Abstract

Suppose we are given a pair of points \( s, t \) and a set \( \mathcal{S} \) of \( n \) geometric objects in the plane, called obstacles. We show that in polynomial time one can construct an auxiliary (multi-)graph \( G \) with vertex set \( \mathcal{S} \) and every edge labeled from \{0, 1\}, such that a set \( \mathcal{S}_d \subseteq \mathcal{S} \) of obstacles separates \( s \) from \( t \) if and only if \( G[\mathcal{S}_d] \) contains a cycle whose sum of labels is odd. Using this structural characterization of separating sets of obstacles we obtain the following algorithmic results.

In the Obstacle-removal problem the task is to find a curve in the plane connecting \( s \) to \( t \) intersecting at most \( q \) obstacles. We give a \( 2.3146n^{O(1)} \) algorithm for Obstacle-removal, significantly improving upon the previously best known \( q^{O(q^3)}n^{O(1)} \) algorithm of Eiben and Lokshtanov (SoCG’20). We also obtain an alternative proof of a constant factor approximation algorithm for Obstacle-removal, substantially simplifying the arguments of Kumar et al. (SODA’21).

In the Generalized Points-separation problem input consists of the set \( \mathcal{S} \) of obstacles, a point set \( \mathcal{A} \) of \( k \) points and \( p \) pairs \((s_1, t_1), \ldots, (s_p, t_p)\) of points from \( \mathcal{A} \). The task is to find a minimum subset \( \mathcal{S}_r \subseteq \mathcal{S} \) such that for every \( i \), every curve from \( s_i \) to \( t_i \) intersects at least one obstacle in \( \mathcal{S}_r \). We obtain \( 2^{O(p)}k^{O(k)}n^{O(k)} \)-time algorithm for Generalized Points-separation. This resolves an open problem of Cabello and Giannopoulos (SoCG’13), who asked about the existence of such an algorithm for the special case where \((s_1, t_1), \ldots, (s_p, t_p)\) contains all the pairs of points in \( \mathcal{A} \). Finally, we improve the running time of our algorithm to \( f(p, k) \cdot n^{O(k)} \) when the obstacles are unit disks, where \( f(p, k) = 2^{O(p)}k^{O(k)} \), and show that, assuming the Exponential Time Hypothesis (ETH), the running time dependence on \( k \) of our algorithms is essentially optimal.

Introduction

Suppose we are given a set \( \mathcal{S} \) of geometric objects in the plane, and we want to modify \( \mathcal{S} \) in order to achieve certain guarantees on coverage of paths between a given set \( \mathcal{A} \) of points. Such problems have received significant interest in sensor networks [3, 5, 7, 20], robotics [11, 14] and computational geometry [4, 10, 13]. There have been two closely related lines of work on this topic: (i) remove a smallest number of obstacles from \( \mathcal{S} \) to satisfy reachability requirements for points in \( \mathcal{A} \), and (ii) retain a smallest number of obstacles to satisfy separation requirements for points in \( \mathcal{A} \).
In the most basic version of these problems the set $A$ consists of just two points $s$ and $t$. Specifically, in Obstacle-removal the task is to find a smallest possible set $S_d \subseteq S$ such that there is a curve from $s$ to $t$ in the plane avoiding all obstacles in $S \setminus S_d$. In 2-Points-separation the task is to find a smallest set $S_r \subseteq S$ such that every curve from $s$ to $t$ in the plane intersects at least one obstacle in $S_r$. It is quite natural to require the obstacles in the set $S$ to be connected. Indeed, removing the connectivity requirements results in problems that are computationally intractable [10, 12, 25].

When the obstacles are required to be connected Obstacle-removal remains $\text{NP}$-hard, but becomes more tractable from the perspective of approximation algorithms and parameterized algorithms. For approximation algorithms, Bereg and Kirkpatrick [5] designed a constant factor approximation for unit disk obstacles. Chan and Kirkpatrick [7, 8] improved the approximation factor for unit disk obstacles. Korman et al. [18] obtained a $(1 + \epsilon)$-approximation algorithm for the case when obstacles are fat, similarly sized, and no point in the plane is contained in more than a constant number of obstacles. Whether a constant factor approximation exists for general obstacles was posed repeatedly as an open problem [4, 7, 8] before it was resolved in the affirmative by a subset of the authors of this article [25].

For parameterized algorithms, Korman et al. [18] designed an algorithm for Obstacle-removal with running time $f(q)n^{O(1)}$ for determining whether there exists a solution $S_d$ of size at most $q$, when obstacles are fat, similarly sized, and no point in the plane is contained in more than a constant number of obstacles. Eiben and Kanj [10, 12] generalized the result of Korman et al. [18], and posed as an open problem the existence of a $f(q)n^{O(1)}$ time algorithm for Obstacle-removal with general connected obstacles. Eiben and Lokshavan [13] resolved this problem in the affirmative, providing an algorithm with running time $q^{O(q^2)}n^{O(1)}$.

Like Obstacle-removal, the 2-Points-separation problem becomes more tractable when the obstacles are connected. Cabello and Giannopoulos [6] showed that 2-Points-separation with connected obstacles is polynomial time solvable. They show that the more general Points-separation problem where we are given a point set $A$ and asked to find a minimum size set $S_r \subseteq S$ that separates every pair of points in $A$, is $\text{NP}$-complete, even when all obstacles are unit disks. They leave as an open problem to determine the existence of $f(k)n^{O(1)}$ and $f(k)n^{\theta(k)}$ time algorithms for Points-separation, where $k = |A|$.

Our Results and Techniques

Our main result is a structural characterization of separating sets of obstacles in terms of odd cycles in an auxiliary graph.

> **Theorem 1.** There exists a polynomial time algorithm that takes as input a set $S$ of obstacles in the plane, two points $s$ and $t$, and outputs a (multi-)graph $G$ with vertex set $S$ and every edge labeled from $\{0, 1\}$, such that a set $S_d \subseteq S$ of obstacles separates $s$ from $t$ if and only if $G[S_d]$ contains a cycle whose sum of labels is odd.

The proof of Theorem 1 is an application of the well known fact that a closed curve separates $s$ from $t$ if and only if it crosses a curve from $s$ to $t$ an odd number of times. Theorem 1 allows us to re-prove, improve, and generalize a number of results for Obstacle-removal, 2-Points-separation and Points-separation in a remarkably simple way. More concretely, we obtain the following results.

> There exists a polynomial time algorithm for 2-Points-separation.

Here is the proof: construct the graph $G$ from Theorem 1 and find the shortest odd cycle, which is easy to do in polynomial time. This re-proves the main result of Cabello
and Giannopoulos [6]. Next we turn to OBSTACLE-REMVAL, and obtain an improved parameterized algorithm and simplified approximation algorithms.

- There exists an algorithm for OBSTACLE-REMVAL that determines whether there exists a solution size set \( S \) of size at most \( q \) in time \( 2.3146^n n^{O(1)} \).

Here is a proof sketch: construct the graph \( G \) from Theorem 1 and determine whether there exists a subset \( S_1 \) of \( S \) of size at most \( q \) such that \( G - S \) does not have any odd label cycle. This can be done in time \( 2.3146^n n^{O(1)} \) using the algorithm of Lokshtanov et al. [22] for Odd Cycle Transversal. This parameterized algorithm improves over the previously best known parameterized algorithm for Obstacle-removal of Eiben and Lokshtanov [13] with running time \( q^{O(q^2)} n^{O(1)} \).

If we run an approximation algorithm for Odd Cycle Transversal on \( G \) instead of a parameterized algorithm, we immediately obtain an approximation algorithm for Obstacle-removal with the same ratio. Thus, the \( O(\sqrt{\log n}) \)-approximation algorithm for Odd Cycle Transversal \([2, 19]\) implies a \( O(\sqrt{\log n}) \)-approximation algorithm for Obstacle-removal as well. Going a little deeper we observe that the standard Linear Programming relaxation of Odd Cycle Transversal on \( G \) only has a constant integrality gap. This yields a constant factor approximation for Obstacle-removal, substantially simplifying the approximation algorithm of Kumar et al \([25]\).

- There exists a a constant factor approximation for Obstacle-removal.

Finally we turn our attention back to a generalization of Points-separation, called Generalized Points-separation. Here, instead of separating all \( k \) points in \( A \) from each other, we are only required to separate \( p \) specific pairs \((s_1, t_1), \ldots, (s_p, t_p)\) of points in \( A \) (which are specified in the input). We apply Theorem 1 several times, each time with the same obstacle set \( S \), but with a different pair \((s_i, t_i)\). Let \( G_i \) be the graph resulting from the construction with the pair \((s_i, t_i)\). Finding a minimum size set \( S_r \) of obstacles that separates \( s_i \) from \( t_i \) for every \( i \) now amounts to finding a minimum size set \( S_r \) such that \( G_i[S_r] \) contains an odd label cycle for every \( i \). The graph in the construction of Theorem 1 does not depend on the points \((s_i, t_i)\) - only the labels of the edges do. Thus \( G_1, \ldots, G_p \) are copies of the same graph \( G \), but with \( p \) different edge labelings. Our task now is to find a subgraph of \( G \) on the minimum number of vertices, such that the subgraph contains an odd labeled cycle with respect to each one of the \( p \) labels. We show that such a subgraph has at most \( O(p) \) vertices of degree at least 3 and use this to obtain a \( 2^{O(p^2)} n^{O(p)} \) time algorithm for Generalized Points-separation. This implies a \( 2^{O(k^2)} n^{O(k)} \) time algorithm for Points-separation, resolving the open problem of Cabello and Giannopoulos [6]. With additional technical effort we are able to bring down the running time of our algorithm for Generalized Points-separation to \( 2^{O(p)} n^{O(k)} \). This turns out to be close to the best one can do. On the other hand, for pseudo-disk obstacles we can get a faster algorithm.

