and $G[B_{x_0}]$ has one edge, we would have that $m \leq 2 \cdot 1 + 1 = 3$. This would be a contradiction with the assumption that $m \geq 6$.

Suppose first that $|B_{x_0}| = 1$, i.e., H is graph consisting of a single vertex u. Then u is incident to 3 bridges in G, and removal of each of these bridges separates from u a connected component having at most $\frac{m-1}{2}$ edges. Let C be the component that has the most vertices among these three ones; then $|E(C)| \geq \frac{m-3}{3} = \frac{m}{3} - 1$. Take a noose $\vec{\gamma}$ of length 1 that visits u and the unique face incident to u, chosen in such a manner that it encloses exactly component C and the bridge connecting C with u. Then, as $m \geq 6$, it follows that

$$\begin{aligned}
|E(\mathbf{enc}(\vec{\gamma},G))| &= |E(C)| + 1 \leq \frac{m-1}{2} + 1 \leq \frac{2}{3}m, \\
|E(\mathbf{exc}(\vec{\gamma},G))| &= m - (|E(C)| + 1) \leq m - \Big(\frac{m}{3} - 1 + 1\Big) = \frac{2}{3}m.
\end{aligned}$$

Hence, $\vec{\gamma}$ satisfies the required properties.

From now on suppose that $|B_{x_0}| > 1$. We apply Theorems 4.13 and 4.14 to H in order to find an sc-decomposition $\langle T, \eta, \vec{\delta} \rangle$ of H of width at most $\sqrt{4.5|V(H)|} \leq \sqrt{4.5n}$. Define a weight function $\mathbf{w}\colon E(H) \to \mathbb{N}$ as follows: We first put $\mathbf{w}(e) = 1$ for all $e \in E(H)$. Then, for every bridge uv such that $u \in B_{x_0}$ and $v \notin B_{x_0}$, consider the connected component C of $G - uv$ that does not contain B_{x_0} and add its edge count (including the bridge uv itself) to the weight of one of the edges of H incident to u; such an edge always exists for H is bridgeless and has more than one vertex. Moreover, every vertex of H is incident either to three edges of H or to two edges of H and one bridge, and hence it is easy to see that the distribution of weights can be done in such a manner that to the weight of every edge e of H we add the weight corresponding to at most one bridge incident to an endpoint of e. Observe that thus we have that $\mathbf{w}(H) = m$, where the weight of a graph is defined as the sum of the weights of its edges. Moreover, since G is 3-regular and every edge of T^* incident to x_0 was directed towards x_0, we have that $\mathbf{w}(e) \leq 2 + \frac{m-1}{2} = \frac{m}{2} + \frac{3}{2}$ for each $e \in E(H)$.

Now, consider any edge $e \in E(T)$. Let (F_1, F_2) is the partition of $E(H)$ as in the definition of $\mathrm{mid}(e)$, and let H_1, H_2 be the subgraphs of H spanned by F_1, F_2; hence we have that one of H_1, H_2 is $\mathbf{enc}(\vec{\delta}(e), H)$ and the second is $\mathbf{exc}(\vec{\delta}(e), H)$, depending on the orientation of $\vec{\delta}(e)$. For every $e \in E(T)$, let us select this orientation so that $\mathbf{w}(\mathbf{enc}(\vec{\delta}(e), H)) \leq \mathbf{w}(\mathbf{exc}(\vec{\delta}(e), H))$, breaking ties arbitrarily in case of equality. Also, let us direct the edge e in T so that it points to the subtree of T whose leaves span $\mathbf{exc}(\vec{\delta}(e), G)$. Again, since after directing edges of T the total sum of outdegrees in $|V(T)| - 1$, there exists a node $z \in V(T)$ such that there is no outgoing edge from z in T.

Consider first the case when z is a leaf in T; let $uv = \eta(z)$ and let e_z be the unique edge of T incident to z. Hence, $\vec{\delta}(e_z)$ is a noose that excludes only the edge e_z, and encloses the whole rest of $E(H)$. Since e_z was directed towards z, we have that $\mathbf{w}(uv) = \mathbf{w}(\mathbf{exc}(\vec{\delta}(e_z), H)) \geq \frac{m}{2}$. Since $m \geq 6$, this means that one of the endpoints of uv, say u, is incident to a bridge uw such that the weight corresponding to this bridge contributed to the weight of uv. More precisely, if C is the connected component of $G - uw$ that does not contain B_{x_0}, then $\mathbf{w}(uv)$ was increased from initial value of 1 by contribution $1 + |E(C)|$. Let $\vec{\gamma}$ be the length-1 noose that visits u and the face around bridge uw, and encloses exactly $E(C)$ and the bridge uw. Then we have that $|E(\mathbf{enc}(\vec{\gamma}, G))| = |E(C)| + 1 = \mathbf{w}(uv) - 1 \leq \frac{m}{2} + \frac{3}{2} - 1 \leq \frac{2}{3}m$, as $m \geq 6$. On the other hand, $|E(\mathbf{exc}(\vec{\gamma}, G))| = m - (|E(C)| + 1) = m - (\mathbf{w}(uv) - 1) \leq m - \frac{m}{2} + 1 \leq \frac{2}{3}m$, again as $m \geq 6$. Hence, $\vec{\gamma}$ satisfies the required properties.

Finally, we are left with the case when z is an internal vertex of T. Let e_1, e_2, e_3 be the edges of T incident to z; recall that they are all directed towards z. For $t = 1, 2, 3$, let $H_t = \mathbf{enc}(\vec{\delta}(e_t), H)$. Then we have that $\mathbf{w}(H_1) + \mathbf{w}(H_2) + \mathbf{w}(H_3) = \mathbf{w}(H) = m$, and $\mathbf{w}(H_t) \leq \frac{m}{2}$ for all $t = 1, 2, 3$. W.l.o.g. suppose that $\mathbf{w}(H_1)$ is the largest among $\mathbf{w}(H_t)$ for $t = 1, 2, 3$; then $\mathbf{w}(H_1) \geq \frac{m}{3}$. Let $\vec{\gamma}' = \vec{\delta}(e_1)$, which is a noose w.r.t. H; recall that the length of $\vec{\gamma}'$ is at most $\sqrt{4.5|V(H)|} \leq \sqrt{4.5n}$. Let us create a noose $\vec{\gamma}$ w.r.t. G by modifying noose $\vec{\gamma}'$ as follows: whenever $\vec{\gamma}'$ traverses a vertex $u \in V(H)$ that in G is incident to a bridge uw connecting u with a component C, then draw $\vec{\gamma}$ around u in such a manner that it leaves uw together with the whole component C on the same side of $\vec{\gamma}$ as the edge of H whose weight C was contributing to (see Figure 10). Note that since G is 3-regular, this is always possible. In this manner, we have that $|E(\mathbf{enc}(\vec{\gamma}, G))| = \mathbf{w}(H_1)$ and $|E(\mathbf{exc}(\vec{\gamma}, G))| = \mathbf{w}(H_2) + \mathbf{w}(H_3)$, and $\vec{\gamma}$ has the same length as $\vec{\gamma}'$.

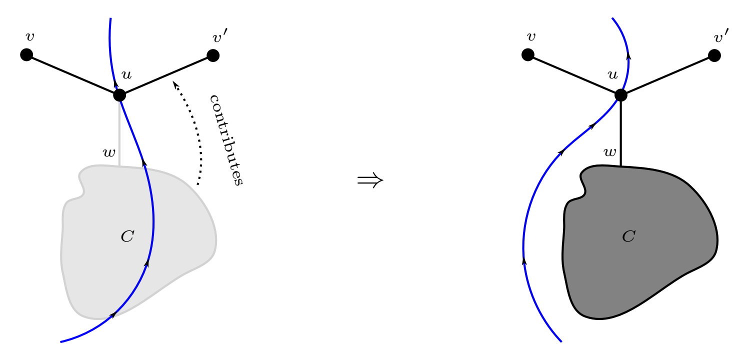


Figure 10: The modification applied to construct $\vec{\gamma}$ from $\vec{\gamma}'$ around vertices of B_{x_0} that are incident to a bridge in G. The left panel shows noose $\vec{\gamma}'$ w.r.t. H, and the right panel shows the adjusted noose $\vec{\gamma}$ after reintroducing component C.

Of course, since $\mathbf{w}(H_1) \leq \frac{m}{2}$, we have that $|E(\mathbf{enc}(\vec{\gamma}, G))| = \mathbf{w}(H_1) \leq \frac{m}{2} \leq \frac{2}{3}m$. On the other hand, since $\mathbf{w}(H_1) \geq \frac{m}{3}$, we have that $|E(\mathbf{exc}(\vec{\gamma}, G))| = m - \mathbf{w}(H_1) \leq \frac{2}{3}m$. Thus, $\vec{\gamma}$ satisfies all the required properties. $\square$

For the sake of later argumentation, we need a slightly different definition of balanceness that concerns faces instead of edges. We say that a noose $\vec{\gamma}$ is *α-face-balanced* if the numbers of faces of G that are strictly enclosed and strictly excluded by $\vec{\gamma}$, respectively, are not larger than $\alpha|F(G)|$. Here, $F(G)$ is the set of faces of G. Note that faces traversed by $\vec{\gamma}$ are neither strictly enclosed nor strictly excluded by $\vec{\gamma}$, and hence they contribute to neither of these numbers.

Lemma 4.16. *Let G be a connected 3-regular multigraph, possibly with loops, and embedded on a sphere Σ. If a noose $\vec{\gamma}$ w.r.t. G is $\frac{2}{3}$-edge-balanced, then it is also $\frac{2}{3}$-face-balanced.*

Proof. Since G is connected and 3-regular, by Euler's formula we have that $|V(G)|-|E(G)|+|F(G)| = 2$ and by degree count we have that $|E(G)| = \frac{3}{2}|V(G)|$. From this it follows that

$$|F(G)| = 2 + \frac{1}{3}|E(G)|. \tag{1}$$

As G is connected and noose $\vec{\gamma}$ visits every face of G at most once, it follows that both $G_1 := \mathbf{enc}(G, \vec{\gamma})$ and $G_2 := \mathbf{exc}(G, \vec{\gamma})$ are connected. Hence, G_1 is a connected plane multigraph, possibly with loops, with maximum degree at most 3. From Euler's formula it follows that $|V(G_1)| - |E(G_1)| + |F(G_1)| = 2$, and from the degree bound we infer that $|E(G_1)| \leq \frac{3}{2}|V(G_1)|$. Therefore,

$$\begin{aligned}
|F(G_1)| &= 2 - |V(G_1)| + |E(G_1)| \leq 2 - \frac{2}{3}|E(G_1)| + |E(G_1)| = 2 + \frac{1}{3}|E(G_1)| \\
&\leq 2 + \frac{1}{3} \cdot \frac{2}{3}|E(G)| = \frac{2}{3} + \frac{2}{3}\left(2 + \frac{1}{3}|E(G)|\right) = \frac{2}{3} + \frac{2}{3}|F(G)|;
\end{aligned}$$

here, the second inequality follows from the assumption that $\vec{\gamma}$ is $\frac{2}{3}$-edge-balanced, whereas the last equality follows from (1). Observe now that the number of faces of G strictly enclosed by $\vec{\gamma}$ is equal to $|F(G_1)| - 1$; indeed the face set of G_1 comprises exactly the faces of G strictly enclosed by $\vec{\gamma}$ plus one extra new face that contains the curve $\vec{\gamma}$. Hence, the number of faces of G strictly enclosed by $\vec{\gamma}$ is at most $\frac{2}{3} + \frac{2}{3}|F(G)| - 1 < \frac{2}{3}|F(G)|$. A symmetric argument shows that also the number of faces strictly excluded by $\vec{\gamma}$ is at most $\frac{2}{3}|F(G)|$. $\square$

From Theorem 4.15 and Lemma 4.16 we infer the following corollary.

Corollary 4.17. *Let G be a connected 3-regular multigraph with n vertices, $m \geq 6$ edges, possibly with loops, and embedded on a sphere Σ. Then there exists a $\frac{2}{3}$-face-balanced noose w.r.t. G that has length at most $\sqrt{4.5n}$.*

4.8 Voronoi separators

Armed with a good understanding of Voronoi diagrams and knowledge about the existence of short balanced nooses in plane graphs, we can combine these two ingredients to design a Divide&Conquer algorithm for Disjoint Network Coverage. More precisely, if $\mathcal{Z}$ is the optimum solution to the considered instance of Disjoint Network Coverage, then the algorithm will iterate through all the possible candidates for balanced nooses of the Voronoi diagram $\widetilde{\mathcal{H}}_{\mathcal{Z}}$. Each candidate noose will separate the instance into a number of subinstances, which will be solved recursively. For the correct selection of the balanced noose of $\widetilde{\mathcal{H}}_{\mathcal{Z}}$, each of the subinstances will contain only at most $\frac{2}{3}k$ objects from the solution, and hence we will apply the algorithm recursively only to instances with at most this value of the parameter; This will ensure that the running time is as promised in Theorem 4.

However, first we need to understand formally what are the "possible candidates" for balanced nooses, and in what sense they separate the instance at hand into subinstances; these two questions are the topics of the current and the following section. More precisely, we will investigate the properties of *Voronoi separators*, which are structures to be used as separators in the forthcoming Divide&Conquer algorithm.

Whenever we have some normal subfamily of objects $\mathcal{F} \subseteq \mathcal{D}$, then we have a corresponding Voronoi partition $\mathbb{M}_{\mathcal{F}}$ and Voronoi diagram $\widetilde{\mathcal{H}}_{\mathcal{F}}$. The intuition now is that the nooses w.r.t. $\widetilde{\mathcal{H}}_{\mathcal{F}}$ can be projected back to the original graph G. That is, suppose in some noose $\vec{\gamma}$ we travel from some center of a face $\mathbf{cen}(f)$ to an adjacent vertex u. Then in G this corresponds to traveling from the center $\mathbf{cen}(p)$ of the corresponding object p to the corresponding branching point of $\widetilde{\mathcal{H}}_{\mathcal{F}}$ using a path inside the tree $\widehat{T}_{\mathcal{F}}(p)$. The crucial observation now is that, for a noose $\vec{\gamma}$, the knowledge of the order in which $\vec{\gamma}$ visits faces of $\widetilde{\mathcal{H}}_{\mathcal{F}}$ (corresponding to objects of $\mathcal{F}$) and branching points of $\widetilde{\mathcal{H}}_{\mathcal{F}}$, is sufficient to deduce the whole projection to G. This is because when projecting we can simply use shortest paths.

Voronoi separators. We move on to a formal argumentation. A *Voronoi separator* of *length* r is a sequence S of the following form:

$$\begin{aligned} S = \langle \quad & p_1, u_1, f_1, v_1, \\ & p_2, u_2, f_2, v_2, \\ & \qquad \ldots \\ & p_r, u_r, f_r, v_r \quad \rangle \end{aligned} \tag{2}$$

For this sequence, we put the following requirements (from now on, indices behave cyclically):

(a) $p_1, p_2, \ldots, p_r$ are pairwise disjoint objects from $\mathcal{D}$, forming a normal family $\mathcal{D}(S) = \{p_1, p_2, \ldots, p_r\}$.
(b) $f_1, f_2, \ldots, f_r$ are pairwise different faces of G.
(c) For $t = 1, 2, \ldots, r$, u_t and v_t are two different vertices on face f_t. Moreover, $v_{t-1}, u_t \in \mathbb{M}_{\mathcal{D}(S)}(p_t)$, for all $t = 1, 2, \ldots, r$. In other words, v_{t-1} and u_t are closest to the location of p_t in terms of the distance measure $\phi(v, p) = \mathrm{dist}(v, \mathbf{loc}(p)) - r(p)$, among all the locations of objects traversed by S.

We shall treat two Voronoi separators that differ only in a cyclic shift of the sequence as the same separator. However, similarly as with nooses, reverting the sequence results in a different separator, which we shall denote by S^{-1}. Observe that, given a sequence like in (2), we can verify in polynomial time whether it satisfies all the required properties of a Voronoi separator.

For $t = 1, 2, \ldots, r$, let P_t be the path from $\mathbf{cen}(p_t)$ to u_t obtained by concatenating a path inside $T(p_t)$ from $\mathbf{cen}(p_t)$ to the vertex of $\mathbf{loc}(p_t)$ closest to u_t, with the shortest path from u_t to $\mathbf{loc}(p_t)$. Similarly define Q_{t+1} for v_t and p_{t+1}. By property (c), paths P_t and Q_t are subpaths of the tree $\widehat{T}_{\mathcal{D}(S)}(p_t)$, for every $t = 1, 2, \ldots, t$. Consequently, paths $\{P_t, Q_t\}_{t=1,2,\ldots,r}$ are all pairwise vertex-disjoint, apart from paths P_t and Q_t that can intersect on a common prefix from the side of $\mathbf{cen}(p_t)$, and otherwise they are vertex-disjoint.

For a Voronoi separator S, the *perimeter* of S is defined as $\Gamma(S) = \bigcup_{t=1}^{r} V(P_t) \cup V(Q_t)$, i.e., it is the union of the vertex sets of all the paths P_t and Q_t. Moreover, we define a directed closed walk $W(S)$ in G by concatenating consecutive paths $Q_1, P_1, Q_2, P_2, \ldots, Q_t, P_t$, where P_t and Q_{t+1} are joined using edge $u_t v_t$, which lies on the boundary of face f_t. Note that $V(W(S)) = \Gamma(S)$. See Figure 11 for an example.

Note that, by the definition of a Voronoi separator, $W(S)$ is a simple cycle in G, possibly with simple paths attached to different vertices that correspond to common prefixes of paths P_t and Q_t. In particular, it still holds that removal of $W(S)$ from Σ partitions Σ into two open disks, one on

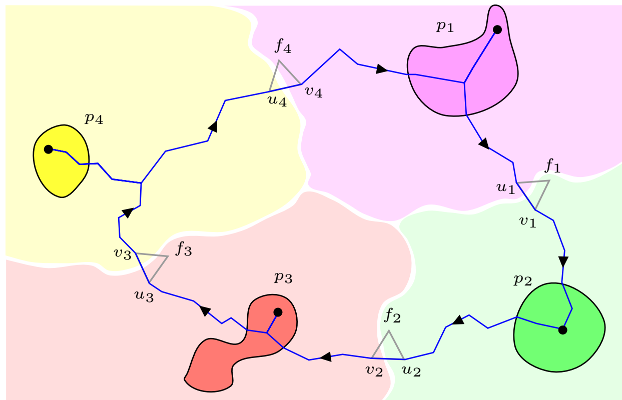


Figure 11: An exemplary Voronoi separator S of length 4, together with the corresponding walk $W(S)$ (blue). The Voronoi regions of $\mathbb{M}_{\mathcal{D}(S)}$ have been depicted in respective light colors.

the right side of $W(S)$ (denoted $\mathbf{enc}(S)$), and the second on the left side (denoted $\mathbf{exc}(S)$). For any object $p \in \mathcal{D} \setminus \mathcal{D}(S)$, we shall say that p is *strictly enclosed* by S if all the vertices of $\mathbf{loc}(p)$ are embedded into $\mathbf{enc}(S)$ (in particular $\mathbf{loc}(p) \cap \Gamma(S) = \emptyset$). Similarly, $p \in \mathcal{D} \setminus \mathcal{D}(S)$ is *strictly excluded* by S if all the vertices of $\mathbf{loc}(p)$ are embedded into $\mathbf{exc}(S)$. Finally, a client $q \in \mathcal{C}$ is *enclosed* by S, resp. *excluded* by S, if $\mathbf{pla}(q)$ is embedded into $\overline{\mathbf{enc}(S)}$, resp. into $\overline{\mathbf{exc}(S)}$; Note that thus every client q such that $\mathbf{pla}(q) \in \Gamma(S)$ is both enclosed and excluded by S. Observe also that any path in G that connects the location of a strictly enclosed object with the location of another, strictly excluded object, must necessarily cross walk $W(S)$, so it has to traverse a vertex of $\Gamma(S)$. The same observation holds also for paths connecting the location of a strictly enclosed object with the placement of an excluded client, and vice versa.

Separators inherited from the diagram. Assume now that we have a normal subfamily of objects $\mathcal{F} \subseteq \mathcal{D}$, and let $\mathbb{M} = \mathbb{M}_{\mathcal{F}}$ and $\widetilde{\mathcal{H}} = \widetilde{\mathcal{H}}_{\mathcal{F}}$ be the corresponding Voronoi partition and diagram. Suppose further that $\vec{\gamma}$ is a noose w.r.t. $\widetilde{\mathcal{H}}$. We define the corresponding Voronoi separator $S(\vec{\gamma})$ as follows; see Figure 12 for a visualization.

Suppose that $\vec{\gamma}$ visits faces and branching points $f_1^*, f_1, f_2^*, f_2, \ldots, f_r^*, f_r$ in this order, where f_t^* are different faces of $\widetilde{\mathcal{H}}$, and f_t are different branching points of $\widetilde{\mathcal{H}}$, that is, faces of G (recall that $\widetilde{\mathcal{H}}$ is constructed from the dual of G). By Lemmas 4.4 and 4.5, faces of $\widetilde{\mathcal{H}}$ correspond one-to-one to objects of $\mathcal{F}$. Let $p_1, p_2, \ldots, p_r \in \mathcal{F}$ be the objects that correspond to $f_1^*, f_2^*, \ldots, f_r^*$, respectively. Suppose that when entering f_t from f_t^*, for some $t \in \{1, 2, \ldots, r\}$, noose $\vec{\gamma}$ entered it between edges $\tilde{e}_1^*$ and $\tilde{e}_2^*$. Let e_1^* and e_2^* be the edges of the prediagram $\mathcal{H}_{\mathcal{F}}$ incident to f_t that got contracted onto $\tilde{e}_1^*$ and $\tilde{e}_2^*$ respectively; in case $\tilde{e}_1^*$ or $\tilde{e}_2^*$ is a loop, we choose the edge of $\mathcal{H}_{\mathcal{F}}$ corresponding to the endpoint of the loop by which f_t was entered from f_t^*. Then we define u_t to be the common endpoint of the primal edges e_1 and e_2 corresponding to e_1^* and e_2^*, respectively (recall that f is a triangle, since G is triangulated). We define v_t analogically, based on how $\vec{\gamma}$ leaves branching point f_t to face f_{t+1}^*. Note that since a noose, when crossing some branching point, never leaves this branching point to the same face and using the same direction as it entered, we have that $u_t \neq v_t$ for all $t = 1, 2, \ldots, r$. Then separator $S(\vec{\gamma})$ is defined using p_t, f_t, u_t, and v_t as in formula (2). Note that $S(\vec{\gamma})$ has the same length as $\vec{\gamma}$, and moreover, $S(\vec{\gamma})^{-1} = S(\vec{\gamma}^{-1})$. The following lemma encapsulates the main properties of separators inherited from the diagram that we shall use later on.

Lemma 4.18. *If $\vec{\gamma}$ is a noose w.r.t. $\widetilde{\mathcal{H}}$, then:*

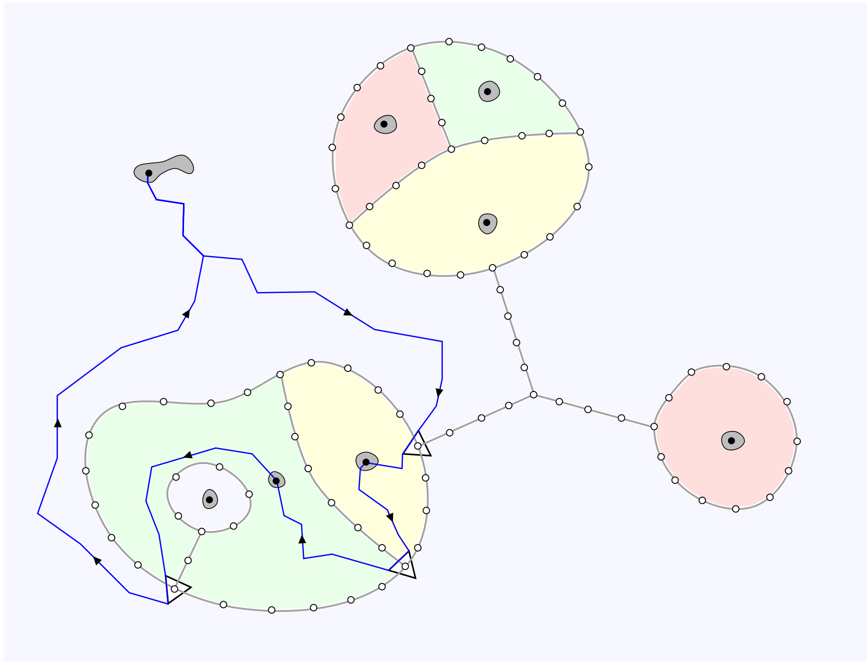


Figure 12: Separator $S(\vec{\gamma})$ and corresponding walk $W(S(\vec{\gamma}))$ for the noose $\vec{\gamma}$ from Figure 9.

(i) $S := S(\vec{\gamma})$ *is a Voronoi separator;*

(ii) $\Gamma(S) \subseteq \bigcup_{p\in\mathcal{D}(S)} \mathbb{M}_{\mathcal{F}}(p)$*;*

(iii) *For every object* $p \in \mathcal{F}$ *such that the face of* $\widetilde{\mathcal{H}}$ *corresponding to* p *is strictly enclosed by* $\vec{\gamma}$ *(resp. strictly excluded by* $\vec{\gamma}$*), we have that* p *is strictly enclosed by* S *(resp. strictly excluded by* S*).*

Proof. For (i), the only non-trivial condition is the last one, that $u_t, v_{t-1} \in \mathbb{M}_{\mathcal{D}(S)}(p_t)$ for every $t = 1, 2, \ldots, r$. However, from the construction of u_t and v_{t-1} and properties of the Voronoi diagram (see Lemmas 4.4 and 4.5), it follows that $u_t, v_{t-1} \in \mathbb{M}_{\mathcal{F}}(p_t)$. Since $\mathcal{D}(S) \subseteq \mathcal{F}$, the fact that also $u_t, v_{t-1} \in \mathbb{M}_{\mathcal{D}(S)}(p_t)$ follows from Lemma 4.3.

For (ii), in the paragraph above we have argued that $u_t, v_{t-1} \in \mathbb{M}_{\mathcal{F}}(p_t)$ for all $t = 1, 2, \ldots, r$. By Lemma 4.2, also the unique shortest paths from u_t and v_{t-1} to $\mathbf{loc}(p_t)$ are entirely contained in $\mathbb{M}_{\mathcal{F}}(p_t)$. As $\mathbf{loc}(p_t) \subseteq \mathbb{M}_{\mathcal{F}}(p_t)$, we have that $V(Q_t), V(P_t) \subseteq \mathbb{M}_{\mathcal{F}}(p_t)$, and hence $\Gamma(S) \subseteq \bigcup_{p\in\mathcal{D}(S)} \mathbb{M}_{\mathcal{F}}(p)$.

Property (iii) follows directly from (ii) and the construction of the Voronoi diagram $\widetilde{\mathcal{H}}$. □

4.9 Interaction graph and separability of the problem

Suppose $(G, \mathcal{D}, \mathcal{C}, k)$ is an instance of Disjoint Network Coverage. By the *interaction graph* $\mathcal{L} = \mathcal{L}(G, \mathcal{D}, \mathcal{C}, k)$ we mean a graph with vertex set $\mathcal{D} \cup \mathcal{C}$, where (a) for $p \in \mathcal{D}$ and $q \in \mathcal{C}$ we put $pq \in E(\mathcal{L})$ if and only if object p covers client q, and (b) for $p, p' \in \mathcal{D}$ we put $pp' \in E(\mathcal{L})$ if and only if p and p' contradict the normality requirement, i.e., family $\{p, p'\}$ is not normal.

We now define the final abstraction of a separator, which we call a *guarded separator*. A guarded separator is simply a pair $\mathbb{X} = (\mathcal{Q}, \Gamma)$, where $\mathcal{Q} \subseteq \mathcal{D}$ is a normal subfamily of objects and $\Gamma \subseteq V(G)$ is an arbitrary subset of vertices. We will use operators $\mathcal{Q}(\mathbb{X})$ and $\Gamma(\mathbb{X})$ to extract the first and the second coordinate of pair $\mathbb{X}$, respectively. The *length* of a guarded separator is simply $|\mathcal{Q}(\mathbb{X})|$. Note that every Voronoi separator S naturally induces a guarded separator $\mathbb{X}(S) = (\mathcal{D}(S), \Gamma(S))$; this will be the main source of guarded separators in our algorithm.

Let $\mathbb{X} = (\mathcal{Q}, \Gamma)$ be a guarded separator. We say that a client $q \in \mathcal{C}$ is *covered* by $\mathbb{X}$ if q is covered by an object belonging to $\mathcal{Q}$. We also say that an object $p \in \mathcal{D} \setminus \mathcal{Q}$ is *banned* by $\mathbb{X}$ if either $\mathcal{Q} \cup \{p\}$ is not normal, or there exists $v \in \Gamma$ such that $\mathrm{dist}(v, \mathbf{loc}(p)) - r(p) < \mathrm{dist}(v, \mathbf{loc}(p')) - r(p')$, where p' is the object of $\mathcal{Q}$ closest to v in terms of the distance measure $\phi(v, p') = \mathrm{dist}(v, \mathbf{loc}(p')) - r(p')$. An object $p \in \mathcal{D} \setminus \mathcal{Q}$ that is not banned by $\mathbb{X}$ is said to be *allowed* by p. By $\mathbf{cov}(\mathbb{X})$ we denote the set of clients covered by $\mathbb{X}$, and by $\mathbf{ban}(\mathbb{X})$ the set of objects banned by $\mathbb{X}$. Obviously, by Lemma 4.1 every object $p \in \mathcal{D} \setminus \mathcal{Q}$ such that $\mathbf{loc}(p) \cap \Gamma \neq \emptyset$ is banned by $\mathbb{X}$.

We say that two allowed objects $p, p' \in \mathcal{D} \setminus \mathcal{Q}$ are *separated* by $\mathbb{X}$ if the shortest path in G between $\mathbf{loc}(p)$ and $\mathbf{loc}(p')$ traverses a vertex of $\Gamma(\mathbb{X})$. Similarly, an allowed object $p \in \mathcal{D} \setminus \mathcal{Q}$ and a client $q \in \mathcal{C}$ are *separated* by $\mathbb{X}$ if the shortest path between $\mathbf{loc}(p)$ and $\mathbf{pla}(q)$ traverses a vertex of $\Gamma(\mathbb{X})$.

The intuition is that if a Voronoi separator S is inherited from some solution, then the objects banned by $\mathbb{X}(S)$ for sure are not used by the solution. This is because the inclusion of any banned object would either contradict the normality, or the fact that walk $W(S)$ traverses only the Voronoi regions corresponding to $\mathcal{D}(S)$ in the diagram induced by the whole solution. Thus, the clients covered by S and the object banned by S form a "border" that separates the part of the solution enclosed by S from the part excluded by S. Here, by "separates" we mean the definition of separation for the guarded separator $\mathbb{X}(S)$. In the following two lemmas we formalize this separation property.