- There exists a \( 2^{O(p)} n^{O(k)} \) time algorithm for Generalized Points-separation, and a \( n^{O(\sqrt{k})} \) time algorithm for Generalized Points-separation with pseudo-disk obstacles.
- A \( f(k)n^{o(k/\log k)} \) time algorithm for Points-separation, or a \( f(k)n^{o(\sqrt{k})} \) time algorithm for Points-separation with pseudo-disk obstacles would violate the ETH \([16]\).

---

1 The only reason this is a proof sketch rather than a proof is that the algorithm of Lokshtanov et al. [22] works for unlabeled graphs, while \( G \) has edges with labels 0 or 1. This difference can be worked out using a well-known and simple trick of subdividing every edge with label 0 (see Section 4).
2 Preliminaries

We begin by reviewing some relevant background and definitions.

Graphs and Arrangements All graphs used in this paper are undirected. It will also be more convenient to sometimes consider multi-graphs, in which self-loops and parallel edges are allowed. The degree of a vertex is the number of adjacent edges.

The arrangement $\text{Arr}(S)$ of a set of obstacles $S$ is a subdivision of the plane induced by the boundaries of the obstacles in $S$. The faces of $\text{Arr}(S)$ are connected regions and edges are parts of obstacle boundaries. The arrangement graph $G_{\text{Arr}} = (V, E)$ is the dual graph of the arrangement whose vertices are faces of $\text{Arr}(S)$ and edges connect neighboring faces. The complexity of the arrangement is the size of its arrangement graph which we denote by $|\text{Arr}(S)|$. We assume that the size of the arrangement is polynomial in the number of obstacles, that is $|\text{Arr}(S)| = |G_{\text{Arr}}| = n^{O(1)}$. This is indeed true for most reasonable obstacle models such as polygons or low-degree splines.

Obstacle-removal and Points-separation on Colored Graphs Traditionally, Obstacle-removal problems have been defined in terms of graph problems on the arrangement graph $G_{\text{Arr}}$. In particular, we can define a coloring function $\text{col} : V \rightarrow 2^S$ which assigns every vertex of $G_{\text{Arr}}$ to the set of obstacles containing it. That is, obstacles correspond to colors in the colored graph $(G_{\text{Arr}}, \text{col})$. It is easy to see that a curve connecting $s$ and $t$ in the plane that intersects $q$ obstacles corresponds to a path $\pi$ in the graph that uses $|\bigcup_{v \in \pi} \text{col}(v)| = q$ colors in $(G_{\text{Arr}}, \text{col})$ and vice versa.

We can also define 2-Points-separation as the problem of computing a min-color separator of the graph $(G_{\text{Arr}}, \text{col})$. Let $V(S_r) \subseteq V$ be the set of vertices of $G_{\text{Arr}}$ that contain at least one color from $S_r$. A set of colors $S_r \subseteq S$ is a color separator if $s$ and $t$ are disconnected in $G_{\text{Arr}} - V(S_r)$. That is, every $s$–$t$ path must intersect at least one color in $S_r$. Therefore, a color separator of minimum cardinality is a solution of 2-Points-separation, that is the minimum set of obstacles separating $s$ from $t$.

The previous work [25] used structural properties of the colored graph $(G_{\text{Arr}}, \text{col})$ to obtain a polytime algorithm for 2-Points-separation and a constant approximation for Obstacle-removal. One key difference in our approach is that instead of working on the colored graph $(G_{\text{Arr}}, \text{col})$, we found it more convenient to work with a so-called labeled intersection graph $(G_S, \text{lab})$ of obstacles which we will formally construct in the next section. Roughly speaking, given a set of obstacles $S$ and a reference curve $\pi$ in the plane connecting $s$ and $t$, we build a multi-graph where vertices are obstacles in $S$ and edges connect a pair of intersecting obstacles. Every edge $e \in E$ is assigned a parity label $\text{lab}(e) \in \{0, 1\}$ based on the reference curve $\pi$. We say that a walk is labeled odd (or even) if the sum of labels of its edges is odd (or even) respectively.

Once this graph is constructed, we can forget about obstacles and formulate our problems using just the parity labels $\text{lab}(e)$ on the edges of $G_S$. Since the parity function is much simpler to work with compared to the color function, this allows us to significantly simplify the results from [25] and obtain new results. In the next section, we describe the construction of graph $G_S$ and prove a key structural result that allow us to cast 2-Points-separation as finding shortest odd labeled cycle in $G_S$ and Obstacle-removal as the smallest Odd Cycle Transversal of $G_S$. Recall that in Odd Cycle Transversal problem, we want to find a set of vertices that “hits” (has non-empty intersection) with every odd-cycle of the graph. We will also need the following important property of plane curves.
Plane curves and Crossings. A plane curve (or simply curve) is specified by a continuous function \( \pi : [0, 1] \to \mathbb{R}^2 \), where the points \( \pi(0) \) and \( \pi(1) \) are called the endpoints (for convenience, we also use the notation \( \pi \) to denote the image of the path function \( \pi \)). A curve is simple if it is injective, and is closed if its two endpoints are the same. We say a curve \( \pi \) separates a pair \((a, b)\) of two points in \( \mathbb{R}^2 \) if \( a \) and \( b \) belong to different connected components of \( \mathbb{R}^2 \backslash \pi \).

A crossing of \( \pi \) with \( \pi' \) is an element of the set \( \{ t \in [0, 1] \mid \pi(t) \in \pi' \} \). We will often be concerned with the number of times \( \pi \) crosses \( \pi' \). This is defined as \( |\{ t \in [0, 1] \mid \pi(t) \in \pi' \}| \). Whenever we count the number of times a curve \( \pi \) crosses another curve \( \pi' \) we shall assume that (and ensure that) \( |\{ t \in [0, 1] \mid \pi(t) \in \pi' \}| \) is finite and that \( \pi \) and \( \pi' \) are transverse. That is for every \( t \in [0, 1] \) such that \( \pi(t) \in \pi' \) there exists an \( \epsilon > 0 \) such that the intersection of \( \pi \cup \pi' \) with an \( \epsilon \) radius ball around \( \pi(t) \) is homotopic with two orthogonal lines. We will make frequent use of the following basic topological fact.

\[ \textbf{Fact 2.} \text{ Let } \pi \text{ be a curve with endpoints } a, b \in \mathbb{R}^2. \text{ We have that} \]

- A simple closed curve \( \gamma \) separates \((a, b)\) \iff \( \pi \) crosses \( \gamma \) an odd number of times.

- If \( \pi \) crosses a closed curve \( \gamma \) an odd number of times, then \( \gamma \) separates \((a, b)\).

Partitions. A partition of a set \( X \) is a collection \( \Phi \) of nonempty disjoint subsets (called parts) of \( X \) whose union is \( X \). For two partitions \( \Phi \) and \( \Phi' \) of \( X \), we say \( \Phi \) is finer than \( \Phi' \), denoted by \( \Phi \preceq \Phi' \) or \( \Phi' \succeq \Phi \), if for any \( Y \in \Phi \) there exists \( Y' \in \Phi' \) such that \( Y \subseteq Y' \). There is a one-to-one correspondence between partitions of \( X \) and equivalence relations on \( X \). For any equivalence relation on \( X \), the set of its equivalence classes is a partition of \( X \). Conversely, any partition of \( X \) induces an equivalence relation \( \sim \) on \( X \) where \( x \sim y \) if \( x \) and \( y \) belong to the same part of the partition. For two partitions \( \Phi \) and \( \Phi' \) of \( X \), we define \( \Phi \odot \Phi' \) as another partition of \( X \) as follows. Let \( \sim_\Phi \) and \( \sim_{\Phi'} \) be the equivalence relations on \( X \) induced by \( \Phi \) and \( \Phi' \), respectively. Define \( \sim \) as the equivalence relation on \( X \) where \( x \sim y \) if \( x \sim_\Phi y \) and \( x \sim_{\Phi'} y \). Then \( \Phi \odot \Phi' \) is defined as the partition corresponding to the equivalence relation \( \sim \). Clearly, \( \odot \) is a commutative and associative binary operation. Thus, for a collection \( \text{Par} \) of partitions on \( X \), we can define \( \bigodot_{\Phi \in \text{Par}} \Phi \) as the partition on \( X \) obtained by “adding” the elements in \( \text{Par} \) using the operation \( \odot \); note that \( \bigodot_{\Phi \in \text{Par}} \Phi \) is well-defined even if \( \text{Par} \) is infinite.