Lemma 4.19. *Suppose $\mathbb{X} = (\mathcal{Q}, \Gamma)$ is a guarded separator. Suppose further that an allowed object $p \in \mathcal{D} \setminus \mathcal{Q}$ and a client $q \in \mathcal{C}$ are separated by $\mathbb{X}$. Then the following implication holds: if p covers q, then q is covered by $\mathbb{X}$.*

Proof. Suppose p covers q, and let P be the shortest path in G between $\mathbf{loc}(p)$ and $\mathbf{pla}(q)$. By the definition of separation, we have that P traverses a vertex of Γ, say v. Let p' be the object that is closest to v among objects of $\mathcal{Q}$ in terms of $\phi(v, p')$. Since p is allowed by $\mathbb{X}$, we have that

$$\mathrm{dist}(v, \mathbf{loc}(p)) - r(p) > \mathrm{dist}(v, \mathbf{loc}(p')) - r(p'). \tag{3}$$

Since p covers q, we also have that

$$\mathrm{dist}(\mathbf{pla}(q), \mathbf{loc}(p)) \leq s(q) + r(p). \tag{4}$$

Finally, since P is the shortest path between $\mathbf{loc}(p)$ and $\mathbf{pla}(q)$, we have that

$$\mathrm{dist}(\mathbf{pla}(q), \mathbf{loc}(p)) = \mathrm{dist}(v, \mathbf{loc}(p)) + \mathrm{dist}(\mathbf{pla}(q), v). \tag{5}$$

Using (3), (4), and (5), we infer that

$$\begin{aligned}
\mathrm{dist}(\mathbf{pla}(q), \mathbf{loc}(p')) &\leq \mathrm{dist}(v, \mathbf{pla}(q)) + \mathrm{dist}(v, \mathbf{loc}(p')) \\
&< \mathrm{dist}(v, \mathbf{pla}(q)) + \mathrm{dist}(v, \mathbf{loc}(p)) + r(p') - r(p) \\
&= \mathrm{dist}(\mathbf{pla}(q), \mathbf{loc}(p)) + r(p') - r(p) \\
&\leq s(q) + r(p').
\end{aligned}$$

This means that client q is covered by $p' \in \mathcal{Q}$. $\square$

Lemma 4.20. *Suppose $\mathbb{X} = (\mathcal{Q}, \Gamma)$ is a guarded separator. Suppose further that allowed objects $p_1, p_2 \in \mathcal{D} \setminus \mathcal{Q}$ are separated by $\mathbb{X}$. Then the family $\{p_1, p_2\}$ is normal.*

Proof. Let P be the shortest path in G between $\mathbf{loc}(p_1)$ and $\mathbf{loc}(p_2)$. By the definition of separation, we have that P traverses some vertex of Γ, say v. Let p' be the object of $\mathcal{Q}$ that is closest to v, in terms of $\phi(v, p')$. Since both p_1 and p_2 are allowed by $\mathbb{X}$, we have that

$$\begin{aligned}
\mathrm{dist}(\mathbf{loc}(p_1), v) + \mathrm{dist}(\mathbf{loc}(p'), v) &\geq \mathrm{dist}(\mathbf{loc}(p_1), \mathbf{loc}(p')) \geq |r(p_1) - r(p')| && (6)\\
\mathrm{dist}(\mathbf{loc}(p_2), v) + \mathrm{dist}(\mathbf{loc}(p'), v) &\geq \mathrm{dist}(\mathbf{loc}(p_2), \mathbf{loc}(p')) \geq |r(p_2) - r(p')| && (7)\\
\mathrm{dist}(\mathbf{loc}(p_1), v) - r(p_1) &> \mathrm{dist}(\mathbf{loc}(p'), v) - r(p') && (8)\\
\mathrm{dist}(\mathbf{loc}(p_2), v) - r(p_2) &> \mathrm{dist}(\mathbf{loc}(p'), v) - r(p'). && (9)
\end{aligned}$$

Since P is the shortest path between $\mathbf{loc}(p_1)$ and $\mathbf{loc}(p_2)$, we have

$$\mathrm{dist}(\mathbf{loc}(p_1), \mathbf{loc}(p_2)) = \mathrm{dist}(v, \mathbf{loc}(p_1)) + \mathrm{dist}(v, \mathbf{loc}(p_2)). \tag{10}$$

By adding (7) and (8), we obtain:

$$\begin{aligned}
& \mathrm{dist}(\mathbf{loc}(p_1), v) + \mathrm{dist}(\mathbf{loc}(p_2), v) + \mathrm{dist}(\mathbf{loc}(p'), v) - r(p_1)\\
> \; & |r(p_2) - r(p')| + \mathrm{dist}(\mathbf{loc}(p'), v) - r(p')\\
\geq \; & \mathrm{dist}(\mathbf{loc}(p'), v) - r(p_2).
\end{aligned}$$

Thus, by (10) we obtain that

$$\mathrm{dist}(\mathbf{loc}(p_1), \mathbf{loc}(p_2)) \geq r(p_1) - r(p_2). \tag{11}$$

By adding equations (6) and (9), we obtain:

$$\begin{aligned}
& \mathrm{dist}(\mathbf{loc}(p_1), v) + \mathrm{dist}(\mathbf{loc}(p_2), v) + \mathrm{dist}(\mathbf{loc}(p'), v) - r(p_2)\\
> \; & |r(p_1) - r(p')| + \mathrm{dist}(\mathbf{loc}(p'), v) - r(p')\\
\geq \; & \mathrm{dist}(\mathbf{loc}(p'), v) - r(p_1)
\end{aligned}$$

Again by (10) we obtain that

$$\mathrm{dist}(\mathbf{loc}(p_1), \mathbf{loc}(p_2)) \geq r(p_2) - r(p_1). \tag{12}$$

Equations (11) and (12) together imply that $\mathrm{dist}(\mathbf{loc}(p_1), \mathbf{loc}(p_2)) \geq |r(p_1) - r(p_2)|$, which means that $\{p_1, p_2\}$ is normal. $\square$

For a guarded separator $\mathbb{X}$, we will consider the modified interaction graph $\mathcal{L}(\mathbb{X}) = \mathcal{L} - (\mathcal{Q}(\mathbb{X}) \cup \mathbf{ban}(\mathbb{X}) \cup \mathbf{cov}(\mathbb{X}))$. Let $\mathcal{F} \subseteq \mathcal{D}$ be a normal subfamily of objects, and let $\mathbb{M} = \mathbb{M}_{\mathcal{F}}$ and $\widetilde{\mathcal{H}} = \widetilde{\mathcal{H}}_{\mathcal{F}}$ be the corresponding Voronoi partition and diagram. We will say that a guarded separator $\mathbb{X}$ is *compatible* with $\mathcal{F}$ if $\mathcal{Q}(\mathbb{X}) \subseteq \mathcal{F}$ and $\Gamma(\mathbb{X}) \subseteq \bigcup_{p \in \mathcal{Q}(\mathbb{X})} \mathbb{M}_{\mathcal{F}}(p)$. We will moreover say that a guarded separator $\mathbb{X}$ compatible with $\mathcal{F}$ is *α-interaction-balanced w.r.t.* $\mathcal{F}$ if every connected component of $\mathcal{L}(\mathbb{X})$ contains at most $\alpha|\mathcal{F}|$ objects from $\mathcal{F}$. Observe that if $\mathbb{X}$ is compatible with $\mathcal{F}$, then $\mathcal{F} \cap \mathbf{ban}(\mathbb{X}) = \emptyset$ by the definitions of banned objects and compatibility.

Lemma 4.21. *If $\vec{\gamma}$ is an α-face-balanced noose in $\widetilde{\mathcal{H}}$, then $\mathbb{X}(S(\vec{\gamma}))$ is compatible and α-interaction-balanced w.r.t. $\mathcal{F}$.*

Proof. Let $S = S(\vec{\gamma})$ and $\mathbb{X} = \mathbb{X}(S)$. Lemma 4.18 ((i) and (ii)) implies that $\mathbb{X}$ is compatible with $\mathcal{F}$, so by the definition of banned objects we have that $\mathcal{F} \cap \mathbf{ban}(\mathbb{X}) = \emptyset$. Hence $\mathcal{F} \setminus \mathcal{D}(S) \subseteq V(\mathcal{L}(\mathbb{X}))$.

Observe that whenever we have two allowed objects $p_1, p_2 \in \mathcal{F} \setminus \mathcal{D}(S)$ such that p_1 is strictly enclosed by S and p_2 is strictly excluded by S, then $\mathbf{loc}(p_1)$ and $\mathbf{loc}(p_2)$ lie entirely in two different connected components of $G - \Gamma(S)$, since any path connecting $\mathbf{loc}(p_1)$ and $\mathbf{loc}(p_2)$ has to cross the walk $W(S)$. Therefore, any such objects p_1, p_2 are separated by $\mathbb{X}$, and Lemma 4.20 implies that they are not adjacent in $\mathcal{L}(\mathbb{X})$. Similarly, if an allowed object $p \in \mathcal{F} \setminus \mathcal{D}(S)$ is strictly enclosed by S and a client $q \in \mathcal{C}$ is excluded by S (or vice versa), then they also lie in different connected components of $G - \Gamma(S)$ (or q lies on $\Gamma(S)$), and hence are separated by $\mathbb{X}$. Then Lemma 4.19 implies that p and q are not adjacent in $\mathcal{L}(\mathbb{X})$.

Concluding, every connected component of $\mathcal{L}(\mathbb{X})$ either consists only of objects strictly enclosed by S and clients enclosed by S, or of objects strictly excluded by S and clients excluded by S. Lemma 4.18 (iii) ensures that objects of $\mathcal{F}$ corresponding to faces of $\widetilde{\mathcal{H}}$ strictly enclosed (resp. excluded) by $\vec{\gamma}$ are exactly the objects of $\mathcal{F}$ that are strictly enclosed (resp. excluded) by S. Since $\vec{\gamma}$ is α-face-balanced, we infer that there are at most $\alpha|\mathcal{F}|$ objects of $\mathcal{F}$ that are strictly enclosed by S, and the same holds also for objects strictly excluded by $\mathcal{F}$. Hence, every connected component of $\mathcal{L}(\mathbb{X})$ can contain at most $\alpha|\mathcal{F}|$ objects of $\mathcal{F}$, and $\mathbb{X}$ is α-interaction-balanced. □

We are finally ready to prove the main result of this section, that is, an enumeration algorithm for candidates for balanced guarded separators. This is the result whose simplified variant was Lemma 2.1.

Theorem 4.22. *There exists an algorithm that, given an instance $\mathcal{I} = (G, \mathcal{D}, \mathcal{C}, k)$ of* Disjoint Network Coverage *with $k \geq 4$, enumerates a family $\mathcal{N}$ of guarded separators with the following properties:*

(i) $|\mathcal{N}| \leq (2d)^{15\sqrt{k}}$; and
(ii) for every normal subfamily $\mathcal{F} \subseteq \mathcal{D}$ of cardinality exactly k, there exists a guarded separator $\mathbb{X} \in \mathcal{N}$ that is $\frac{2}{3}$-interaction-balanced for $\mathcal{F}$.

The algorithm works in total time $(2d)^{15\sqrt{k}} \cdot (dcn)^{\mathcal{O}(1)}$ and outputs the guarded separators of $\mathcal{N}$ one by one, using additional (working tape) space $(dcn)^{\mathcal{O}(1)}$.

Proof. We first apply the algorithm of Theorem 4.7 to compute a family $\mathcal{E}$ of important faces of size at most d^4. Then, we enumerate all candidates sequences for Voronoi separators of length at most $3\sqrt{k}$ as in formula (2), where faces f_t are chosen from the family $\mathcal{E}$. Note that for a fixed length r ($1 \leq r \leq 3\sqrt{k}$), we have at most $d^r \cdot d^{4r} \cdot 6^r$ such candidates, since every object p_t is chosen among d options, every face f_t is chosen among at most d^4 options, and for choosing vertices u_t, v_t on face f_t we have 6 options. Observe that

$$\sum_{r=1}^{\lfloor 3\sqrt{k} \rfloor} (6d^5)^r \leq \sum_{r=1}^{\lfloor 3\sqrt{k} \rfloor} \frac{1}{4}(2d)^{5r} \leq (2d)^{15\sqrt{k}}.$$

As family $\mathcal{N}$, we output guarded separators $\mathbb{X}(S)$ for all the candidates S that are indeed a Voronoi separator; recall that this property can be checked in polynomial time, as well as the construction of $\mathbb{X}(S)$ takes polynomial time. Thus we have $|\mathcal{N}| \leq (2d)^{15\sqrt{k}}$, which is exactly property (i). Moreover, $\mathcal{N}$ can be enumerated by examining the candidates one by one within the required time and additional space. We are left with proving property (ii).

Take any normal subfamily $\mathcal{F} \subseteq \mathcal{D}$ with $|\mathcal{F}| = k$, and let $\mathbb{M} = \mathbb{M}_{\mathcal{F}}$ and $\widetilde{\mathcal{H}} = \widetilde{\mathcal{H}}_{\mathcal{F}}$ be the corresponding Voronoi partition and diagram. By Lemma 4.5, we have that $\widetilde{\mathcal{H}}$ is a connected, 3-regular multigraph (possibly with loops), $|V(\widetilde{\mathcal{H}})| = 2|\mathcal{F}| - 4 = 2k - 4$, and $|E(\widetilde{\mathcal{H}})| = 3|\mathcal{F}| - 6 = 3k - 6$. Since $k \geq 4$, we have that $|E(\widetilde{\mathcal{H}})| \geq 6$ and Corollary 4.17 asserts that there exists a $\frac{2}{3}$-face-balanced noose $\vec{\gamma}$ w.r.t. $\widetilde{\mathcal{H}}$ that has length at most $\sqrt{4.5|V(\widetilde{\mathcal{H}})|} \leq 3\sqrt{k}$. By Lemma 4.18(i), $S(\vec{\gamma})$ is a Voronoi separator of the same length as $\vec{\gamma}$. Since $\mathcal{F} \subseteq \mathcal{D}$ is a normal subfamily, by Theorem 4.7 we have that $V(\widetilde{\mathcal{H}}) \subseteq \mathcal{E}$, and hence $S(\vec{\gamma})$ uses only faces from $\mathcal{E}$. This means that $\mathbb{X}(S(\vec{\gamma})) \in \mathcal{N}$. By Lemma 4.21, we have that guarded separator $\mathbb{X}(S(\vec{\gamma}))$ is $\frac{2}{3}$-interaction-balanced w.r.t. $\mathcal{F}$, and hence property (ii) is proven. □

4.10 The algorithm

We first briefly discuss the intuition. Suppose $\mathcal{Z}$ is a solution to the considered instance $(G, \mathcal{D}, \mathcal{C}, k)$, i.e., it is a normal family of exactly k objects from $\mathcal{D}$. We can assume that $k \geq 4$, since otherwise the instance can be solved in polynomial time by brute force. Theorem 4.22 gives us a method to enumerate a small family of candidate guarded separators with a promise, that one of them separates the optimum solution evenly. More precisely, there is a guarded separator whose objects all belong to the optimum solution $\mathcal{Z}$, and moreover after inferring all the information from this fact, i.e., excluding banned objects and covered clients, we arrive at the situation where every connected component of the remaining interaction graph contains only two thirds of the objects of $\mathcal{Z}$. This gives rise to a natural Divide&Conquer algorithm: For every guarded separator $\mathbb{X}$ output by Theorem 4.22, we recurse into all the components of the modified interaction graph $\mathcal{L}(\mathbb{X})$ for all parameters between 0 and $\lfloor \frac{2}{3}k \rfloor$, and then combine the results using a standard knapsack dynamic programming. Note that in the new subinstances solved recursively we only modify the sets of objects and clients, whereas the underlying graph G remains unchanged. We now proceed to implementing this plan formally.

Let us first introduce some notation. For an instance $\mathcal{I} = (G, \mathcal{D}, \mathcal{C}, k)$ of Disjoint Network Coverage, by $\mathbf{Val}[\mathcal{I}]$ we denote the maximum revenue of a solution in $\mathcal{I}$ that should be reported by the algorithm. When $\mathcal{Z} \subseteq \mathcal{D}$ is a normal subfamily of objects, by $\mathbf{Val}[\mathcal{I}, \mathcal{Z}]$ we denote the revenue of $\mathcal{Z}$ in $\mathcal{I}$. By $\mathcal{N}$ we denote the family of guarded separators computed for instance $\mathcal{I}$ using the algorithm of Theorem 4.22. Suppose $\mathbb{X}$ is some guarded separator in $\mathcal{N}$. By $\mathrm{cc}(\mathbb{X})$ we denote the set of connected components of $\mathcal{L}(\mathbb{X}) = \mathcal{L} - (\mathcal{Q}(\mathbb{X}) \cup \mathbf{ban}(\mathbb{X}) \cup \mathbf{cov}(\mathbb{X}))$, where $\mathcal{L}$ is the interaction graph of instance $\mathcal{I}$. For every $C \in \mathrm{cc}(\mathbb{X})$, by $\mathcal{D}(C)$ and $\mathcal{C}(C)$ we denote the sets of objects and clients in C, respectively. Finally, we shall say that a vector of nonnegative integers $\mathbf{k} = (k_C)_{C \in \mathrm{cc}(\mathbb{X})}$ is *compatible with* $\mathbb{X}$ if $k_C \leq \frac{2}{3}k$ for each $C \in \mathrm{cc}(\mathbb{X})$ and $\sum_{C \in \mathrm{cc}(\mathbb{X})} k_C = k - |\mathcal{Q}(\mathbb{X})|$. The set of vectors compatible with $\mathbb{X}$ will be denoted by $\mathrm{comp}(\mathbb{X})$.