\[ \textbf{Fact 3.} \text{ Let } X \text{ be a set of size } k \text{ and } \Phi_1, \ldots, \Phi_r \text{ be partitions of } X. \text{ Then there exists } T \subseteq [r] \text{ with } |T| < k \text{ such that } \bigodot_{t=1}^r \Phi_t = \bigodot_{t \in T} \Phi_t. \]

**Proof.** Let \( T \subseteq [r] \) be a minimal subset satisfying \( \bigodot_{t=1}^r \Phi_t = \bigodot_{t \in T} \Phi_t \). We show \( |T| < k \) by contradiction. Assume \( T = \{t_1, \ldots, t_m\} \) where \( m \geq k \). Define \( \Psi_s = \bigodot_{t \neq s} \Phi_t \), for \( s \in [m] \). Then we have \( \Psi_1 \geq \cdots \geq \Psi_m \), which implies \( 1 \leq |\Psi_1| \geq \cdots \geq |\Psi_m| \leq k \). It is impossible that \( 1 \leq |\Psi_1| < \cdots < |\Psi_m| \leq k \), because \( m \geq k \). Therefore, \( \Psi_s = \Psi_{s+1} \) for some \( s \in [m-1] \). It follows that

\[
\bigodot_{t \in T} \Phi_t = \Psi_{s+1} \odot \left( \bigodot_{i=s+2}^m \Phi_{t_i} \right) = \Psi_s \odot \left( \bigodot_{i=s+2}^m \Phi_{t_i} \right) = \bigodot_{t \in T \backslash \{t_{s+1}\}} \Phi_t,
\]

which contradicts the minimality of \( T \).

\[ \textbf{Fact 4.} \text{ Let } \Phi \text{ be a partition of } X \text{ and suppose } |\Phi| = z. \text{ For an integer } 0 \leq d < z, \text{ the number of partitions } \Phi' \text{ satisfying } |\Phi'| = z-d \text{ and } \Phi' \succeq \Phi \text{ is bounded by } z^\Theta(d). \text{ Furthermore, these partitions can be computed in } z^\Theta(d) \text{ time given } \Phi. \]
We begin by describing the construction of the labeled intersection graph \( G \). We remark that the above fact immediately implies another well-known property of pseudo-disks \([17]\).

\[ \text{Proof.} \] Consider the following procedure for generating a “coarser” partition from \( \Phi \). We begin from the partition \( \Phi \). At each step, we pick two elements \( Y, Y' \) in the current partition and then replace them with their union \( Y \cup Y' \) to obtain a new partition. After \( d \) steps, we obtain a partition \( \Phi' \) satisfying \( |\Phi'| = z - d \) and \( \Phi' \succeq \Phi \). Note that every partition \( \Phi' \) where \( |\Phi'| = z - d \) and \( \Phi' \succeq \Phi \) can be constructed in this way. Furthermore, the number of different choices at the \( i \)-th step is \( \binom{z + 1 - i}{2} = O(z^2) \). Therefore, the number of possible outcomes of the procedure, i.e., the number of partitions \( \Phi' \) satisfying \( |\Phi'| = z - d \) and \( \Phi' \succeq \Phi \), is bounded by \( z^{O(d)} \). These partitions can be directly computed in \( z^{O(d)} \) time via the procedure. \( \blacktriangle \)

**Pseudo-disks.** A set \( S \) of geometric objects in \( \mathbb{R}^2 \) is a set of pseudo-disks, if each object \( S \in S \) is topologically homeomorphic to a disk (and hence its boundary is a simple cycle in the plane) and the boundaries of any two objects \( S, S' \in S \) intersect at most twice. Let \( U \) be the union of a set \( S \) of pseudo-disks. The boundary of \( U \) consists of arcs (each of which is a portion of the boundary of an object in \( S \)) and break points (each of which is an intersection point of the boundaries of two objects in \( S \)). We say two objects \( S, S' \in S \) contribute to \( U \) if an intersection point of the boundaries of \( S \) and \( S' \) is a break point on the boundary of \( U \). We shall use the following well-known property of pseudo-disks \([17]\).

**Fact 5.** Let \( S \) be a set of pseudo-disks, and \( U \) be the union of the objects in \( S \). Then the graph \( G = (S, E) \) where \( E = \{(S, S') : S, S' \in S \text{ contribute to} \ U \} \) is planar.

We remark that the above fact immediately implies another well-known property of pseudo-disks: the complexity of the union of a set of \( n \) pseudo-disks is \( O(n) \) \([17]\). But this property will not be used in this paper.

## 3 Labeled Intersection Graph of Obstacles

We begin by describing the construction of the labeled intersection graph \( G_S = (S, X) \) of the obstacles \( S \). For the ease of exposition, we will use \( S \) to refer to the obstacle \( S \in S \) as well as the vertex for \( S \) in \( G_S \) interchangeably.

**Constructing the graph \( G_S \)** For every obstacle \( S \in S \) we first select an arbitrary point \( \text{ref}(S) \in S \) and designate it to be the reference point of the obstacle. Next, we select the reference curve \( \pi \) to be a simple curve in the plane connecting \( s \) and \( t \) such that including it to the arrangement \( \text{Arr}(S) \) does not significantly increase its complexity. That is, we want to ensure that \( |\text{Arr}(S \cup \pi)| = O(|\text{Arr}(S)|) \). Additionally, the reference curve \( \pi \) is chosen such that there exists an \( \epsilon > 0 \) and \( \pi \) is disjoint from an \( \epsilon \) ball around every intersection point of two obstacles in \( \text{Arr}(S) \) and from an \( \epsilon \) ball around every reference point \( \text{ref}(S) \) for \( S \in S \).

As long as the intersection of every pair of obstacles is finite and their arrangement has bounded size, a suitable choice for \( \pi \) always exists (and can be efficiently computed). For example one can choose \( \pi \) to be the plane curve corresponding to an \( s-t \) path in \( G_{\text{Arr}} \).

We will now add edges to \( G_S \) as follows. (See also Figure 1(c) for an example.)

- For every obstacle \( S \in S \) that contains \( s \) or \( t \), add a self-loop \( e = (S, S) \) with \( \text{lab}(e) = 1 \).
- For every pair of obstacles \( S, S' \in S \) that intersect, we add edges to \( G \) as follows.
  - Add an edge \( e_0 = (S, S') \) with \( \text{lab}(e_0) = 0 \) if there exists a curve connecting \( \text{ref}(S) \) and \( \text{ref}(S') \) contained in the region \( S \cup S' \) that crosses \( \pi \) an even number of times.
  - Add an edge \( e_1 = (S, S') \) with \( \text{lab}(e_1) = 1 \) if there exists a curve connecting \( \text{ref}(S) \) and \( \text{ref}(S') \) contained in the region \( S \cup S' \) that crosses \( \pi \) an odd number of times.
Checking whether there exists a curve contained in the region \( S \cup S' \) with endpoints \( \text{ref}(S) \) and \( \text{ref}(S') \) that crosses \( \pi \) an odd (resp. even) number of times can be done in time linear in the size of arrangement \( \text{Arr}' = \text{Arr}(S \cup S' \cup \pi) \). Specifically, we build the arrangement graph \( G_{\text{Arr}} \) and only retain edges \((f_i, f_j)\) such that the faces \( f_i, f_j \in S \cup S' \). If the common boundary of faces \( f_i, f_j \) is a portion of \( \pi \), we assign a label 1 to the edge \((f_i, f_j)\), otherwise we assign it a label 0. An odd (resp. even) labeled walk in \( G_{\text{Arr}} \) connecting the faces containing \( \text{ref}(S) \) and \( \text{ref}(S') \) gives us the desired plane curve \( \pi_{ij} \). Since edges of \( G_{\text{Arr}} \) connect adjacent faces of \( \text{Arr}' \), we can ensure that the intersections between curve \( \pi_{ij} \) and the edges of arrangement (including parts of reference curve \( \pi \)) are all transverse.

We are now ready to prove the following important structural property of the graph \( G_S \).

\[ \textbf{Lemma 6.} \] A set of obstacles \( S' \subseteq S \) in the graph \( G_S \) separates the points \( s \) and \( t \) if and only if the induced graph \( H = G_S[S'] \) contains an odd labeled cycle.