The following lemma is the main argument justifying the correctness of the algorithm.

Lemma 4.23. *Provided $k \geq 4$, the following recursive formula holds for every instance $(G, \mathcal{D}, \mathcal{C}, k)$ of* Disjoint Network Coverage*:*

$$\mathbf{Val}[G, \mathcal{D}, \mathcal{C}, k] = \max_{\mathbb{X} \in \mathcal{N}} \max_{\mathbf{k} \in comp(\mathbb{X})} \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C \in cc(\mathbb{X})} \mathbf{Val}[G, \mathcal{D}(C), \mathcal{C}(C), k_C] \tag{13}$$

Proof. Let $\mathcal{I} = (G, \mathcal{D}, \mathcal{C}, k)$. For a connected component C on the right-hand side we shall denote $\mathcal{I}_C = (G, \mathcal{D}(C), \mathcal{C}(C), k_C)$. Let L and R denote the left- and right-hand side of formula (13), respectively.

We first show that $L \leq R$. If $L = -\infty$ then the inequality is obvious, so suppose that $\mathcal{Z} \subseteq \mathcal{D}$ is a normal subfamily of cardinality exactly k that maximizes the revenue on the left-hand side, i.e., $\mathbf{Val}[\mathcal{I}] = \mathbf{Val}[\mathcal{I}, \mathcal{Z}]$. By Theorem 4.22, there exists a guarded separator $\mathbb{X} \in \mathcal{N}$ that is $\frac{2}{3}$-interaction-balanced w.r.t. $\mathcal{Z}$. In particular $\mathbb{X}$ is compatible with $\mathcal{Z}$, so $\mathbf{ban}(\mathbb{X}) \cap \mathcal{Z} = \emptyset$.

For a connected component $C \in \mathrm{cc}(\mathbb{X})$, let $\mathcal{Z}_C = \mathcal{Z} \cap \mathcal{D}(C)$ and $k_C = |\mathcal{Z}_C|$; observe that $\mathcal{Z}_C$ is a normal subfamily of $\mathcal{D}(C)$. Since $\mathbf{ban}(\mathbb{X}) \cap \mathcal{Z} = \emptyset$, we infer that $\mathcal{Z}$ is a disjoint union of $\mathcal{Q}(\mathbb{X})$ and sets $\mathcal{Z}_C$ for $C \in \mathrm{cc}(\mathbb{X})$. Hence $\sum_{C \in \mathrm{cc}(\mathbb{X})} k_C = k - |\mathcal{Q}(\mathbb{X})|$. Moreover, since $\mathbb{X}$ is $\frac{2}{3}$-interaction-balanced w.r.t. $\mathcal{Z}$, we have that $k_C \leq \frac{2}{3}k$ for all $C \in \mathrm{cc}(\mathbb{X})$. This implies that vector $\mathbf{k} := (k_C)_{C\in\mathrm{cc}(\mathbb{X})}$ is compatible with $\mathbb{X}$.

Observe now that $\mathbf{Val}[\mathcal{I},\mathcal{Z}] = \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C\in\mathrm{cc}(\mathbb{X})} \mathbf{Val}[\mathcal{I}_C,\mathcal{Z}_C]$, where we denote $\mathcal{I}_C = (G, \mathcal{D}(C), \mathcal{C}(C), k_C)$. Indeed, the cost of every object of $\mathcal{Z}$ is counted once on both sides of the equality, and the same holds also for the prizes of clients covered by $\mathcal{Z}$: Prize of a client covered by an object of $\mathcal{Q}(\mathbb{X})$ is counted only in the term $\Pi(\mathcal{Q}(\mathbb{X}))$ due to removing these clients when constructing $\mathcal{L}(\mathbb{X})$, and every client not covered by $\mathcal{Q}(\mathbb{X})$ but covered by $\mathcal{Z}$ is counted only in the term $\mathbf{Val}[\mathcal{I}_C,\mathcal{Z}_C]$ for the component C it belongs to. Recall also that $\mathbf{Val}[\mathcal{I}] = \mathbf{Val}[\mathcal{I},\mathcal{Z}]$ and $\mathbf{Val}[\mathcal{I}_C] \geq \mathbf{Val}[\mathcal{I}_C,\mathcal{Z}_C]$ by the definition of $\mathbf{Val}[\mathcal{I}_C]$. Therefore, we have:

$$L = \mathbf{Val}[\mathcal{I}] = \mathbf{Val}[\mathcal{I},\mathcal{Z}] = \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C\in\mathrm{cc}(\mathbb{X})} \mathbf{Val}[\mathcal{I}_C,\mathcal{Z}_C] \leq \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C\in\mathrm{cc}(\mathbb{X})} \mathbf{Val}[\mathcal{I}_C] \leq R;$$

the last inequality follows from the fact that pair $(\mathbb{X},\mathbf{k})$ is considered in the maxima on the right-hand side of (13).

Now we prove that $L \geq R$. For this, it suffices to show that

$$\mathbf{Val}[\mathcal{I}] \geq \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C\in\mathrm{cc}(\mathbb{X})} \mathbf{Val}[\mathcal{I}_C] \tag{14}$$

for every guarded separator $\mathbb{X}$ and every vector $\mathbf{k} = (k_C)_{C\in\mathrm{cc}(\mathbb{X})}$ such that $\sum_{C\in\mathrm{cc}(\mathbb{X})} k_C = k - |\mathcal{Q}(\mathbb{X})|$; here, we denote $\mathcal{I}_C = (G, \mathcal{D}(C), \mathcal{C}(C), k_C)$. Hence, let us fix such pair $(\mathbb{X},\mathbf{k})$. For each $C \in \mathrm{cc}(\mathbb{X})$, let $\mathcal{Z}_C \subseteq \mathcal{D}(C)$ be a normal subfamily of cardinality k_C that maximizes the revenue in the instance $\mathcal{I}_C$; i.e., $\mathbf{Val}[\mathcal{I}_C,\mathcal{Z}_C] = \mathbf{Val}[\mathcal{I}_C]$. If such a family does not exist, then $\mathbf{Val}[\mathcal{I}_C] = -\infty$ and the claimed inequality is trivial.

Let us now construct $\mathcal{Z} := \mathcal{Q}(\mathbb{X}) \cup \bigcup_{C\in\mathrm{cc}(\mathbb{X})} \mathcal{Z}_C$. Since all the sets in this union are pairwise disjoint, we have that $|\mathcal{Z}| = k$.

We first verify that $\mathcal{Z}$ is normal. For the sake of contradiction, suppose that there are two objects $p, p' \in \mathcal{Z}$ that contradict the definition of normality. If $p, p' \in \mathcal{Q}(\mathbb{X})$, then $\mathcal{Q}(\mathbb{X})$ would not be normal, a contradiction with the definition of a guarded separator. If $p \in \mathcal{Q}(\mathbb{X})$ and $p' \notin \mathcal{Q}(\mathbb{X})$, then it would follow that $p' \in \mathbf{ban}(\mathbb{X})$, which means that p' could not have been included into any $\mathcal{Z}_C \subseteq \mathcal{D}(C)$, since these sets are disjoint with $\mathbf{ban}(\mathbb{X})$. If $p, p' \notin \mathcal{Q}(\mathbb{X})$ but $p, p' \in \mathcal{Z}_C$ for the same component C, then we would have a contradiction with the normality of family $\mathcal{Z}_C$. Finally, if $p \in \mathcal{Z}_C$ and $p' \in \mathcal{Z}_{C'}$ for different components $C, C' \in \mathrm{cc}(\mathbb{X})$, then we would have that $pp' \in E(\mathcal{L}(\mathbb{X}))$, a contradiction with C and C' being different connected components of $\mathcal{L}(\mathbb{X})$.

Now, we verify that

$$\mathbf{Val}[\mathcal{I},\mathcal{Z}] = \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C\in\mathrm{cc}(\mathbb{X})} \mathbf{Val}[\mathcal{I}_C,\mathcal{Z}_C]. \tag{15}$$

Firstly, for the objects' costs, since $\mathcal{Z} = \mathcal{Q}(\mathbb{X}) \cup \bigcup_{C\in\mathrm{cc}(\mathbb{X})} \mathcal{Z}_C$ we have that the cost of every object of $\mathcal{Z}$ is counted exactly once on each side of (15). Second, we check that the prize of every client $q \in \mathcal{C}$ covered by $\mathcal{Z}$ is counted exactly once in the right-hand side of (15); note that this formula does not count the prize of any client not covered by $\mathcal{Z}$. If q is covered by $\mathcal{Q}(\mathbb{X})$, then $\pi(q)$ is counted once in $\Pi(\mathcal{Q}(\mathbb{X}))$ and in none of the terms $\mathbf{Val}[\mathcal{I}_C,\mathcal{Z}_C]$, since q was removed when constructing the graph $\mathcal{L}(\mathbb{X})$. If now q is not covered by $\mathcal{Q}(\mathbb{X})$, then q belongs to some component $C \in \mathrm{cc}(\mathbb{X})$. Since q is covered by $\mathcal{Z}$, there exists some $p \in \mathcal{Z}$ that covers q; observe that by the definition of $\mathcal{L}(\mathbb{X})$, for each such p we have that $pq \in E(\mathcal{L}(\mathbb{X}))$, which implies that each such p must also belong to C. We infer that $\pi(q)$ is counted once in term $\mathbf{Val}[\mathcal{I}_C,\mathcal{Z}_C]$, and is counted zero times in every term $\mathbf{Val}[\mathcal{I}_{C'},\mathcal{Z}_{C'}]$ for $C' \neq C$. This concludes the proof of formula (15).

Now, since $\mathbf{Val}[\mathcal{I}] \geq \mathbf{Val}[\mathcal{I}, \mathcal{Z}]$ and $\mathbf{Val}[\mathcal{I}_C] = \mathbf{Val}[\mathcal{I}_C, \mathcal{Z}_C]$ for every $C \in \mathrm{cc}(\mathbb{X})$, by (15) we obtain that

$$\mathbf{Val}[\mathcal{I}] \geq \mathbf{Val}[\mathcal{I}, \mathcal{Z}] = \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C \in \mathrm{cc}(\mathbb{X})} \mathbf{Val}[\mathcal{I}_C, \mathcal{Z}_C] = \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C \in \mathrm{cc}(\mathbb{X})} \mathbf{Val}[\mathcal{I}_C],$$

which proves (14). Since $\mathbb{X}$ and $\mathbf{k}$ were chosen arbitrarily, we have thus proved that $L \geq R$. $\square$

Lemma 4.23 justifies the correctness of the following recursive algorithm for computing the value of $\mathbf{Val}[G, \mathcal{D}, \mathcal{C}, k]$, summarized as Algorithm `SolveDNC`. As the border case, if $k \leq 3$, then we compute the optimum revenue in a brute-force manner, by iterating through all the k-tuples of the objects. Otherwise, we aim at computing $\mathbf{Val}[\mathcal{I}]$ using formula (13). To this end, we run the algorithm of Theorem 4.22 to enumerate the family $\mathcal{N}$ of candidates for guarded separators separating evenly the solution. For each enumerated guarded separator $\mathbb{X} \in \mathcal{N}$ we need to compute the value $\max_{\mathbf{k} \in \mathrm{comp}(\mathbb{X})} \ \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C \in \mathrm{cc}(\mathbb{X})} \mathbf{Val}[G, \mathcal{D}(C), \mathcal{C}(C), k_C]$, since Lemma 4.23 asserts that then computing $\mathbf{Val}[G, \mathcal{D}, \mathcal{C}, k]$ will boil down to taking the maximum of these values. For a fixed guarded separator $\mathbb{X}$, we investigate all the connected components of $\mathrm{cc}(\mathbb{X})$, and for each $C \in \mathrm{cc}(\mathbb{X})$ and every integer ℓ with $0 \leq \ell \leq \frac{2}{3}k$ we compute the value $\mathbf{Val}[G, \mathcal{D}(C), \mathcal{C}(C), \ell]$ using a recursive call to Algorithm `SolveDNC`. Computing value $\max_{\mathbf{k} \in \mathrm{comp}(\mathbb{X})} \ \Pi(\mathcal{Q}(\mathbb{X})) + \sum_{C \in \mathrm{cc}(\mathbb{X})} \mathbf{Val}[G, \mathcal{D}(C), \mathcal{C}(C), k_C]$ based on all the relevant values $\mathbf{Val}[G, \mathcal{D}(C), \mathcal{C}(C), \ell]$ amounts to running a standard knapsack dynamic programming algorithm that keeps track of the optimum revenue for selecting ℓ objects from the first i connected components of $\mathrm{cc}(\mathbb{X})$, for all $\ell \in \{0, 1, \ldots, k\}$ and $i = 1, 2, \ldots, |\mathrm{cc}(\mathbb{X})|$.

Lemma 4.23 ensures that the algorithm correctly computes the value $\mathbf{Val}[\mathcal{I}]$, so we are left with estimating the running time.

Lemma 4.24. *Algorithm* `SolveDNC`*, applied to an instance $(G, \mathcal{D}, \mathcal{C}, k)$ with $|V(G)| = n$, $|\mathcal{D}| = d$, and $|\mathcal{C}| = c$, runs in time $d^{\mathcal{O}(\sqrt{k})} \cdot (cn)^{\mathcal{O}(1)}$ and space $(dcn)^{\mathcal{O}(1)}$.*

Proof. We shall say that a subcall $(G, \mathcal{D}', \mathcal{C}', k')$ to Algorithm `SolveDNC` is *trivial* if $\mathcal{D}' = \emptyset$. Note that trivial subcalls are resolved by `SolveDNC` in polynomial time since family $\mathcal{N}$ is empty.

Consider the application of `SolveDNC` to an instance $(G, \mathcal{D}, \mathcal{C}, k)$ with $k \geq 4$, and let us count the number of subcalls to `SolveDNC` it invokes. We have that $|\mathcal{N}| \leq (2d)^{15\sqrt{k}}$, and for every $\mathbb{X} \in \mathcal{N}$ we call `SolveDNC` on $|\mathrm{cc}(\mathbb{X})| \cdot (\lfloor \frac{2}{3}k \rfloor + 1)$ subinstances. At most d components of $\mathrm{cc}(\mathbb{X})$ contain an object of $\mathcal{D}$, which means that at most $d \cdot (\lfloor \frac{2}{3}k \rfloor + 1) \leq dk$ of these calls are non-trivial. In total, we have at most $dk \cdot (2d)^{15\sqrt{k}} \leq (2d)^{17\sqrt{k}}$ nontrivial subcalls.