\textbf{Proof.} (\( \Rightarrow \)) For the forward direction, suppose we are given a set of obstacles \( S' \) that separate \( s \) from \( t \). If \( s \) or \( t \) are contained in some obstacle, then we must have an odd self-loop in \( G_S \) and we will be done. Otherwise, assume that \( s, t \) lie in the exterior of all obstacles, so we have \( s, t \notin \mathcal{R}(S') \) where \( \mathcal{R}(S') = \bigcup_{S \in S'} S \) is the region bounded by obstacles in \( S' \). Observe that \( s, t \) must lie in different connected regions \( R_s, R_t \) of \( \mathbb{R}^2 \setminus \mathcal{R}(S') \) or else the set \( S' \) would not separate them. At least one of \( R_s \) or \( R_t \) must be bounded, wolog assume it is \( R_s \). Let \( \gamma' \) be the simple closed curve that is the common boundary of \( \mathcal{R}(S') \) and \( R_s \). We have that \( \gamma' \) encloses \( s \) but not \( t \) and therefore separates \( s \) from \( t \). Using first statement of Fact 2, we obtain that \( \gamma' \) crosses the reference curve \( \pi \) an odd number of times. Observe that the curve \( \gamma' \) consists of multiple sections \( \alpha'_i \to \alpha'_2 \to \cdots \to \alpha'_{i_{\text{ref}}} \), where each curve \( \alpha'_i \) is part of the boundary of some obstacle \( S_i \). For each of these curves \( \alpha'_i \), we add a detour to and back from the reference point \( \text{ref}(S_i) \) of the obstacle it belongs. Specifically, let \( q_i \) be an arbitrary point on the curve \( \alpha'_i \) and let \( \alpha'_{i_{\text{ref}}}, \alpha'_i \) be the portion of \( \alpha'_i \) before and after \( q_i \), respectively. We add the detour curve \( \delta_i = q_i \to \text{ref}(S_i) \to q_i \) ensuring that it always stays within the obstacle \( S_i \), which is possible because the obstacles are connected. (Same as before the curve \( \delta_i \) can be chosen to be transverse with \( \pi \) by considering the corresponding walk in graph of \( \text{Arr}(S_i \cup \pi) \).) Let \( \alpha_1 = \alpha'_{i_{\text{ref}}} \to \delta_i \to \alpha'_i \) be the curve obtained by adding detour \( \delta_i \) to \( \alpha'_i \). Let \( \gamma = \alpha_1 \to \alpha_2 \to \cdots \to \alpha_\ell \alpha_\ell \to \cdots \to \alpha_1 \) be the closed curve obtained by adding these detours to \( \gamma' \). Note that \( \gamma \) is not necessarily simple as the detour curves may intersect each other. Every detour \( \delta_i \) consists of identical copies of two curves, so it crosses the reference curve \( \pi \) an even number of times. Since \( \gamma' \) crosses \( \pi \) an odd number of times, the curve \( \gamma \) also crosses \( \pi \) an odd number of times. (See also Figure 1.) Observe that \( \gamma \) and \( \gamma' \) are transverse with \( \pi \) because intersections of \( \pi \) and obstacle boundaries are transverse and the detour curves \( \delta_i \) are chosen to be transverse with \( \pi \).

\[ \textbf{Figure 1} \] (a) The curve \( \gamma' \) shown shaded in blue is the common boundary of \( \mathcal{R}(S') \) and region \( R_s \). (b) Adding detours \( \delta_i \) to obtain curve \( \gamma \). (c) Labeled Intersection graph \( G_S \) of obstacles.

\[ \text{CVIT 2016} \]
We will now translate the curve $\gamma$ to a walk in the labeled intersection graph $G_S$. Specifically, consider the section of $\gamma$ between two consecutive detours: $\gamma_{i,i+1} = \text{ref}(S_i) \rightarrow q_i \rightarrow q_{i+1} \rightarrow \text{ref}(S_{i+1})$. Therefore the obstacles $S_i, S_{i+1}$ must intersect and we have a curve $\gamma_{i,i+1}$ connecting their reference points contained in the region $S_i \cup S_{i+1}$ that also intersects the reference curve $\pi$ an odd (resp. even) number of times. By construction, $G_S$ must contain an edge $e_{i,i+1}$ with label 1 (resp. 0). By replacing all these sections of $\gamma$ with the corresponding edges of $G_S$, we obtain an odd-labeled closed walk $W$ in $G_S$. Of all the odd-labeled closed sub-walks of $W$, we select one that is inclusion minimal. This gives a simple odd-labeled cycle in $G_S[S']$.

$(\Leftarrow)$ The reverse direction is relatively simpler. Given an odd-labeled cycle in $G_S[S']$, we obtain a closed curve $\gamma$ in the plane contained in region $R(S')$ as follows. For every edge $e_i = (S,S')$ of the cycle with label $\text{lab}(e_i)$, we consider the curve $\gamma_i$ that connects the reference points $\text{ref}(S)$ and $\text{ref}(S')$ contained in $S \cup S'$ and crosses the reference curve $\pi$ consistent with $\text{lab}(e_i)$. Moreover $\gamma_i$ needs to be transverse with $\pi$. Such a curve exists by construction of $G_S$. Combining these curves $\gamma_i$ in order gives us a closed curve $\gamma$ in the plane that crosses $\pi$ an odd number of times. Although this curve may be self-intersecting, from second statement of Fact 2, we have that $\gamma$ separates $s$ and $t$.

The construction of the graph $G_S$, together with Lemma 6 prove Theorem 1.

2-Points-separation as Shortest Odd Cycle in $G_S$. From Lemma 6, it follows that a minimum set of obstacles that separates $s$ from $t$ corresponds to an odd-labeled cycle in $G_S$ with fewest vertices. This readily gives a polytime algorithm for 2-Points-separation. In particular, for a fixed starting vertex, we can compute the shortest odd cycle in $G_S$ in $O(|S|^2)$ time by the following well-known technique. Consider an unlabeled auxiliary graph $G'$ with vertex set is $S \times \{0,1\}$. For every edge $e = (S,S')$ of $G_S$, we add edges $\{(S,0),(S',0)\}$ and $\{(S,1),(S',1)\}$ if $\text{lab}(e) = 0$. Otherwise, we add the edges $\{(S,0),(S',1)\}$ and $\{(S,1),(S',0)\}$. The shortest odd cycle containing a fixed vertex $S$ is the shortest path in $G'$ between vertices $(S,0)$ and $(S,1)$. Repeating over all starting vertices gives the shortest odd cycle in $G_S$. This can be easily extended for the node-weighted case which gives us the following useful lemma that also yields a polynomial time algorithm for 2-Points-separation, reproving a result of Cabello and Giannopoulos [6].

Lemma 7. There exists a polynomial time algorithm for computing a minimum weight labeled odd cycle in the graph $G_S$.

Next we prove one more structural property of labeled intersection graph $G_S$ that will be useful later. We define a (labeled) spanning tree $T$ of a connected labeled multi-graph $G_S$ to be a subgraph of $G_S$ that is a tree and connects all vertices in $S$. An edge $e = (u,v) \in G_S$ is a tree edge if $(u,v) \in T$, otherwise it is called a non-tree edge.

Lemma 8. Let $G_S$ be a connected labeled intersection graph and $T$ be a spanning tree of $G_S$. If $G_S$ contains an odd labeled cycle, then it also contains an odd labeled cycle with exactly one non-tree edge.

Proof. Let $C$ be an odd cycle in $G_S$ that contains fewest non-tree edges. If $C$ consists of exactly one non-tree edge, we are done. Otherwise, $C$ contains more than one non-tree edge. Let $e = (u,v) \in C$ be a non-tree edge and $C' \subset C$ be the remainder of $C$ without the edge $e$. Since $C$ is odd labeled, we must have $\text{lab}(C') \neq \text{lab}(e)$.

Let $\pi_{uv}$ be the unique path connecting $u,v$ in $T$. This gives us a path $\pi_{uv}$ with label $\text{lab}(\pi_{uv})$. Recall that $\text{lab}(C') \neq \text{lab}(e)$. We have two cases. (i) If $\text{lab}(\pi_{uv}) \neq \text{lab}(e)$, then
we obtain an odd labeled cycle \( \pi_{uv} \oplus e \) that has one non-tree edge, namely \( e \), and we are done. (ii) Otherwise, \( \text{lab}(\pi_{uv}) = \text{lab}(e) \neq \text{lab}(C') \). This gives us an odd labeled closed walk \( W^* = \pi_{uv} \oplus C' \) which contains one less non-tree edge than \( C \). Let \( C^* \subseteq W^* \) be an odd-labeled inclusion minimal closed sub-walk of \( W^* \) (one such \( C^* \) always exists). Therefore, \( C^* \) is an odd-labeled cycle in \( G_S \) that has fewer non-tree edges than \( C \). But \( C \) was chosen to be an odd labeled cycle with fewest non-tree edges, a contradiction.