Let T^* be the tree of nontrivial recursive subcalls of the application of `SolveDNC` to instance $(G, \mathcal{D}, \mathcal{C}, k)$. Observe that T^* has depth $\mathcal{O}(\log k)$, since for each call of `SolveDNC`, in all the recursive subcalls the target size of the sought family is at most a $\frac{2}{3}$-fraction of the original one. For $0 \leq \ell \leq k$, let $\phi(\ell)$ be the maximum number of leaves in the subtrees of T^* rooted at calls where the target size of the sought family is at most ℓ (or 1, if there are no such calls). Since in the recursive calls both the target family size and the total number of objects can only decrease, we have that $\phi(\ell)$ satisfies that:

$$\phi(\ell) \leq (2d)^{17\sqrt{\ell}} \cdot \phi(\lceil 2\ell/3 \rceil) \qquad \text{for } 4 \leq \ell \leq k. \tag{16}$$

Moreover, since for $\ell \leq 3$ Algorithm `SolveDNC` solves the instance by brute-force, we have that

$$\phi(\ell) = 1 \qquad \text{for } 0 \leq \ell \leq 3. \tag{17}$$

Claim 4.25. *For every function $\phi : \mathbb{N} \to \mathbb{N}$ that satisfies (16) and (17), the following holds:*

$$\phi(\ell) \leq (2d)^{17(3+\sqrt{6})\sqrt{\ell}} \qquad \textit{for } 0 \leq \ell \leq k. \tag{18}$$

Input: An instance $\mathcal{I} = (G, \mathcal{D}, \mathcal{C}, k)$
Output: Value $\mathbf{Val}[\mathcal{I}]$

```
if k ≤ 3 then
    Compute the optimum revenue ret by iterating through all the k-tuples of objects
    return ret
end
ret ← −∞
for 𝕏 ∈ 𝒩, enumerated using the algorithm of Theorem 4.22 do
    Compute Π(𝒬(𝕏))
    Compute sets ban(𝕏) and cov(𝕏)
    Compute ℒ(𝕏) = ℒ − (𝒬(𝕏) ∪ ban(𝕏) ∪ cov(𝕏))
    C_1, C_2, ..., C_p ← Connected components of ℒ(𝕏)
    for i = 1 to p do
        for ℓ = 0 to ⌊(2/3)k⌋ do
            A[i][ℓ] ← SolveDNC(G, 𝒟(C_i), 𝒞(C_i), ℓ)
        end
    end
    D[0][0] ← 0
    for ℓ = 1 to k do
        D[0][ℓ] ← −∞
    end
    for i = 1 to p do
        for ℓ = 0 to k do
            D[i][ℓ] ← −∞
            for ℓ' = 0 to min(ℓ, ⌊(2/3)k⌋) do
                D[i][ℓ] ← max(D[i][ℓ], A[i][ℓ'] + D[i − 1][ℓ − ℓ'])
            end
        end
    end
    ret ← max(ret, Π(𝒬(𝕏)) + D[p][k − |𝒬(𝕏)|])
end
return ret
```

Algorithm 1: Algorithm `SolveDNC`

Proof. We prove the claim by induction. The base for $\ell \leq 3$ is trivial by (17), so let us show the inductive step for $\ell \geq 4$:

$$\begin{aligned}
\phi(\ell) &\leq (2d)^{17\sqrt{\ell}} \cdot \phi(\lceil 2\ell/3 \rceil) \\
&\leq (2d)^{17\sqrt{\ell}} \cdot (2d)^{17(3+\sqrt{6})\sqrt{\frac{2}{3}}\sqrt{\ell}} \\
&= (2d)^{17(1+\sqrt{6}+2)\sqrt{\ell}} = (2d)^{17(3+\sqrt{6})\sqrt{\ell}};
\end{aligned}$$

Here, in the first inequality we used (16) and in the second we used the induction hypothesis. ⌟

Claim 4.25 implies that the subcall tree T^* has at most $(2d)^{17(3+\sqrt{6})\sqrt{k}} < (2d)^{93\sqrt{k}}$ leaves. Since the depth of T^* is $\mathcal{O}(\log k)$, we infer that T^* has $\mathcal{O}(\log k \cdot (2d)^{93\sqrt{k}})$ nodes. Now observe that for each call of `SolveDNC`, the amount of work used in the main body of this algorithm (excluding the recursive subcalls) is equal to the maximum possible number of guarded separators considered at this step, i.e., $(2d)^{15\sqrt{k}}$, times a factor polynomial in n, d, and c; here, we assume that the time used for resolving trivial subcalls is charged to the parent call. Hence, the total running time used by the algorithm is at most $(2d)^{108\sqrt{k}} \cdot (dcn)^{\mathcal{O}(1)} \leq d^{\mathcal{O}(\sqrt{k})} \cdot (cn)^{\mathcal{O}(1)}$, as requested. To argue that the algorithm runs in polynomial space, observe that at each moment the algorithm keeps track of the data stored for $\mathcal{O}(\log k)$ recursive calls of `SolveDNC`, and, due to the algorithm of Theorem 4.22 using only polynomial working space, each call uses only polynomial space for its internal data. ☐

Thus, Lemmas 4.23 and 4.24 conclude the proof of Theorem 3.1.

5 Hardness results

In this section, we prove the lower bounds in Theorems 1.8–1.11 suggesting that for many natural covering problems the $n^{O(k)}$ time brute force algorithms are almost optimal. Let us start with a very simple reduction showing such a lower bound in the setting of planar graphs. Pătraşcu and Williams gave a very tight lower bound for Dominating Set, assuming the Strong Exponential Time Hypothesis (SETH).

Theorem 5.1 ([44]). *Assuming SETH, there is no $f(k) \cdot n^{k-\epsilon}$ time algorithm for* Dominating Set*, where n is the number of vertices of the input graph G.*

Using this result, the proof of Theorem 1.8 is very simple and transparent.

Restated Theorem 1.8 (covering vertices with connected sets, lower bound). *Let G be a planar graph and let $\mathcal{D}$ be a set of connected vertex sets of G. Assuming SETH, there is no $f(k) \cdot (|\mathcal{D}| + |V(G)|)^{k-\epsilon}$ time algorithm for any computable function f and any $\epsilon > 0$ that decides if there are k sets in $\mathcal{D}$ whose union covers $|V(G)|$.*

Proof. Let H be an arbitrary graph with n vertices $v_1, \ldots, v_n$. Let G be a star with leaves $v_1, \ldots, v_n$ and center x. For every vertex $v_i \in V(H)$, let us introduce a set S_i into $\mathcal{D}$ that contains v_i, every neighbor of v_i in H, and the center vertex x of G; clearly, the set S_i is connected in G. It is easy to see that k vertices of H form a dominating set of G if and only if the corresponding k sets of $\mathcal{D}$ cover every vertex of G. Therefore, an $f(k) \cdot (|\mathcal{D}| + |V(G)|)^{k-\epsilon}$ time algorithm for the covering problem on G would give an $f(k)n^{k-\epsilon}$ time algorithm for solving Dominating Set on H; this contradicts SETH by Theorem 5.1. ☐

In Sections 5.1–5.3, we give similar lower bounds for geometric objects, proving Theorems 1.9–1.11.

5.1 Convex polygons

The proof of Theorem 1.9 is again a simple reduction from Dominating Set. We observe that if a family of points lies on a convex curve, then any subset of them is in convex position, hence convex polygons can cover arbitrary subsets of these points. This simple idea was also used, for example, by Har-Peled [27], but here we use it to obtain a different conclusion (tight lower bound on the exponent instead of APX-hardness).

Restated Theorem 1.9 (covering points with convex polygons, lower bound). *Let $\mathcal{D}$ be a set of convex polygons and let $\mathcal{P}$ be a set of points in the plane. Assuming SETH, there is no $f(k) \cdot (|\mathcal{D}| + |\mathcal{P}|)^{k-\epsilon}$ time algorithm for any computable function f and $\epsilon > 0$ that decides if there are k polygons in $\mathcal{D}$ that together cover $\mathcal{P}$.*

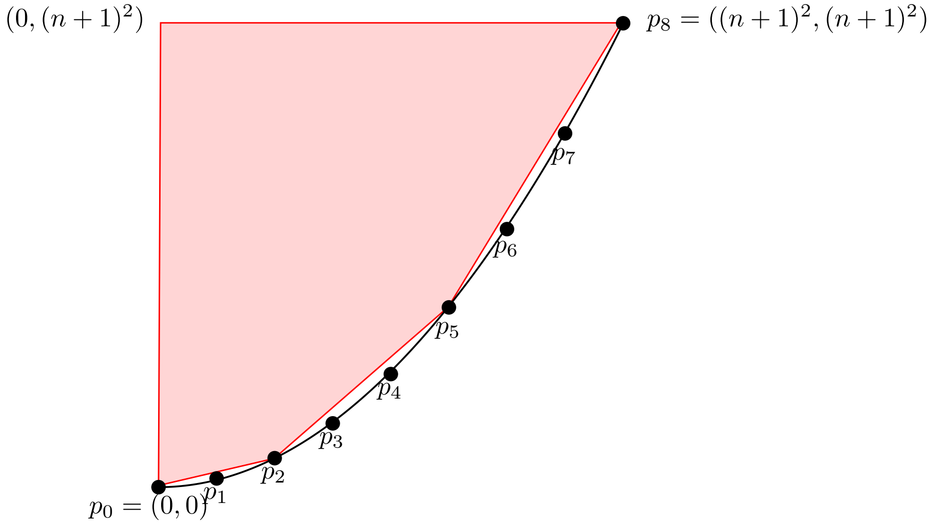


Figure 13: Proof of Theorem 1.9. The points on a parabola and the convex polygon corresponding to a vertex whose closed neighborhood is $\{v_2, v_5\}$.

Proof. Given an instance (G,k) of Dominating Set with $n = |V(G)|$, we construct a set $\mathcal{D}$ of n convex polygons and a set $\mathcal{P}$ of $n+2$ points such that G has a dominating set of size k if and only if $\mathcal{P}$ can be fully covered by k convex polygons from $\mathcal{D}$. Therefore, an $f(k) \cdot n^{k-\epsilon}$ time algorithm for the problem would imply an algorithm for Dominating Set with similar running time, which would contradict SETH by Theorem 5.1.

We define the point set $\mathcal{P} = \{p_i = (i(n+1), i^2) \mid 0 \le i \le n+1\}$, which is a set of $n+1$ points on a parabolic curve (see Figure 13). Let $v_1, \dots, v_n$ be the vertices of G. For every $1 \le i \le n$, we introduce into $\mathcal{D}$ a convex polygon corresponding to v_i. Suppose that the closed neighborhood of v_i is $v_{j_1}, \dots, v_{j_d}$ for some $1 \le j_1 < j_2 < \dots < j_d \le n$. Then v_i is represented by a convex polygon P_i with vertices $(0,(n+1)^2)$, p_0, p_{j_1}, p_{j_2}, $\dots$, p_{j_r}, p_{n+1}. As the parabola is a convex curve, this polygon P_i is convex and it covers exactly the points p_0, p_{j_1}, p_{j_2}, $\dots$, p_{j_r}, p_{n+1}. That is, if v_j is not in the closed neighborhood of v_i, then p_j is outside P_j. It is easy to see that a subset of $\mathcal{D}$ covers $\mathcal{P}$ if and only if the union of the closed neighborhoods of the corresponding vertices of G cover $V(G)$, or in other words, the corresponding vertices of G form a dominating set. $\square$

5.2 Thin rectangles

In the proof of Theorem 1.10, we are reducing from the Partitioned Biclique problem: given a graph G with a partition $(A_1, \dots, A_k, B_1, \dots, B_k)$ of the vertices, the task is to find $2k$ vertices $a_1 \in A_1, \dots, a_k \in A_k$, $b_1 \in B_1, \dots, b_k \in B_k$ such that a_i and b_j are adjacent for every $1 \le i, j \le k$. There is a very simple reduction from Clique to Partitioned Biclique (see, e.g., [10]) where the output parameter equals the input one, hence the lower bound of Chen et al. [8] can be transferred to Partitioned Biclique.

Theorem 5.2. *Assuming ETH,* Partitioned Biclique *cannot be solved in time $f(k)n^{o(k)}$ for any computable function f.*

The proof of Theorem 1.10 is a parameterized reduction from Partitioned Biclique.

Restated Theorem 1.10 (covering points with thin rectangles, lower bound). *Consider the problem of covering a set $\mathcal{P}$ of points by selecting k axis-parallel rectangles from a set $\mathcal{D}$.*

Assuming ETH, there is no algorithm for this problem with running time $f(k) \cdot (|\mathcal{P}| + |\mathcal{D}|)^{o(k)}$ for any computable function f, even if each rectangle in $\mathcal{D}$ is of size $1 \times k$ or $k \times 1$.

Proof. We prove the theorem by a reduction from Partitioned Biclique. Let G be a graph with a partition $(A_1, \dots, A_k, B_1, \dots, B_k)$ of the vertices. Without loss of generality, we assume that every class of the partition has the same size n. For $1 \le i \le k$, we denote the vertices of A_i and B_i by $a_{i,j}$ and $b_{i,j}$ $(1 \le j \le n)$, respectively.

Construction. We construct a set $\mathcal{R}$ of axis-parallel rectangles and a set $\mathcal{P}$ of points such that there are $k' := 8k$ rectangles in $\mathcal{R}$ covering $\mathcal{P}$ if and only if G contains the required partitioned biclique. For simplicity of notation, we construct open rectangles, that is, selecting a rectangle does not cover the points on its boundary. However, it is easy to modify the reduction (by decreasing the size of each rectangle slightly) so that it works for closed rectangles as well.

First, for every $1 \le i \le k$, we define the rectangles (see Figure 14)

$$\begin{aligned} V_i^T &= (4i - 0.5, 4i + 0.5) \times (4(k+1), 8(k+1)) \\ V_i^B &= (4i - 0.5, 4i + 0.5) \times (-4(k+1), 0) \\ V_i^L &= (4i - 1, 4i) \times (0, 4(k+1)) \\ V_i^R &= (4i, 4i + 1) \times (0, 4(k+1)) \end{aligned}$$

and for every $1 \le j \le k$, we define the rectangles

$$\begin{aligned} H_j^T &= (0, 4(k+1)) \times (4j, 4j + 1) \\ H_j^B &= (0, 4(k+1)) \times (4j - 1, 4j) \\ H_j^L &= (-4(k+1), 0) \times (4j - 0.5, 4j + 0.5) \\ H_j^R &= (4(k+1), 8(k+1)) \times (4j - 0.5, 4j + 0.5) \end{aligned}$$

These rectangles themselves do not appear in the set $\mathcal{R}$, but they will used be for the definition of rectangles in $\mathcal{R}$, as follows. For a rectangle R, we use the notation $R + (x, y)$ to denote the rectangle obtained from R by shifting it horizontally by x and vertically by y. We use the notation $\nearrow$, $\searrow$ etc. for the diagonal vectors $(1, 1)$, $(1, -1)$ etc. with unit projections; then, e.g., $\lambda \cdot \searrow$ denotes the vector $(\lambda, -\lambda)$.