The above lemma also gives a simple \( O(S^2) \) algorithm to detect whether there exists an odd label cycle in \( G_S \). Specifically, consider an arbitrary spanning tree of \( T \) of \( G_S \) and for each edge not in \( T \), compare its label with the label of the path connecting its endpoints in \( T \).

\[ \textbf{Lemma 9.} \quad \text{Given a labeled graph } G_S, \text{ there exists an } O(S^2) \text{ time algorithm to detect whether } G_S \text{ contains an odd labeled cycle.} \]

\section{Application to Obstacle-removal}

We will show how to cast \textsc{Obstacle-removal} as a Labeled Odd Cycle Transversal problem on the graph \( G_S \). Recall that in \textsc{Obstacle-removal} problem, we want to remove a set \( S_d \subseteq S \) of obstacles from the input so that \( s \) and \( t \) are connected in \( S \setminus S_d \). Equivalently, we want to select a subset \( S_d \) of obstacles such that the complement set \( S \setminus S_d \) does not separate \( s \) and \( t \). From Lemma 6, it follows that the obstacles \( S \setminus S_d \) do not separate \( s \) and \( t \) if and only if \( G_S[S \setminus S_d] \) does not contain an odd labeled cycle. This gives us the following important lemma.

\[ \textbf{Lemma 10.} \quad \text{A set of obstacles } S_d \subseteq S \text{ is a solution to } \textsc{Obstacle-removal} \text{ if and only if the set of vertices } S_d \text{ is a solution to } \textit{Odd Cycle Transversal} \text{ of } G_S. \]

This allows us to apply the set of existing results for \textit{Odd Cycle Transversal} to obstacle removal problems. In particular, this readily gives an improved algorithm for \textsc{Obstacle-removal} when parameterized by the solution size (number of removed obstacles). Let \( G^+_S \) denote the graph \( G_S \) where every edge \( e \) with \( \text{lab}(e) = 0 \) is subdivided. Clearly an odd-labeled cycle in \( G_S \) has odd length in \( G^+_S \) and vice versa. Applying the FPT algorithm for \textit{Odd Cycle Transversal} from [22] on the graph \( G^+_S \) gives us the following result.

\[ \textbf{Theorem 11.} \quad \text{There exists a } 2.3146^k n^{O(1)} \text{ algorithm for } \textsc{Obstacle-removal} \text{ parameterized by } k, \text{ the number of removed obstacles.} \]

This also immediately gives us an \( O(\sqrt{\log n}) \) approximation for \textsc{Obstacle-removal} by using the best known \( O(\sqrt{\log n}) \)-approximation [1] for on the graph \( G^+_S \). Observe that instances of obstacle removal are special cases of odd cycle transversal, specifically where the graph \( G_S \) is an intersection graph of obstacles. By applying known results on small diameter decomposition of region intersection graphs, Kumar et al. [25] obtained a constant factor approximation for \textsc{Obstacle-removal}. In the next section we present an alternative constant factor approximation algorithm. Although our algorithm follows a similar high level approach of using small diameter decomposition of \( G_S \), we give an alternative proof of the approximation bound which significantly simplifies the arguments of [25].

\section{Constant Approximation for Obstacle-removal}

Our algorithm is based on formulating and rounding a standard LP for labeled odd cycle transversal on labeled intersection graph \( G_S \). Let \( 0 \leq x_i \leq 1 \) be an indicator variable that
denotes whether obstacle $S_i$ is included to the solution or not. The LP formulation which will be referred as Hit-odd-cycles-LP can be written as follows:

$$\min \sum_{S_i \in S} x_i$$

subject to:

$$\sum_{S_j \in C} x_j \geq 1 \quad \text{for all odd-labeled cycles } C \in G_S$$

Although this LP has exponentially many constraints, it can be solved in polynomial time using ellipsoid method with the polynomial time algorithm for minimum weight odd cycle in $G_S$ (Lemma 7) as separation oracle. The next step is to round the fractional solution $\hat{x} = x_1, x_2, \ldots, x_n$ obtained from solving the Hit-odd-cycles-LP. We will need some background on small diameter decomposition of graphs.

**Small Diameter Decomposition** Given a graph $G = (V, E)$ and a distance function $d : V \to \mathbb{R}^+$ associated with each vertex, we can define the distance of each edge as $d(e) = d(v) + d(w)$ for every edge $e = (v, w) \in E$. We can then extend the distance function to any pair of vertices $d(u, v)$ as the shortest path distance between $u$ and $v$ in the edge-weighted graph with distance values of edges as edge weights. We use the following result of Lee [21] for the special case of region intersection graph over planar graphs.

**Lemma 12.** Let $G = (V, E)$ be a node-weighted intersection graph of connected regions in the plane, then there exists a set $X \subseteq V$ of $|X| = O(1/\Delta) \cdot \sum d(v)$ vertices such that the diameter of $G - X$ is at most $\Delta$ in the metric $d$. Moreover, such a set $X$ can be computed in polynomial time.

For the sake of convenience, we assume that $G_S$ does not contain an obstacle $S_i$ with a self-loop, because if so, we must always include $S_i$ to the solution. Let $G^*_S$ be the underlying unlabeled graph obtained by removing labels and multi-edges from $G_S$. Since $G^*_S$ is simply the intersection graph of connected regions in the plane, it is easy to show that $G^*_S$ is a region intersection graph over a planar graph (See also Lemma 4.1 [25] for more details.)

**(Algorithm: Hit-Odd-Cycles)** With small diameter decomposition for $G^*_S$ in place, the rounding algorithm is really simple.

- Assign distance values to remaining vertices of $G^*_S = (S \setminus S_0, E)$ as $d(S_i) = x_i$, where $x_i$ is the fractional solution obtained from solving Hit-odd-cycles-LP.
- Apply Lemma 12 on graph $G^*_S$ with diameter $\Delta = 1/2$. Return the set of vertices $X$ obtained from applying the lemma as solution.

It remains to show that the set $X \subseteq S$ returned above indeed hits all the odd labeled cycles in $G_S$. Define a ball $B(c, R) = \{v \in V : d(c, v) < R - d(v)/2\}$ with center $c$, radius $R$ and distance metric $d$ defined before. Intuitively, $B(c, R)$ consists of the vertices that lie strictly inside the radius $R$ ball drawn with $c$ as center.

**Lemma 13.** The set $X$ returned by algorithm Hit-Odd-Cycles hits all odd labeled cycles in $G_S$.

**Proof.** The proof is by contradiction. Let $C$ be an odd labeled cycle such that $C \cap X = \emptyset$. Then $C$ must be contained in a single connected $\kappa$ component of $G_S - X$. Let $v_1$ be an arbitrary vertex of $C$ and consider a ball $B = B(v_1, 1/2)$ of radius $1/2$ centered at $v_1$. We
have \( \kappa \subseteq B \) due to the choice of diameter \( \Delta \). Consider the shortest path tree \( T \) of ball \( B \) rooted at \( v_1 \) using the distance function \( d(e) \) in the unlabeled graph \( G'_S \). For every edge \((u,v) \in T \) assign the label \( \text{lab}(e) \) of \( e = (u,v) \in G_S \). If multiple labeled edges exist between \( u \) and \( v \), choose one arbitrarily.

Now consider the induced subgraph \( G'_S = G_S[B] \) which is a connected labeled intersection graph of obstacles in the ball \( B \). Moreover, \( T \) is a spanning tree of \( G'_S \), and \( G'_S \) contains an odd-labeled cycle because \( \kappa \subseteq G'_S \). Applying Lemma 8 gives us an odd-labeled cycle \( C \in G'_S \) that contains exactly one edge \( e \notin T \). The cost of this cycle is \( \text{cost}(C) < 1/2 + 1/2 = 1 \). This contradicts the constraint of \textsc{Hrt-Odd-Cycle-LP} corresponding to \( C \).

We conclude with the main result for this section.

\textbf{Theorem 14.} There exists a polynomial time constant factor approximation algorithm for \textsc{Obstacle-Removal}.

\section{A Simple Algorithm for Generalized Points-separation}

So far, we have focused on separating a pair of points \( s,t \) in the plane. In this section, we consider the more general problem where we are given a set \( S \) of \( n \) obstacles, a set of points \( A \) and a set and \( P = \{(s_1,t_1),\ldots,(s_p,t_p)\} \) of \( p \) pairs of points in \( A \) which we want to separate. First we show how to extend the labeled intersecting graph \( G_S \) to \( p \) source-destination pairs and that the optimal solution subgraph \( G_S[S_{OPT}] \) exhibits a 'nice' structure. Then we exploit this structure to obtain an \( 2^{O(p)}n^{O(p)} \) exact algorithm for \textsc{Generalized Points-separation}. Since \( p = O(k^2) \), this algorithm runs in polynomial time for any fixed \( k \), resolving an open question of \cite{6}. Using a more sophisticated approach, we later show how to improve the running time to \( 2^{O(p)}O^{O(k)} \).

Recall the construction of the labeled intersection graph \( G_S \) for a single point pair \( (s,t) \) from Section 3. The label \( \text{lab}(e) \in \{0,1\} \) of each edge \( e \in G_S \) denotes the parity of edge \( e \) with respect to reference curve \( \pi \) connecting \( s \) and \( t \). As we generalize the graph \( G_S = (S,E) \) to \( p \) point pairs, we extend the label function \( \text{lab} : E \rightarrow \{0,1\}^p \) as a \( p \)-bit binary string that denotes the parity with respect to reference curve \( \pi_i \) connecting \( s_i \) and \( t_i \) for all \( i \in [p] \). We will use \( \text{lab}_i(e) \) to denote the \( i \)-th bit of \( \text{lab}(e) \).

**Generalized Label Intersection Graph:**

- For each \( \{(s_i,t_i) \in P \) and each \( S \in \mathcal{S} \) that contains at least one of \( s_i \) or \( t_i \), we add a self loop \( e \) on \( S \) with \( \text{lab}_i(e) = 1 \) and \( \text{lab}_j(e) = 0 \) for all \( j \neq i \).
- For every pair of intersecting obstacles \( S,S' \) and a \( p \)-bit string \( \ell \in \{0,1\}^p \):
  - Let \( \Pi = \{\pi_i \mid s_i,t_i \notin S \cup S'\} \) be the set of reference curves that do not have endpoints in \( S \cup S' \).
  - We add an edge \( e = (S,S') \) with \( \text{lab}(e) = \ell \) if there exists a plane curve connecting \( \text{ref}(S) \) and \( \text{ref}(S') \) contained in \( S \cup S' \) that crosses all reference curves \( \pi_i \in \Pi \) with parity consistent with label \( \ell \). That is, the curve crosses \( \pi_i \) and odd (resp. even) number of times if \( i \)-th bit of \( \ell \) is 1 (resp. zero).

Similar to the one pair case, we can build an unlabeled graph \( G' \) with vertex set \( S \times \{0,1\}^p \) and edges between them based on the arrangement \( \text{Arr}(S \cup S' \cup \bigcup \pi_i) \). Using this graph, we can obtain the following lemma. The proof is the same as that of Lemma 25, with \( p \) bit labels instead of \( k \) bit labels.

\textbf{Lemma 15.} The generalized labeled graph \( G_S \) with \( p \)-bit labels can be constructed in \( 2^{O(p)}O^{O(1)} \) time.
Suppose we define $G_S(i)$ to be the image of $G_S$ induced by the labeling $\text{lab}_i : E \rightarrow \{0,1\}$. Specifically, we obtain $G_S(i)$ from $G_S$ by replacing label of each edge by the $i$-th bit $\text{lab}_i(e)$, followed by removing parallel edges that have the same label. Observe that $G_S(i)$ is precisely the graph obtained by applying algorithm from Section 3 with reference curve $\pi_i$.

We say that a subgraph $G'_S \subseteq G_S$ is well-behaved if $G'_S(i)$ contains an odd labeled cycle for all $i \in [p]$. We have the following lemma that can be obtained by applying Lemma 6 for every pair $(s_i,t_i) \in P$.

Lemma 16. A set of obstacles $S' \subseteq S$ separate all point pairs in $P$ iff $G_S([S'])$ is well-behaved.

We will prove the following important property of well-behaved subgraphs of $G_S$.

Lemma 17. Let $G \subseteq G_S$ be an inclusion minimal well-behaved subgraph of $G_S$. Then there exists a set $V_c \subseteq V(G)$ of connector vertices such that $G$ consists of the vertex set $V_c$ and a set of $K$ chains (path of degree 2 vertices) with endpoints in $V_c$. Moreover, $|V_c| \leq 4p$ and $|K| \leq 5p$.

Proof. Since $G$ is inclusion minimal well-behaved subgraph, it does not contain a proper subgraph that is also well-behaved. Therefore, $G$ does not contain a vertex of degree at most 1 because such vertices and edges adjacent to them cannot be part of any cycle. Suppose $G$ has $r$ connected components $C_1, \ldots, C_r$. We fix a spanning tree $T_j$ of $C_j$ for each $j \in [r]$. We construct the set $V_c$ by including every vertex of degree three or more to $V_c$. The components $C_j$ that do not contain a vertex of degree three must be a simple cycle because $G$ does not have degree-1 vertices. For every such $C_j$, we include vertices adjacent to the only non-tree edge of $C_j$. It is easy to verify that $G$ consists of $K$ chains connecting vertices in $V_c$.

Let $E_{0}$ be the set of non-tree edges, that are edges not in $T_j$ for some $j \in [r]$. We claim that $|E_{0}| \leq p$. Since $G$ is well-behaved, $G(i)$ consists an odd-labeled cycle for all $i \in [p]$. Using Lemma 8, and the spanning tree $T_j$ of the component containing that odd labeled cycle, we can transform into an odd-labeled cycle that uses at most one non-tree edge. Repeating this for all pairs, we can use at most $p$ edges from $E_{0}$. If $|E_{0}| > p$, then we would have a proper subgraph of $G$ with at most $p$ edges that is also well-behaved, which is not possible because $G$ was chosen to be inclusion minimal. Therefore $|E_{0}| \leq p$.

The graph $G$ only contains vertices of degree 2 or higher, hence each leaf node of the trees $T_1, \ldots, T_r$ must be adjacent to some edge in $E_{0}$. Therefore, the number of leaf nodes is at most $2p$, and so the number of nodes of degree three or above in $T_1, \ldots, T_r$ is also at most $2p$. Observe that the vertices in $V_c$ are either adjacent to some edge in $E_{0}$ or have degree three or more in some tree $T_j$. The number of both these type of vertices is at most $2p$, which gives us $|V_c| \leq 4p$. Finally, we bound $|K|$, the number of chains. Note that each edge of $G$ belongs to exactly one chain in $K$. Therefore, the number of chains containing at least one edge in $E_{0}$ is at most $p$, because $|E_{0}| \leq p$. All the other chains that do not have any edge in $E_{0}$, are contained in the trees $T_1, \ldots, T_r$. It follows that these chains do not form any cycle, and thus their number is less than $|V_c|$. This gives us $|K| \leq 5p$.

It is easy to see that if $S' \subseteq S$ is an optimal set of obstacles separating all pairs in $P$, then there exists an inclusion minimal well-behaved subgraph $G$ of $G_S[S']$ that satisfies the property of Lemma 17. Observe that the $K$ chains of graph $G$ are vertex disjoint, so for every chain $K_{\ell}$ connecting vertices $s_i, s_j \in V_c$ that has $\text{lab}(K\ell) = \ell$, an optimal solution will always choose the walk in $G_S$ that has label $\ell$ and has fewest vertices. To that end, we will need the following simple lemma which is a generalization of algorithm to compute shortest odd cycle in $G_S$ with 1-bit labels.
Lemma 18. Given a labeled graph \( G_S = (S, E) \) with labeling \( \text{lab} : E \rightarrow \{0,1\}^p \), the shortest walk between any pair of vertices \( S_i, S_j \) with a fixed label \( \ell \in \{0,1\}^p \) can be computed in \( 2^{O(p)}n^{O(1)} \) time.

Algorithm: Separate-Point-Pairs
1. For every pair of vertices \( S_i, S_j \in S \) and every label \( \ell \in \{0,1\}^p \), precompute the shortest walk connecting \( S_i, S_j \) with label \( \ell \) in \( G_S \) using Lemma 18.
2. For all possible sets \( V_c \subseteq S \) and ways of connecting \( V_c \) by \( K \) chains:
   a. Let \( G \subseteq G_S \) be the labeled graph consisting of vertices \( V_c \) and chains \( K_I \in K \) replaced by shortest walk between endpoints of \( K_I \) with label \( \text{lab}(K_I) \), already computed in Step 1.
   b. Check if the graph \( G \) is well-behaved. If so, add its vertices as one candidate solution.
3. Return the candidate vertex set with smallest size as solution.

Precomputing labeled shortest walks in Step 1 takes at most \( 2^{O(p)}n^{O(p)} \) time. The total number of candidate graphs \( G \) is \( n^{O(1)} \cdot p^{O(p)} \cdot 2^{O(p^2)} \), and checking if it is well behaved can be done in \( n^{O(1)} \) time. We have the following result.

Theorem 19. Generalized Points-separation for connected obstacles in the plane can be solved in \( 2^{O(p)}n^{O(p)} \) time, where \( n \) is the number of obstacle and \( p \) is the number of point-pairs to be separated.

Corollary 20. Point-Separation for connected obstacles in the plane can be solved in \( 2^{O(k^2)}n^{O(k^2)} \) time, where \( n \) is the number of obstacles and \( k \) is the number of points. This is polynomial in \( n \) for every fixed \( k \).

6 A Faster Algorithm for Generalized Points-separation

Recall that the labeled graph \( G_S \) constructed in the previous section consisted of labels that are \( p \)-bit binary strings. As a result, the running time has a dependence of \( n^{O(k)} \) which in worst case could be \( n^{O(k^2)} \), for example, in the case of Points-separation when \( P \) consists of all point pairs. In this section, we describe an alternative approach that builds a labeled intersection graph whose labels are \( k \)-bit strings. Using this graph and the notion of parity partitions, we obtain an \( 2^{O(p)}n^{O(k)} \) algorithm for Generalized Points-separation which gets rid of the \( n^{O(k^2)} \) dependence for Points-separation. The construction of graph \( G_S \) is almost the same as before, except that now we choose the reference curves \( \pi_i \) differently. In particular, let \( A = \{a_1, a_2, \ldots, a_k\} \) be the set of points and \( P \) be a set of pairs \((a_i, a_j)\) of points we want to separate. We pick an arbitrary point \( o \) in the plane, and for each \( i \in [k] \), we fix a plane curve with endpoints \( a_i \) and \( o \) as the reference curve \( \pi_i \). For an edge \( e \), the parity of crossing with respect to \( \pi_i \) defines the \( i \)-th bit of \( \text{lab}(e) \). The graph \( G_S \) constructed in this fashion has \( k \)-bit labels and will be referred as \( k \)-labeled graph.

Definition 21 (labeled graphs). For an integer \( k \geq 1 \), a \( k \)-labeled graph is a multi-graph \( G = (V, E) \) and where each edge \( e \in E \) has a label \( \text{lab}(e) \in \{0,1\}^k \) which is \( k \)-bit binary string; we use \( \text{lab}_i(e) \) to denote the \( i \)-th bit of \( \text{lab}(e) \) for \( i \in [k] \).