Let $\epsilon = 1/(100n)$. For every $1 \le i \le k$ and $1 \le \alpha \le n$, we introduce the following rectangles into $\mathcal{R}$ (see the directions shown by the arrows in Figure 14):

$$\begin{aligned} V_{i,\alpha}^T &= V_i^T + \alpha\epsilon \cdot \searrow \\ V_{i,\alpha}^B &= V_i^B + \alpha\epsilon \cdot \nwarrow \\ V_{i,\alpha}^L &= V_i^L + \alpha\epsilon \cdot \nearrow \\ V_{i,\alpha}^R &= V_i^R + \alpha\epsilon \cdot \searrow \end{aligned}$$

Furthermore, for every $1 \le j \le k$ and $1 \le \beta \le n$, we introduce the following rectangles:

$$\begin{aligned} H_{j,\beta}^T &= H_j^T + \beta\epsilon \cdot \nearrow \\ H_{j,\beta}^B &= H_j^B + \beta\epsilon \cdot \nwarrow \\ H_{j,\beta}^L &= H_j^L + \beta\epsilon \cdot \nearrow \\ H_{j,\beta}^R &= H_j^R + \beta\epsilon \cdot \swarrow \end{aligned}$$

This completes the description of $\mathcal{R}$; note that $|\mathcal{R}| = 8kn$. To define $\mathcal{P}$, for every $1 \le i \le k$ and $1 \le \alpha \le n$, we introduce the points (see again the directions shown by arrows in Figure 14)

$$v_{i,\alpha}^{TL} = (4i - 0.5, 4(k+1)) + (\alpha + 0.5)\epsilon \cdot \nearrow \qquad v_{i,\alpha}^{TR} = (4i + 0.5, 4(k+1)) + (\alpha + 0.5)\epsilon \cdot \searrow$$
$$v_{i,\alpha}^{BL} = (4i - 0.5, 0) + (\alpha + 0.5)\epsilon \cdot \nwarrow \qquad v_{i,\alpha}^{BR} = (4i + 0.5, 4(k+1)) + (\alpha + 0.5)\epsilon \cdot \swarrow$$

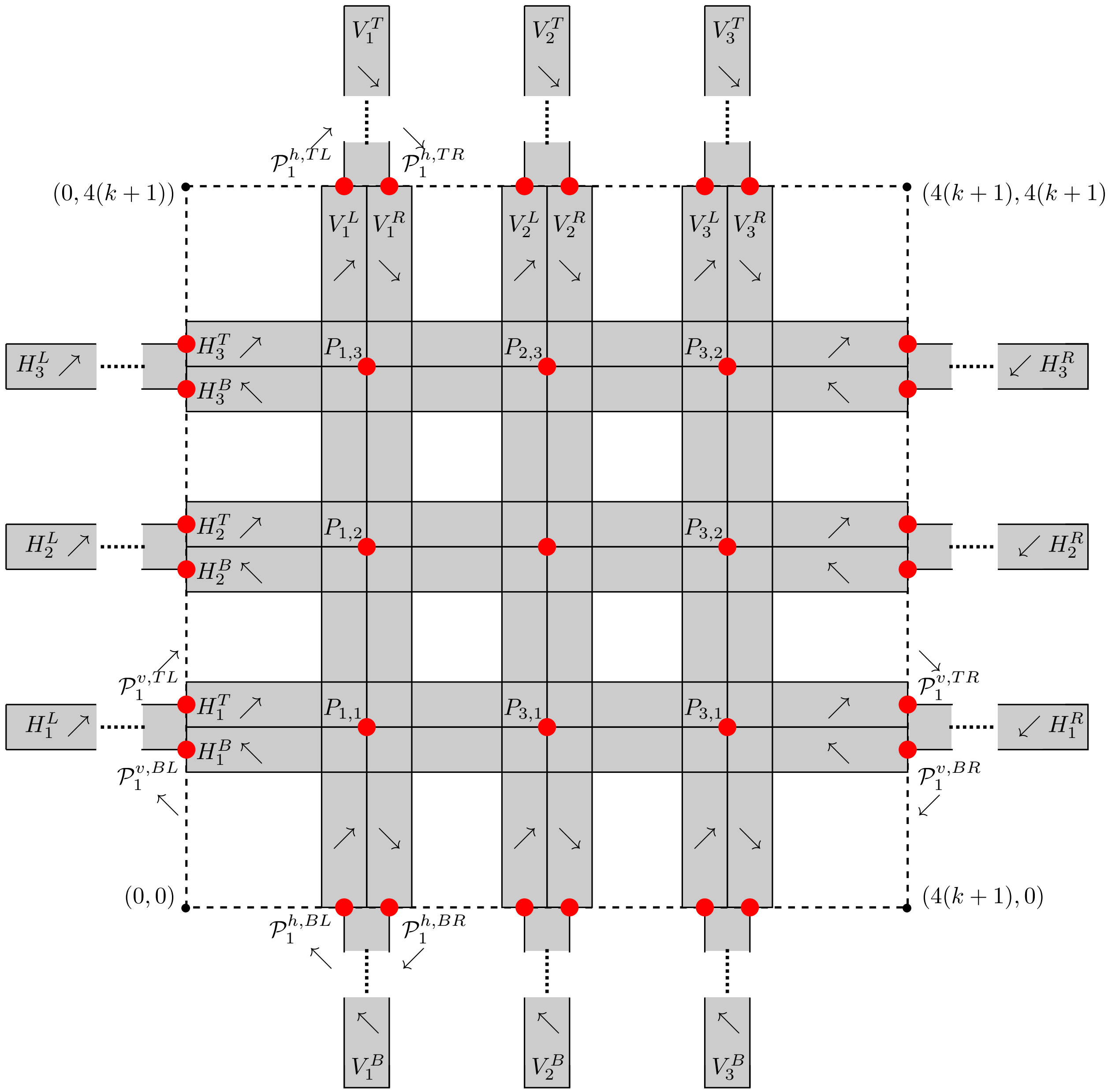


Figure 14: Proof of Theorem 1.10. The rectangles on the left and the right are shortened to save space. The directions of the arrows on the rectangles show the offsets of the rectangles $V_{i,\alpha}^T$ etc. as α and β grow. Similarly, the arrows at $\mathcal{P}_i^{h,TL}$ etc. show the offset of the vertices h_i^{TL}.

and for every $1 \le j \le k$ and $1 \le \beta \le n$, we introduce the points

$$\begin{array}{ll} h^{TL}_{j,\beta} = (0, 4j + 0.5) + (\beta + 0.5)\epsilon \cdot \nearrow & h^{TR}_{j,\beta} = (4(k+1), 4j + 0.5) + (\beta + 0.5\epsilon) \cdot \searrow \\ h^{BL}_{j,\beta} = (0, 4j - 0.5) + (\beta + 0.5)\epsilon \cdot \nwarrow & h^{BR}_{j,\beta} = (4(k+1), 4j - 0.5) + (\beta + 0.5)\epsilon \cdot \swarrow \end{array}$$

We will use the notation $\mathcal{P}^{v,TL}_i := \{v^{TL}_{i,\alpha} : 1 \le \alpha \le n\}$ etc. for these sets of points. Notice that these points are not on the boundary of any of the rectangles (because of the terms 0.5ϵ), thus it does not matter for the covering of these points that the rectangles are open.

Finally, for every $1 \le i, j \le k$, we add a set $P_{i,j}$ of points to $\mathcal{P}$ the following way: for every $1 \le \alpha, \beta \le n$, the point $(4i + \alpha\epsilon, 4j + \beta\epsilon)$ is in $P_{i,j}$ if $a_{i,\alpha}$ and $b_{j,\beta}$ are *not* adjacent in G. This completes the description of construction, we have

$$\mathcal{P} = \bigcup_{i=1}^{k} (\mathcal{P}^{v,TL}_i \cup \mathcal{P}^{v,TR}_i \cup \mathcal{P}^{v,BL}_i \cup \mathcal{P}^{v,BR}_i) \cup \bigcup_{j=1}^{k} (\mathcal{P}^{h,TL}_j \cup \mathcal{P}^{h,TR}_j \cup \mathcal{P}^{h,BL}_h \cup \mathcal{P}^{h,BR}_j) \cup \bigcup_{1 \le i,j \le k} P_{i,j}.$$

Note that $|\mathcal{P}| \le 8kn + k^2n^2$.

We claim that we can cover the points in $\mathcal{P}$ with $8k$ rectangles of $\mathcal{R}$ if and only if the Partitioned Biclique instance has a solution. The intuition behind the construction is the following. A solution of size $8k$ has to select exactly one rectangle of each of the $8k$ types $V^T_{1,\alpha}$, $V^B_{1,\alpha}$ etc. The fact that the point set $\mathcal{P}^{v,TL}_i \cup \mathcal{P}^{v,TR}_i \cup \mathcal{P}^{v,BL}_i \cup \mathcal{P}^{v,BR}_i$ has to be covered by the four selected rectangles of the form V^T_{i,α_T}, V^B_{i,α_B}, V^L_{i,α_L}, V^R_{i,α_R} ensures that the selection of these rectangles are synchronized, that is, we have to select V^T_{i,α_i}, V^B_{i,α_i}, V^L_{i,α_i}, V^R_{i,α_i} for some $1 \le \alpha_i \le n$. Similarly, for every $1 \le j \le k$, we have to select H^T_{j,β_j}, H^B_{j,β_j}, H^L_{j,β_j}, H^R_{j,β_j} for some $1 \le \beta_j \le n$. The role of the point set $P_{i,j}$ is to ensure that vertices a_{i,α_i} and b_{j,β_j} are adjacent in G. Indeed, if they are not adjacent, then the point $(4i + \alpha\epsilon, 4j + \beta\epsilon)$ is in $P_{i,j}$, but it is not covered by any rectangle: it is on the boundary of each of V^L_{i,α_i}, V^R_{i,α_i}, H^T_{j,β_j}, H^B_{j,β_j}.

For the formal proof of the equivalence, let us start with a useful preliminary observation.

Claim 5.3. *For every $1 \le j \le k$ and $1 \le \alpha_T, \alpha_B, \alpha_L, \alpha_R \le n$, the rectangles V^T_{i,α_T}, V^B_{i,α_B}, V^L_{i,α_L}, V^R_{i,α_R} cover all the points in $\mathcal{P}^{v,TL}_i \cup \mathcal{P}^{v,TR}_i \cup \mathcal{P}^{v,BL}_i \cup \mathcal{P}^{v,BR}_i$ if and only if $\alpha_T = \alpha_B = \alpha_L = \alpha_R$.*

Proof. Assume first that $\alpha_T = \alpha_B = \alpha_L = \alpha_R = \alpha^*$. Consider a point $v^{TL}_{i,\alpha}$. If $\alpha < \alpha^*$, then rectangle V^L_{i,α^*} covers this point: the vertical coordinate of this point is $4(k+1) + \alpha\epsilon + \epsilon/2$ and the top boundary of V^L_{i,α^*} is at $4(k+1) + \alpha^*\epsilon$. If $\alpha \ge \alpha^*$, then rectangle V^T_{i,α^*} covers the point: the horizontal coordinate of this point is $4i - 0.5 + (\alpha + 0.5)\epsilon$ and the left boundary of V^T_{i,α^*} is at $4i - 0.5 + \alpha^*\epsilon$. In a similar way, we can verify that the four rectangles cover $v^{TR}_{i,\alpha}$, $v^{BL}_{i,\alpha}$, $v^{BR}_{i,\alpha}$ for every $1 \le \alpha \le n$.

Suppose now that the four rectangles cover all the $4n$ points. If $\alpha_L < \alpha_T$, then point v^{TL}_{i,α_L} is not covered: rectangle V^L_{i,α_L} covers only the points $v^{TL}_{1,1}, \ldots, v^{TL}_{1,\alpha_L - 1}$ of $\mathcal{P}^{v,TL}_i$, while rectangle V^L_{i,α_T} covers only the points $v^{TL}_{i,\alpha_T}, \ldots, v^{TL}_{i,n}$ of $\mathcal{P}^{v,TL}_i$. Therefore, we have $\alpha_L \ge \alpha_T$. In a similar way, we can prove $\alpha_T \ge \alpha_R \ge \alpha_B \ge \alpha_L \ge \alpha_T$, implying that all these values are equal. ⌟

We can prove an analogous statement for the vertical rectangles.

Claim 5.4. *For every $1 \le j \le k$ and $1 \le \beta_T, \beta_B, \beta_L, \beta_R \le n$, the rectangles H^T_{j,β_T}, H^B_{j,β_B}, H^L_{j,β_L}, H^R_{j,β_R} cover all the points in $\mathcal{P}^{h,TL}_j \cup \mathcal{P}^{h,TR}_j \cup \mathcal{P}^{h,BL}_j \cup \mathcal{P}^{h,BR}_j$ if and only if $\beta_T = \beta_B = \beta_L = \beta_R$.*

Biclique $\Rightarrow$ covering rectangles. Suppose that vertices $a_{1,\alpha_1}, \ldots, a_{k,\alpha_k}, b_{1,\beta_1}, \ldots, b_{k,\beta_k}$ form a biclique in G. We claim that the $8k$ rectangles

- V^T_{i,α_i}, V^B_{i,α_i}, V^L_{i,α_i}, V^R_{i,α_i} for $1 \le i \le k$, and
- H^T_{j,β_j}, H^B_{j,β_j}, H^L_{j,β_j}, H^R_{j,β_j} for $1 \le j \le k$

cover every point in $\mathcal{P}$. By Claim 5.3, for every $1 \le i \le k$, the rectangles V^T_{i,α_i}, V^B_{i,α_i}, V^L_{i,α_i}, V^R_{i,α_i} cover every point in $\mathcal{P}^{v,TL}_i \cup \mathcal{P}^{v,TR}_i \cup \mathcal{P}^{v,BL}_i \cup \mathcal{P}^{v,BR}_i$ and by Claim 5.4, for every $1 \le j \le k$, the rectangles H^T_{j,β_j}, H^B_{j,β_j}, H^L_{j,β_j}, H^R_{i,β_j} cover every point in $\mathcal{P}^{h,TL}_j \cup \mathcal{P}^{h,TR}_j \cup \mathcal{P}^{h,BL}_h \cup \mathcal{P}^{h,BR}_j$. Consider now a point $p = (4i + \alpha\epsilon, 4j + \beta\epsilon) \in P_{i,j}$. If $\alpha < \alpha_i$, then V^L_{i,α_i} covers p; if $\alpha > \alpha_i$, then V^R_{i,α_i} covers p. If $\beta < \beta_j$, then H^B_{j,β_j} covers p; if $\beta > \beta_j$, then V^T_{j,β_j} covers p. Therefore, we have a problem only if $\alpha = \alpha_i$ and $\beta = \beta_j$. However, we know that vertices a_{i,α_i} and b_{j,β_j} are adjacent in G, which means that the point $p = (4i + \alpha\epsilon, 4j + \beta\epsilon)$ was *not* added to $P_{i,j}$. Therefore, the selected $8k$ rectangles indeed cover every point in $\mathcal{P}$.

Covering rectangles $\Rightarrow$ biclique. For the proof of the reverse direction, suppose that $\mathcal{P}$ was covered by $8k$ rectangles from $\mathcal{R}$. For every $1 \le i \le k$, at least one rectangle of the from $V^T_{i,\alpha}$ has to be selected: no rectangle of some other form can cover the point $v^{TL}_{i,n}$ with vertical coordinate $4(k+1) + n\epsilon + \epsilon/2$. Similarly, at least one rectangle of each of the forms $V^B_{i,\alpha}$, $V^L_{i,\alpha}$, $V^R_{i,\alpha}$ has to be selected. An analogous statement holds for every $1 \le j \le k$ and rectangles of the form $H^T_{j,\beta}$, $H^B_{j,\beta}$, $H^L_{j,\beta}$, $H^R_{j,\beta}$. As exactly $8k$ rectangles were selected, exactly one rectangle has to be selected for each type. Consider some $1 \le i \le k$ and suppose that V^T_{i,α_T}, V^B_{i,α_B}, V^L_{i,α_L}, V^R_{i,α_R} were selected. As these rectangles have to cover $\mathcal{P}^{v,TL}_i \cup \mathcal{P}^{v,TR}_i \cup \mathcal{P}^{v,BL}_i \cup \mathcal{P}^{v,BR}_i$, Claim 5.3 implies that $\alpha_T = \alpha_B = \alpha_L = \alpha_R$; let us define α_i to be this number. Similarly, using Claim 5.4, we can show that for every $1 \le j \le k$, there is a $1 \le \beta_j \le n$ such that the rectangles H^T_{j,β_j}, H^B_{j,β_j}, H^L_{j,β_j}, H^R_{j,β_j} were selected.

We claim that $a_{1,\alpha_1}, \dots, a_{k,\alpha_k}, b_{1,\beta_1}, \dots, b_{k,\beta_k}$ form a solution to the Partitioned Biclique problem. Suppose for a contradiction that a_{i,α_i} and b_{j,α_j} are not adjacent in G. Then the point $(4i + \alpha_i\epsilon, 4j + \beta_j\epsilon)$ is in $P_{i,j}$. We arrive to a contradiction by showing that this point is not covered by any of the selected rectangles. Indeed, this point is on the boundary of each of V^L_{i,α_i}, V^R_{i,α_i}, H^T_{j,β_j}, H^B_{j,β_j} and recall that all the rectangles are open. Therefore, we have shown that the Partitioned Biclique instance has a solution. □

5.3 Almost squares

The goal of this section is to prove Theorem 1.11, which is another lower bound for covering points with axis-parallel rectangles. In Theorem 1.10, we had only two types of "thin" rectangles. In this section, we consider sets of rectangles of many different sizes, but they are all "almost unit squares": every rectangle has width and height in the range $[1 - \epsilon_0, 1 + \epsilon_0]$ for some fixed $\epsilon_0 > 0$.