A \( P \)-separator refers to a subset \( S' \subseteq S \) that separates all point-pairs \((a_i, a_j)\) for \((i, j) \in P \). Our goal is to find a \( P \)-separator with the minimum size. To this end, we first introduce the notion of labeled graphs and some related concepts.
Let $G$ be a $k$-labeled graph. For a cycle (or a path) $\gamma$ in $G$ with edge sequence $(e_1, \ldots, e_r)$, we define $\text{parity}(\gamma) = \bigoplus_{i=1}^{r} \text{lab}(e_i)$ and denote by $\text{parity}_i(\gamma)$ the $i$-th bit of $\text{parity}(\gamma)$ for $i \in [k]$. Here the notation “$\oplus$” denotes the bitwise XOR operation for binary strings. Also, we define $\Phi(\gamma)$ as the partition of $[k]$ consisting of two parts $I_0 = \{i : \text{parity}_i(\gamma) = 0\}$ and $I_1 = \{i : \text{parity}_i(\gamma) = 1\}$. Next, we define an important notion called $\text{parity partition}$.

**Definition 22 (parity partition).** Let $G$ be a $k$-labeled graph. The $\text{parity partition}$ induced by $G$, denoted by $\Phi_G$, is the partition of $[k]$ defined as $\Phi_G = \bigoplus_{e \in E_G} \Phi(\gamma)$. In other words, $i, j \in [k]$ belong to the same part of $\Phi_G$ iff $\text{parity}_i(\gamma) = \text{parity}_j(\gamma)$ for every cycle $\gamma$ in $G$.

The following two lemmas state some basic properties of the $\text{parity partition}$.

**Lemma 23.** Let $G$ be a $k$-labeled graph, and $C_1, \ldots, C_r$ be the connected components of $G$ each of which is regarded as a $k$-labeled graph. Then $\Phi_G = \bigcup_{i=1}^{r} \Phi(C_i)$.

**Proof.** Note that a cycle in $G$ must be contained in some connected component $C_t$ for $t \in [r]$, i.e., $\Gamma_G = \bigcup_{t=1}^{r} \Gamma_t$. Thus, $\Phi_G = \bigcup_{\gamma \in \Gamma_G} \Phi(\gamma) = \bigcup_{t=1}^{r} (\bigcup_{\gamma \in \Gamma_{C_t}} \Phi(\gamma)) = \bigcup_{i=1}^{r} \Phi(C_i)$. △

**Lemma 24.** Let $G$ be a connected $k$-labeled graph, and $T$ be a spanning tree of $G$. Let $E_0$ be the edges of $G$ that are not in $T$. Then $\Phi_G = \bigcup_{e \in E_0} \Phi(\gamma_e)$, where $\gamma_e$ is the cycle in $G$ consists of the edge $e$ and the (unique) simple path between the two endpoints of $e$ in $T$.

**Proof.** The proof is similar to and more general form of Lemma 8. It is clear that $\Phi_G \subseteq \bigcup_{e \in E_0} \Phi(\gamma_e)$ because $\gamma_e \in I_G$ for all $e \in E_0$. To show $\Phi_G \supseteq \bigcup_{e \in E_0} \Phi(\gamma_e)$, we use contradiction. Assume $\Phi_G \not\subseteq \bigcup_{e \in E_0} \Phi(\gamma_e)$. Then there exist $i, j \in [k]$ which belong to different parts in $\Phi_G$ but belong to the same part in $\bigcup_{e \in E_0} \Phi(\gamma_e)$, i.e., $\text{parity}_i(\gamma_e) = \text{parity}_j(\gamma_e)$ for all $e \in E_0$. Since $i$ and $j$ belong to different parts in $\Phi_G$, we have $\text{parity}_i(\gamma) \neq \text{parity}_j(\gamma)$ for some $\gamma \in I_G$. Let $\gamma^* \in I_G$ be the cycle satisfying $\text{parity}_i(\gamma^*) \neq \text{parity}_j(\gamma^*)$ that contains the smallest number of edges in $E_0$. Note that $\gamma^*$ contains at least one edge in $E_0$, for otherwise $\gamma^*$ is a cycle in the tree $T$ and hence $\text{parity}_i(\gamma^*) = \text{parity}_j(\gamma^*) = 0$ (simply because a cycle in a tree goes through each edge even number of times). Let $e = (u, v)$ be an edge of $\gamma^*$ that is in $E_0$. We create a new cycle $\gamma'$ from $\gamma^*$ by replacing the edge $e$ in $\gamma^*$ with the (unique) simple path $\pi_{uv}$ between $u$ and $v$ in $T$. Recall that $\text{parity}_i(\gamma_e) = \text{parity}_j(\gamma_e)$. Since $\text{parity}_i(\gamma_e) = \text{lab}_i(e) \circ \text{parity}_i(\pi_{uv})$ and $\text{parity}_j(\gamma_e) = \text{lab}_j(e) \circ \text{parity}_j(\pi_{uv})$, we have $\text{lab}_i(e) \circ \text{parity}_i(\pi_{uv}) = \text{lab}_j(e) \circ \text{parity}_j(\pi_{uv})$. Because $\text{parity}_i(\gamma^*) \neq \text{parity}_j(\gamma^*)$, we further have

$$
\text{parity}_i(\gamma^*) = \text{parity}_i(\gamma') \circ (\text{lab}_i(e) \circ \text{parity}_i(\pi_{uv}))
= \text{parity}_i(\gamma') \circ (\text{lab}_j(e) \circ \text{parity}_j(\pi_{uv}))
\neq \text{parity}_j(\gamma^*) \circ (\text{lab}_j(e) \circ \text{parity}_j(\pi_{uv})) = \text{parity}_j(\gamma').
$$

However, this is impossible because $\gamma'$ has fewer edges in $E_0$ than $\gamma^*$ and $\gamma^*$ is the cycle satisfying $\text{parity}_i(\gamma^*) \neq \text{parity}_j(\gamma^*)$ that contains the smallest number of edges in $E_0$. Therefore, $\Phi_G \supseteq \bigcup_{e \in E_0} \Phi(\gamma_e)$ and hence $\Phi_G = \bigcup_{e \in E_0} \Phi(\gamma_e)$. △

Now we are ready to describe our algorithm. The first step of our algorithm is to build a $k$-labeled graph $G_S$ for the obstacle set $S$. The vertices of $G_S$ are the obstacles in $S$, and the labeled edges of $G_S$ “encode” enough information for determining whether a subset of $S$ is a $P$-separator. Once we obtain $G_S$, we can totally forget the input obstacles and points, and the rest of our algorithm will work on $G_S$ only.

We build $G_S$ as follows. For each $S \in S$, we pick a reference point $\text{ref}(S)$ inside the obstacle $S$. Let $\text{Arr}(S)$ denote the arrangement induced by the boundaries of the obstacles in
$S$, and $|\text{Arr}(S)|$ be the complexity of $\text{Arr}(S)$. By assumption, $|\text{Arr}(S)| = n^{O(1)}$. We pick an arbitrary point $o$ in the plane, and for each $i \in [k]$, we fix a plane curve $\pi_i$ with endpoints $a_i$ and $o$. We choose the curves $\pi_1, \ldots, \pi_k$ carefully such that including them does not increase the complexity of the arrangement $\text{Arr}(S)$ significantly. Specifically, we require the complexity of the arrangement induced by the boundaries of the obstacles in $S$ and these curves to be bounded by $k^{O(1)} \cdot |\text{Arr}(S)|$, which is clearly possible. As mentioned before, the vertices of $G_S$ are the obstacles in $S$. The edge set $E_{G_S}$ of $G_S$ is defined as follows. For each $i \in [k]$ and each $S \in S$ such that $a_i \in S$, we include in $E_{G_S}$ a self-loop $e$ on $S$ with $\text{lab}_r(e) = 1$ and $\text{lab}_v(e) = 0$ for all $i' \in [k] \setminus \{i\}$. For each pair $(S,S')$ of obstacles in $S$ and each $l \in \{0,1\}^k$, we include in $E_{G_S}$ an edge $e = (S,S')$ with $\text{lab}(e) = l$ if there exists a plane curve inside $S \cup S'$ with endpoints $\text{ref}(S)$ and $\text{ref}(S')$ which crosses $\pi_i$ an odd (resp., even) number of times for all $i \in [k]$ such that $a_i \notin S \cup S'$ and the $i$-th bit of $l$ is equal to 1 (resp., 0). The next lemma shows $G_S$ can be constructed in $2^{O(k)}n^{O(1)}$ time, as $|\text{Arr}(S)| = n^{O(1)}$.

Lemma 25. The $k$-labeled graph $G_S$ can be constructed in $2^{O(k)}n^{O(1)} \cdot |\text{Arr}(S)|$ time.