Restricting the problem to almost unit squares makes the hardness proof significantly more challenging. Let us observe that the proof of Theorem 1.10 crucially relied on the fact that the selection of the horizontal rectangles $H^T_{j,\beta}$, $H^B_{j,\beta}$ interacted with the selection of the vertical rectangles $V^L_{i,\alpha}$, $V^R_{i,\alpha}$ for every $1 \le i, j \le k$. Intuitively speaking, the selection of these $O(k)$ rectangles were constrained by $O(k^2)$ interactions. The reason why such a large number of interactions could be reached is because a wide horizontal rectangle could be intersected by $O(k)$ independent thin vertical rectangles. However, if every rectangle is almost a unit square, then this is no longer possible. After trying some possible configurations that may be useful for a hardness proof, one gets the impression that an almost unit square can have independent interactions only with $O(1)$ other almost unit squares. This means that we can have a total of $O(k)$ interactions, which does not seem to be sufficient to express a problem such as Clique or Partitioned Biclique, as these problems have a richer interaction structure with $O(k^2)$ interactions. Therefore, it seems difficult to reduce from these problems with only a linear blowup of the parameter, which is necessary for the lower bounds that we want to prove.

To get around these limitations, we are reducing from a problem where the interaction structure is sparser and a lower bound ruling out $f(k)n^{o(k)}$ (or so) algorithms is known even with only $O(k)$ constraints. The Partitioned Subgraph Isomorphism problem is similar to Partitioned Biclique, but now the graph we are looking for can be arbitrary, it is not necessarily a biclique. The input consists of a graph H with vertex set $\{u_1, \ldots, u_k\}$ and a graph G whose vertex set is partitioned into k classes $V_1, \ldots, V_k$. The task is to find a mapping $\mu : V(H) \to V(G)$ such that $\mu(u_i) \in V_i$ for every $1 \le i \le k$ and μ is a subgraph embedding, that is, if u_i and u_j are adjacent in H, then $\mu(u_i)$ and $\mu(u_j)$ are adjacent in G. This problem contains Partitioned Biclique as a special case, thus Theorem 5.2 implies that it cannot be solved in time $f(k)n^{o(k)}$ for any computable function f. However, if we express Partitioned Biclique with Partitioned Subgraph Isomorphism, then H has $O(k^2)$ edges. Interestingly, the lower bound remains valid (up to a log factor) even if we restrict H to be sparse.

Theorem 5.5 ([39]). *Assuming ETH,* Partitioned Subgraph Isomorphism *cannot be solved in time $f(k)n^{o(k/\log k)}$ (where $k = |V(H)|$) for any computable function f, even when H is restricted to be a 3-regular bipartite graph.*

We prove a lower bound first for an intermediary problem, covering points by 2-track intervals, and then obtain Theorem 1.11 using a simple reduction of Chan and Grant [7]. Many natural optimization problems on intervals (such as finding a maximum set of disjoint intervals or covering points on the line by the minimum number of intervals) are polynomial-time solvable. One can obtain generalizations of these problems by considering *c-intervals* instead of intervals, that is, the objects given in the input are unions of c intervals and we have to select k such objects subject to some disjointness or covering condition [6, 20, 21, 25, 31, 32]. These problems are typically much harder than the analogous problems on intervals. A special case of a c-interval is a *c-track interval:* we imagine the problem to be defined on c independent lines (tracks) and each c-track interval consists of an interval on each of the tracks. Many of the hardness results for c-intervals remain valid also for c-track intervals.

Let us formally define the intermediary problem and prove the lower bound we need. A *2-track interval* is an ordered pair (I^1, I^2) of intervals on the real line; for concreteness the reader may assume that the intervals are closed, but this does not have any influence on the validity of the claims to follow. We say that 2-track interval (I^1, I^2) *covers a point x on the first (resp., second) track* if $x \in I^1$ (resp., $x \in I^2$). A *2-track point set* $\mathcal{P}$ is an ordered pair $(\mathcal{P}^1, \mathcal{P}^2)$ of two sets of real numbers. We say that that a set $\mathcal{I}$ of 2-track intervals covers a 2-track point set $(\mathcal{P}^1, \mathcal{P}^2)$ if

- for every $x \in \mathcal{P}^1$, there is a 2-track interval $(I^1, I^2) \in \mathcal{I}$ that covers x on the first track (i.e., $x \in I^1$) and
- for every $x \in \mathcal{P}^2$, there is a 2-track interval $(I^1, I^2) \in \mathcal{I}$ that covers x on the second track (i.e., $x \in I^2$).

We prove a lower bound for the problem of covering a 2-track point set by selecting k 2-track intervals. Note that this problem is related to, but not the same as, the Dominating Set problem for 2-track interval graphs, for which W[1]-hardness results are known [20, 32] (however no ETH-based lower bound was stated).

Theorem 5.6. *Consider the problem of covering a 2-track point set $\mathcal{P} = (\mathcal{P}^1, \mathcal{P}^2)$ by selecting k 2-track intervals from a set $\mathcal{I}$. Assuming ETH, there is no algorithm for this problem with running time $f(k) \cdot (|\mathcal{P}| + |\mathcal{I}|)^{o(k/\log k)}$ for any computable function f.*

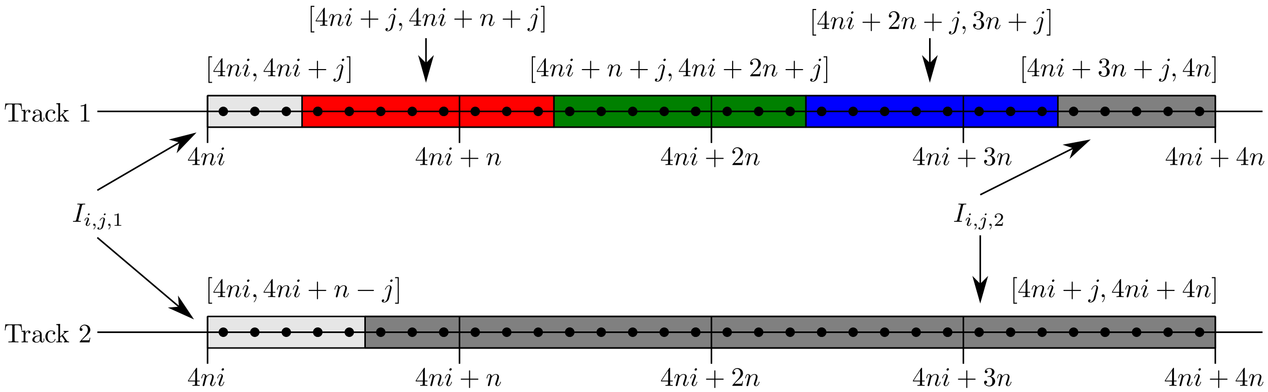


Figure 15: Proof of Theorem 5.6. Let $n = 8$ and $j = 3$. The figure shows the 2-track intervals $I_{i,j,1}$ (light gray) and $I_{i,j,2}$ (dark gray). Notice that together they cover $[4ni, 4ni + 4n]$ on the second track. On the first track, the three intervals of length n (shown by red, green, and blue) are needed to complete the covering of the range $[4ni, 4ni + 4n]$. These intervals can be provided by the 2-track intervals $I^*_{e^0}$, $I^*_{e^1}$, $I^*_{e^2}$, where e^c is an edge of G incident to $v_{i,j}$ and having color c.

Proof. We prove the theorem by a reduction from Partitioned Subgraph Isomorphism. Let H and G be two graphs, let $V(H) = \{u_1, \dots, u_k\}$, and let $(V_1, \dots, V_k)$ be a partition of $V(G)$. By copying vertices if necessary, we may assume that every V_i has the same size n; let us denote by $\{v_{i,1}, \dots, v_{i,n}\}$ the vertices in V_i. As H is bipartite and 3-regular, the edges of H are 3-colorable. Let us fix a 3-coloring of the edges of H with colors in $\{0, 1, 2\}$. If there is an edge $u_{i_a}u_{i_b}$ in H having color $c \in \{0, 1, 2\}$, then we will also refer to the edges of G between V_{i_a} and V_{i_b} as having color c. We may assume that every edge of G receives a color this way, that is, there is no edge induced by any V_i and there can be an edge between V_{i_a} and V_{i_b} only if $u_{i_a}u_{i_b}$ is an edge of H (every other edge of G is useless for the purposes of this problem). As H is 3-regular, the two partite classes of H have to be of the same size; we may assume without loss of generality that $\{u_1, \dots, u_{k/2}\}$ and $\{u_{k/2+1}, \dots, u_k\}$ are these two classes.

Construction. We define the set $\mathcal{I}$ of 2-track intervals the following way (see Figure 15). First, for every $1 \le i \le k/2$ and $1 \le j \le n$, let us introduce the following two 2-track intervals into $\mathcal{I}$:

- $I_{i,j,1} := ([4ni, 4ni + j], [4ni, 4ni + n - j])$
- $I_{i,j,2} := ([4ni + 3n + j, 4ni + 4n], [4ni + n - j, 4ni + 4n])$

For every $k/2 + 1 \le i \le k$ and $1 \le j \le n$, we proceed similarly, but exchanging the role of the two tracks. That is, we introduce the following two 2-track intervals into $\mathcal{I}$:

- $I_{i,j,1} := ([4ni, 4ni + n - j], [4ni, 4ni + j])$
- $I_{i,j,2} := ([4ni + n - j, 4ni + 4n], [4ni + 3n + j, 4ni + 4n])$.

Next, we introduce a 2-track interval into $\mathcal{I}$ for every edge of G. Let $e = v_{i_a,j_a}v_{i_b,j_b}$ be an edge of G with $1 \le i_a \le k/2$ and $k/2 + 1 \le i_b \le k$. Suppose that this edge e has color $c \in \{0, 1, 2\}$ in the edge coloring we fixed. Then we introduce

- $I^*_e := ([4ni_a + cn + j_a, 4ni_a + (c + 1)n + j_a], [4ni_b + cn + j_b, 4ni_b + (c + 1)n + j_b])$

into the set $\mathcal{I}$. This completes the construction of $\mathcal{I}$; note that $|\mathcal{I}| = 3.5nk$ (as H is 3-regular).

Let $\mathcal{P} = \{4n + 0.5, 4n + 1.5, \dots, 4n(k + 1) - 0.5\}$. Let $k' = 2|V(H)| + |E(H)| = 3.5k$. We claim that it is possible to select k' members of $\mathcal{I}$ to cover the 2-track point set $(\mathcal{P}, \mathcal{P})$ if and only if the Partitioned Subgraph Isomorphism instance has a solution.

Subgraph embedding $\Rightarrow$ 2-track intervals. Suppose first that vertices $v_{1,s_1}, \ldots, v_{k,s_k}$ form a solution of the PARTITIONED SUBGRAPH ISOMORPHISM instance. Then we can cover $(\mathcal{P},\mathcal{P})$ by selecting the following set $\mathcal{I}'$ of 2-track intervals:

- $I_{i,s_i,1}$ and $I_{i,s_i,2}$ for $1 \le i \le k$, and
- for every edge $u_{i_a}u_{i_b}$ of H (with $1 \le i_a \le k/2$ and $k/2+1 \le i_b \le k$), edge $e = v_{i_a,s_{i_a}}v_{i_b,s_{i_b}}$ has to be an edge of G; we select the corresponding 2-track interval I^*_e.

We claim that every point in $[4n, 4n(k+1)]$ is covered by $\mathcal{I}'$ on both tracks; this clearly implies that $(\mathcal{P},\mathcal{P})$ is covered. Consider the range $[4ni, 4ni+4n]$ for some $1 \le i \le k/2$ (the case $k/2+1 \le i \le k$ is similar). On the second track, $I_{i,s_i,1}$ covers $[4ni, 4ni+n-s_i]$ and $I_{i,s_i,2}$ covers $[4ni+n-s_i, 4ni+4n]$, thus the range is indeed covered (see Figure 15). Let $k/2 \le i^0, i^1, i^2 \le k$ be the three neighbors of u_i in H, with the edge $u_iu_{i^c}$ having color c. This means that the edge $e^c = v_{i,s_i}v_{i^c,s_{i^c}}$ exists in G and $I^*_{e^c}$ was selected into $\mathcal{I}'$ for every $c \in \{0,1,2\}$. Recall from the definition that $I^*_{e^c}$ covers $[4ni+cn+s_i, 4ni+(c+1)n+s_i]$ on the first track. Now we have that, on the first track,

- $[4ni, 4ni+s_i]$ is covered by $I_{i,j,1}$,
- $[4ni+s_i, 4ni+n+s_i]$ is covered by $I^*_{e^0}$,
- $[4ni+n+s_i, 4ni+2n+s_i]$ is covered by $I^*_{e^1}$,
- $[4ni+2n+s_i, 4ni+3n+s_i]$ is covered by $I^*_{e^2}$, and
- $[4ni+3n+s_i, 4ni+4n]$ is covered by $I_{i,j,2}$.

Thus $[4ni, 4ni+4n]$ is indeed covered and we have shown that $(\mathcal{P},\mathcal{P})$ is covered by $\mathcal{I}'$.

2-track intervals $\Rightarrow$ subgraph embedding. Suppose now that there is subset $\mathcal{I}' \subseteq \mathcal{I}$ of size k' that covers the 2-track point set $(\mathcal{P},\mathcal{P})$. Let us define the following subsets of $\mathcal{I}$:

- $\mathcal{I}_{i,1} := \{I_{i,j,1} \mid 1 \le j \le n\}$,
- $\mathcal{I}_{i,2} := \{I_{i,j,2} \mid 1 \le j \le n\}$,
- $\mathcal{I}^*_{i,c} := \{I^*_e \mid$ edge e has color c and has an endpoint in $V_i\}$.

Notice that if $u_{i_a}u_{i_b}$ is an edge of H having color c, then $\mathcal{I}^*_{i_a,c} = \mathcal{I}^*_{i_b,c}$ (as every edge with color c and having an endpoint in V_{i_a} has its other endpoint in V_{i_b}). Looking at the definitions, we can observe that

- for $1 \le i \le k$, point $4ni+0.5$ on the first track is covered only by members of $\mathcal{I}_{i,1}$,
- for $1 \le i \le k/2$, point $4ni+4n-0.5$ on the second track is covered only by members of $\mathcal{I}_{i,2}$,
- for $k/2+1 \le i \le k$, point $4ni+4n-0.5$ on the first track is covered only by members of $\mathcal{I}_{i,2}$,
- for $1 \le i \le k/2$, $0 \le c \le 2$, point $4ni+(c+1)n+0.5$ on the first track is covered only by members of $I^*_{i,c}$, and
- for $k/2+1 \le i \le k$, $0 \le c \le 2$, point $4ni+(c+1)n+0.5$ on the second track is covered only by members of $I^*_{i,c}$, and

That is, at least one member has to be selected from each of the $5k$ sets $\mathcal{I}_{i,1}$, $\mathcal{I}_{i,2}$, $\mathcal{I}^*_{i,c}$. Note that every member of $\mathcal{I}$ is contained either in exactly one of these sets (in $\mathcal{I}_{i,1}$ or $\mathcal{I}_{i,2}$ for some $1 \le i \le k$) or in exactly two of these sets (in $\mathcal{I}^*_{i_a,c}$ and $\mathcal{I}^*_{i_b,c}$ for some $1 \le i_a \le k/2$, $k/2+1 \le i_b \le k$, $c \in \{0,1,2\}$). Therefore, the only way to cover the $5k$ points enumerated above by selecting $k' = 3.5k$ members of $\mathcal{I}$ is to select exactly one member of each $\mathcal{I}_{i,1}$, $\mathcal{I}_{i,2}$, and $\mathcal{I}^*_{i,c}$.

Suppose that for some $1 \le i \le k/2$, the solution $\mathcal{I}'$ contains the unique members $I_{i,j} \in \mathcal{I}_{i,1}$, $I_{i,j'} \in \mathcal{I}_{i,2}$, and $I^*_{e^c} \in \mathcal{I}^*_{i,c}$ for $c \in \{0,1,2\}$. By definition of $\mathcal{I}^*_{i,c}$, edge e^c has an endpoint $v_{i,j^c} \in V_i$. We claim that $j = j' = j^0 = j^1 = j^2$. Recall that $I_{i,j,1}$ covers $[4ni, 4ni+n-j]$ on the second track, while $I_{i,j',2}$ covers $[4ni+n-j', 4ni+4n]$ on the second track. Thus if $j > j'$, then point

$4ni + n - j' - 0.5$ would not be covered on the second track; hence we have $j' \geq j$. Observe next that $I_{i,j,1}$ covers $[4ni, 4ni + j]$ and $I^*_{e^0}$ covers $[4ni + j^0, 4ni + n + j^0]$ on the first track. Thus we infer that $j \geq j^0$: otherwise, point $4ni + j^0 - 0.5$ on the first track would not be covered. In a similar way, we can show that $j^1 \geq j^2$ and $j^2 \geq j'$ also hold. Therefore, we get the chain of inequalities

$$j \geq j^0 \geq j^1 \geq j^2 \geq j' \geq j,$$

implying that we have equalities throughout, as claimed.

In the previous paragraph, we have shown that for every $1 \leq i \leq k/2$, there is a $1 \leq s_i \leq n$ such that $\mathcal{I}'$ contains the unique members $I_{i,s_i} \in \mathcal{I}_{i,1}$, $I_{i,s_i} \in \mathcal{I}_{i,2}$, and $I^*_{e^c} \in \mathcal{I}^*_{i,c}$ for $c \in \{0, 1, 2\}$ with v_{i,s_i} being an endpoint of e^c. In a similar way, we can prove the same statement for every $k/2 + 1 \leq i \leq k$. We claim that $v_{1,s_1}, \ldots, v_{k,s_k}$ form a solution. Suppose that $u_{i_a}u_{i_b}$ for some $1 \leq i_a \leq k/2$, $k/2 + 1 \leq i_b \leq k$ is an edge of H, having color $c \in \{0, 1, 2\}$. We have to show that $v_{i_a,s_{i_a}}$ and $v_{i_b,s_{i_b}}$ are adjacent in G. Recall that $\mathcal{I}^*_{i_a,c} = \mathcal{I}^*_{i_b,c}$ and consider the unique member $I^*_e \in \mathcal{I}^*_{i_a,c}$ selected into $\mathcal{I}'$. As shown above, this means that edge e of G has an endpoint $v_{i_a,s_{i_a}} \in V_{i_a}$ and also an endpoint $v_{i_b,s_{i_b}} \in V_{i_b}$, what we had to show. □

We are now able to prove Theorem 1.11 by a simple and transparent reduction from covering points by 2-track intervals. The basic idea is that if we have two parallel lines, then for any pair of segments on these two lines, there is a rectangle that intersects these lines exactly at these two segments. Therefore, by arranging the points of a 2-track point set on two parallel lines, we can easily turn the problem of covering by 2-track intervals into a problem of covering by rectangles. Chan and Grant [7] used the same reduction to establish the APX-hardness of the problem (by reducing a certain special case of covering by 2-track intervals). Thus once we have Theorem 5.6, our lower bound for covering by 2-track intervals, the reduction of Chan and Grant [7] implies Theorem 1.11.

Restated Theorem 1.11 (covering points with almost squares, lower bound). *Consider the problem of covering a set $\mathcal{P}$ of points by selecting k axis-parallel rectangles from a set $\mathcal{D}$. Assuming ETH, for every $\epsilon_0 > 0$, there is no algorithm for this problem with running time $f(k) \cdot (|\mathcal{P}| + |\mathcal{D}|)^{o(k/\log k)}$ for any computable function f, even if each rectangle in $\mathcal{D}$ has both width and height in the range $[1 - \epsilon_0, 1 + \epsilon_0]$.*

Proof. The proof is by reduction from the problem of covering points by 2-track intervals. Let $\mathcal{I}$ be a set of 2-track intervals, let $\mathcal{P} = (\mathcal{P}^1, \mathcal{P}^2)$ be a 2-track point set, and let k be an integer. By rescaling if necessary, we may assume without loss of generality that every point in $\mathcal{P}^1$ and $\mathcal{P}^2$ is in the interval $[0, 1]$ and hence every $(I^1, I^2) \in \mathcal{I}$ satisfies $I^1, I^2 \subseteq [0, 1]$. As shown in Figure 16, we arrange the points of $\mathcal{P}^1$ (resp., $\mathcal{P}^2$) on the line $y = x + 1$ (resp., $y = x - 1$). That is, we construct a 2-dimensional point set $\mathcal{P}^*$ that contains

- point $(\epsilon_0 s, 1 + \epsilon_0 s)$ for every $s \in \mathcal{P}^1$ and
- point $(1 + \epsilon_0 s, \epsilon_0 s)$ for every $s \in \mathcal{P}^2$.

Then we represent every 2-track interval $([a_1, b_1], [a_2, b_2]) \in \mathcal{I}$ by a rectangle whose horizontal projection is $[\epsilon_0 a_1, 1 + \epsilon_0 b_2]$ and vertical projection is $[\epsilon_0 a_2, 1 + \epsilon_0 b_1]$; note that in this proof we assume that the rectangles are closed, but to obtain the same result for open rectangles we can simply make them slightly larger. Observe that the sizes of both projections are between $1 - \epsilon_0$ and $1 + \epsilon_0$. Moreover, these rectangles faithfully represent the way the 2-track intervals cover the points in $(\mathcal{P}^1, \mathcal{P}^2)$. For example, 2-track interval $([a_1, b_1], [a_2, b_2])$ covers point s on the first track if and only if $a_1 \leq s \leq b_1$ and the corresponding rectangle covers point $(\epsilon_0 s, 1 + \epsilon_0 s)$ exactly under the

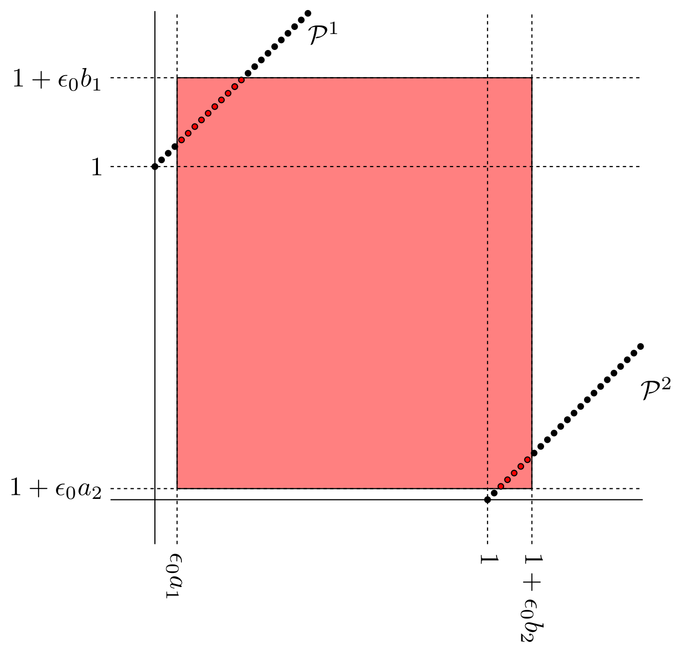


Figure 16: Proof of Theorem 1.11. The rectangle $[\epsilon_0 a_1, 1 + \epsilon_0 b_2] \times [\epsilon_0 a_2, 1 + \epsilon_0 b_1]$ covers exactly those points that correspond to points in $[a_1, b_1]$ on the first track and or to points in $[a_2, b_2]$ on the second track.

same condition (if $s < a_1$, then the point is to the left of the rectangle; if $s > b_1$, then the point is above the rectangle). Therefore, we can cover the point set $\mathcal{P}^*$ with k of the constructed rectangles if and only if we can cover the 2-track point set $\mathcal{P}$ with k of the 2-track intervals from $\mathcal{I}$. $\square$

References

[1] A. Adamaszek and A. Wiese. Approximation schemes for maximum weight independent set of rectangles. In *FOCS 2013*, pages 400–409, 2013.

[2] A. Adamaszek and A. Wiese. A QPTAS for maximum weight independent set of polygons with polylogarithmically many vertices. In *SODA 2014*, pages 645–656, 2014.

[3] P. K. Agarwal, M. H. Overmars, and M. Sharir. Computing maximally separated sets in the plane. *SIAM J. Comput.*, 36(3):815–834, 2006.

[4] P. K. Agarwal and C. M. Procopiuc. Exact and approximation algorithms for clustering. *Algorithmica*, 33(2):201–226, 2002.

[5] J. Alber and J. Fiala. Geometric separation and exact solutions for the parameterized independent set problem on disk graphs. *J. Algorithms*, 52(2):134–151, 2004.

[6] A. Butman, D. Hermelin, M. Lewenstein, and D. Rawitz. Optimization problems in multiple-interval graphs. *ACM Transactions on Algorithms*, 6(2), 2010.

[7] T. M. Chan and E. Grant. Exact algorithms and apx-hardness results for geometric packing and covering problems. *Comput. Geom.*, 47(2):112–124, 2014.

[8] J. Chen, X. Huang, I. A. Kanj, and G. Xia. Strong computational lower bounds via parameterized complexity. *J. Comput. Syst. Sci.*, 72(8):1346–1367, 2006.

[9] R. H. Chitnis, M. Hajiaghayi, and D. Marx. Tight bounds for Planar Strongly Connected Steiner Subgraph with fixed number of terminals (and extensions). In *SODA 2014*, pages 1782–1801, 2014.

[10] H. Dell and D. Marx. Kernelization of packing problems. In *SODA 2012*, pages 68–81, 2012.

[11] E. D. Demaine, F. V. Fomin, M. T. Hajiaghayi, and D. M. Thilikos. Bidimensional parameters and local treewidth. *SIAM J. Discrete Math.*, 18(3):501–511, 2004.

[12] E. D. Demaine, F. V. Fomin, M. T. Hajiaghayi, and D. M. Thilikos. Fixed-parameter algorithms for (k, r)-Center in planar graphs and map graphs. *ACM Transactions on Algorithms*, 1(1):33–47, 2005.

[13] E. D. Demaine, F. V. Fomin, M. T. Hajiaghayi, and D. M. Thilikos. Subexponential parameterized algorithms on bounded-genus graphs and H-minor-free graphs. *J. ACM*, 52(6):866–893, 2005.

[14] E. D. Demaine and M. Hajiaghayi. The bidimensionality theory and its algorithmic applications. *Comput. J.*, 51(3):292–302, 2008.

[15] E. D. Demaine and M. Hajiaghayi. Linearity of grid minors in treewidth with applications through bidimensionality. *Combinatorica*, 28(1):19–36, 2008.

[16] E. D. Demaine and M. T. Hajiaghayi. Fast algorithms for hard graph problems: Bidimensionality, minors, and local treewidth. In *Graph Drawing*, pages 517–533, 2004.

[17] F. Dorn, F. V. Fomin, D. Lokshtanov, V. Raman, and S. Saurabh. Beyond bidimensionality: Parameterized subexponential algorithms on directed graphs. In *STACS 2010*, pages 251–262, 2010.

[18] F. Dorn, F. V. Fomin, and D. M. Thilikos. Subexponential parameterized algorithms. *Computer Science Review*, 2(1):29–39, 2008.

[19] F. Dorn, E. Penninkx, H. L. Bodlaender, and F. V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010.

[20] M. R. Fellows, D. Hermelin, F. A. Rosamond, and S. Vialette. On the parameterized complexity of multiple-interval graph problems. *Theor. Comput. Sci.*, 410(1):53–61, 2009.

[21] F. V. Fomin, S. Gaspers, P. A. Golovach, K. Suchan, S. Szeider, E. J. van Leeuwen, M. Vatshelle, and Y. Villanger. k-gap interval graphs. In D. Fernández-Baca, editor, *LATIN 2012*, volume 7256 of *Lecture Notes in Computer Science*, pages 350–361. Springer, 2012.

[22] F. V. Fomin, D. Lokshtanov, V. Raman, and S. Saurabh. Subexponential algorithms for partial cover problems. *Inf. Process. Lett.*, 111(16):814–818, 2011.

[23] F. V. Fomin and D. M. Thilikos. A simple and fast approach for solving problems on planar graphs. In *STACS 2004*, pages 56–67, 2004.

[24] F. V. Fomin and D. M. Thilikos. Dominating sets in planar graphs: Branch-width and exponential speed-up. *SIAM J. Comput.*, 36(2):281–309, 2006.

[25] M. C. Francis, D. Gonçalves, and P. Ochem. The maximum clique problem in multiple interval graphs. *Algorithmica*, 71(4):812–836, 2015.

[26] Q.-P. Gu and H. Tamaki. Optimal branch-decomposition of planar graphs in $O(n^3)$ time. *ACM Transactions on Algorithms*, 4(3), 2008.

[27] S. Har-Peled. Being fat and friendly is not enough. *CoRR*, abs/0908.2369, 2009.

[28] S. Har-Peled. Quasi-polynomial time approximation scheme for sparse subsets of polygons. In *SoCG 2014*, page 120, 2014.

[29] R. Z. Hwang, R. C. T. Lee, and R. C. Chang. The slab dividing approach to solve the euclidean p-center problem. *Algorithmica*, 9(1):1–22, 1993.

[30] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. System Sci.*, 63(4):512–530, 2001.

[31] M. Jiang. Recognizing d-interval graphs and d-track interval graphs. *Algorithmica*, 66(3):541–563, 2013.

[32] M. Jiang and Y. Zhang. Parameterized complexity in multiple-interval graphs: Domination, partition, separation, irredundancy. *Theor. Comput. Sci.*, 461:27–44, 2012.

[33] P. N. Klein and D. Marx. Solving Planar k-Terminal Cut in $O(n^{c\sqrt{k}})$ time. In *ICALP 2012 (1)*, pages 569–580, 2012.

[34] P. N. Klein and D. Marx. A subexponential parameterized algorithm for Subset TSP on planar graphs. In *SODA 2014*, pages 1812–1830, 2014.

[35] D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the Exponential Time Hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

[36] D. Marx. Efficient approximation schemes for geometric problems? In *ESA 2005*, pages 448–459, 2005.

[37] D. Marx. Parameterized complexity of independence and domination on geometric graphs. In *IWPEC 2006*, pages 154–165, 2006.

[38] D. Marx. On the optimality of planar and geometric approximation schemes. In *FOCS 2007*, pages 338–348, 2007.

[39] D. Marx. Can you beat treewidth? *Theory of Computing*, 6(1):85–112, 2010.

[40] D. Marx and A. Sidiropoulos. The limited blessing of low dimensionality: when $1 - 1/d$ is the best possible exponent for d-dimensional geometric problems. In *SoCG 2014*, page 67, 2014.

[41] N. H. Mustafa, R. Raman, and S. Ray. QPTAS for geometric set-cover problems via optimal separators. *CoRR*, abs/1403.0835, 2014.

[42] M. Pilipczuk, M. Pilipczuk, P. Sankowski, and E. J. van Leeuwen. Subexponential-time parameterized algorithm for Steiner Tree on planar graphs. In *STACS 2013*, pages 353–364, 2013.

[43] M. Pilipczuk, M. Pilipczuk, P. Sankowski, and E. J. van Leeuwen. Network sparsification for Steiner problems on planar and bounded-genus graphs. In *FOCS 2014*, pages 276–285. IEEE Computer Society, 2014.

[44] M. Pătraşcu and R. Williams. On the possibility of faster SAT algorithms. In *SODA 2010*, pages 1065–1075, 2010.

[45] P. D. Seymour and R. Thomas. Call routing and the ratcatcher. *Combinatorica*, 14(2):217–241, 1994.

[46] W. D. Smith and N. C. Wormald. Geometric separator theorems & applications. In *FOCS 1998*, pages 232–243, 1998.

[47] D. M. Thilikos. Fast sub-exponential algorithms and compactness in planar graphs. In *ESA 2011*, pages 358–369, 2011.