Proof. The self-loops of $G_S$ can be constructed in $O(kn)$ time by checking for $i \in [k]$ and $S \in S$ whether $a_i \in S$. For each pair $(S,S')$ of obstacles in $S$, we show how to compute the edges in $G_S$ between $S$ and $S'$ in $2^{O(k)} \cdot |\text{Arr}(S)|$ time. Let $K = \{i \in [k] : a_i \notin S \cup S'\}$; without loss of generality, assume $K = \{a_1, \ldots, a_j\}$. Denote by $\text{Arr}(S,S')$ the arrangement induced by the boundary of $S \cup S'$ and the curves $\pi_1, \ldots, \pi_j$, and define $\mathcal{F}$ as the set of faces of $\text{Arr}(S,S')$ that are contained in $S \cup S'$. See Figure 2 for an illustration of the arrangement $\text{Arr}(S,S')$. We say two faces $F,F' \in \mathcal{F}$ are adjacent if they share a common edge $\sigma(F,F')$ of $\text{Arr}(S,S')$. For two adjacent faces $F,F' \in \mathcal{F}$, we define $\theta(F,F') \in \{0,1\}^j$ by setting the $i$-th bit of $\theta(F,F')$ to be 1 for all $i \in [j]$ such that $\sigma(F,F')$ is a portion of $\pi_i$, and setting the other bits to be 0. We construct a (unlabeled and undirected) graph $G$ with vertex set $\mathcal{F} \times \{0,1\}^j$ as follows. For any two vertices $(F,l)$ and $(F',l')$ such that $F,F'$ are adjacent and $l \oplus l' = \theta(F,F')$, we connect them by an edge in $G$.

![Figure 2 An illustration of the arrangement $\text{Arr}(S,S')$. The grey area is $S \cup S'$. The set $\mathcal{F}$ consists of five faces $F_1, \ldots, F_5$.](image)

Let $F \in \mathcal{F}$ and $F' \in \mathcal{F}$ be the faces containing the reference points $\text{ref}(S)$ and $\text{ref}(S')$, respectively, and denote by $0 \in \{0,1\}^j$ the element with all bits 0. We claim that there is an edge $(S,S')$ in $G_S$ with label $l$ iff the vertices $(F,0)$ and $(F',l)$ are in the same connected component of $G$. To prove the claim, we first make a simple observation about the graph $G$ we constructed. Let $(F_1,l_1), \ldots, (F_m,l_m)$ be a path in $G$. From the construction of $G$, it is easy to see (by a simple induction on $m$) that any plane curve from a point in $F_1$ to a
point in $F_m$ that visits the faces $F_1, \ldots, F_m$ in order crosses $\pi_i$ an odd (resp., even) number of times for all $i \in [j]$ such that the $i$-th bit of $l_1 \oplus l_m$ is equal to 1 (resp., 0). Therefore, if there is a path in $G$ from $(F, 0)$ to $(F', l)$, then there exists a plane curve from $\text{ref}(S)$ to $\text{ref}(S')$ that crosses $\pi_i$ an odd (resp., even) number of times for all $i \in [j]$ such that the $i$-th bit of $l$ is equal to 1 (resp., 0), which implies that there is an edge $(S, S')$ in $G_S$ with label $l$. This proves the “if” part of the claim. To see the “only if” part, assume there is an edge $(S, S')$ in $G_S$ with label $l$. Then there exists a plane curve $\pi$ from $\text{ref}(S)$ to $\text{ref}(S')$ that crosses $\pi_i$ an odd (resp., even) number of times for all $i \in [j]$ such that the $i$-th bit of $l$ is equal to 1 (resp., 0). Let $F_1, \ldots, F_m$ be the sequence of faces visited by $\pi$ in order, where $F_1 = F$ and $F_m = F'$. Then there is a path $(F_1, l_1), \ldots, (F_m, l_m)$ in $G$ where $l_1 = 0$ and $l_t = l_{t-1} \oplus \theta(F_{t-1}, F_t)$ for $t \in [m] \setminus \{1\}$. By our above observation, we have $l_1 \oplus l_m = l$, which implies $l_m = l$. It follows that $(F, 0)$ and $(F', l)$ are in the same connected component of $G$.

By the above discussion, to compute the edges in $G_S$ between $S$ and $S'$, it suffices to compute the connected component $C$ of $G$ that contains the vertex $(F, 0)$: we have an edge $(S, S')$ in $G_S$ with label $l \in \{0, 1\}^k$ iff $(F', l') \in C$ where $l' \in \{0, 1\}^j$ consists of the first $j$-bits of $l$. The number of vertices and edges of $G$ is $2^{O(k)} \cdot |\text{Arr}(S)|$, by our assumption that the complexity of the arrangement induced by the boundaries of the obstacles in $S$ and the curves $\pi_1, \ldots, \pi_m$ is bounded by $k^{O(1)} \cdot |\text{Arr}(S)|$. Therefore, $C$ can be computed in $2^{O(k)} \cdot |\text{Arr}(S)|$ time. As a result, $G_S$ can be constructed in $2^{O(k)} n^{O(1)} \cdot |\text{Arr}(S)|$ time. ▶

We say a $k$-labeled graph $G$ is $P$-good if for all $(i, j) \in P$, $i$ and $j$ belong to different parts in $\Phi_G$. Note that if a subgraph of $G$ is $P$-good, then so is $G$. The following key lemma establishes a characterization of $P$-separators using $P$-goodness. Note that the notion of $P$-goodness is almost the same as that of well-behaved subgraphs from Lemma 16, except that it is defined using parity partitions.

Lemma 26. A subset $S' \subseteq S$ is a $P$-separator iff the induced subgraph $G_S[S']$ is $P$-good.

Proof. We first introduce some notations. For $(i, j) \in P$, denote by $\pi_{i,j}$ the plane curve with endpoints $a_i$ and $a_j$ obtained by concatenating the curves $\pi_i$ and $\pi_j$. For each edge $e = (S, S')$ of $G_S$ with $S \neq S'$, we fix a representative curve $\text{rep}(e)$ of $e$, which is a plane curve contained in $S \cup S'$ with endpoints $\text{ref}(S)$ and $\text{ref}(S')$ that crosses $\pi_i$ an odd (resp., even) number of times for all $i \in [k]$ such that $\text{lab}_i(e) = 1$ (resp., $\text{lab}_i(e) = 0$); such a curve exists by our construction of $G_S$.

To prove the “if” part, assume $G_S[S']$ is $P$-good. Let $(i, j) \in P$ be a pair and we want to show that $(a_i, a_j)$ is separated by $S'$. If $a_i \in \bigcup_{S \subseteq S'} S \cup a_j \in \bigcup_{S \subseteq S'} S$, we are done. So assume $a_i \notin \bigcup_{S \subseteq S'} S$ and $a_j \notin \bigcup_{S \subseteq S'} S$. Since $G_S[S']$ is $P$-good, there exists a cycle $c$ in $G_S[S']$ such that $\text{parity}_i(c) \neq \text{parity}_j(c)$. Without loss of generality, we assume $\text{parity}_i(c) = 0$ and $\text{parity}_j(c) = 1$. Also, we can assume that $c$ does not contain any self-loop edges; indeed, removing any self-loop edges from $c$ does not change $\text{parity}_i(c)$ and $\text{parity}_j(c)$ because $a_i \notin \bigcup_{S \subseteq S'} S$ and $a_j \notin \bigcup_{S \subseteq S'} S$ (hence the $i$-th and $j$-th bits of the label of any self-loop on a vertex $S \in S'$ are equal to 0). Suppose the vertex sequence of $c$ is $(S_0, \ldots, S_r)$ where $S_0 = S_t$ and the edge sequence of $c$ is $(e_1, \ldots, e_r)$ where $e_t = (S_{t-1}, S_t)$ for $t \in [r]$. We concatenate the representative curves $\text{rep}(e_1), \ldots, \text{rep}(e_r)$ to obtain a closed curve $\hat{c}$ in the plane. Because $\text{parity}_i(\hat{c}) = 0$ and $\text{parity}_j(\hat{c}) = 1$, $\pi_i$ crosses $\hat{c}$ an even number of times and $\pi_j$ crosses $\hat{c}$ an odd number of times. It follows that $\pi_{i,j} \cap \hat{c}$ an odd number of times. By the second statement of Fact 2, $\hat{c}$ separates $(a_i, a_j)$. Since $\text{rep}(e_t) \subseteq S_t \cup S_t$, we have $\hat{c} \subseteq \bigcup_{i=1}^{r} S_t \subseteq \bigcup_{S \subseteq S'} S$. Therefore, $S'$ separates $(a_i, a_j)$.

To prove the “only if” part, assume $S' \subseteq S$ is a $P$-separator, i.e., $S'$ separates all point-pairs $(a_i, a_j)$ for $(i, j) \in P$. We want to show that $i$ and $j$ belong to different parts in $\Phi_{G_S[S']}$.
for all \((i,j) \in P\), or equivalently, for each \((i,j) \in P\) there exists a cycle \(\gamma\) in \(G_{S}[S']\) such that 
\[
\text{parity}_i(\gamma) \neq \text{parity}_j(\gamma).
\]
Let \(U = \bigcup_{S \in S'} S\). We distinguish two cases: \(\{a_i, a_j\} \cap U \neq \emptyset\) and \(\{a_i, a_j\} \cap U = \emptyset\). In the case \(\{a_i, a_j\} \cap U \neq \emptyset\), we may assume \(a_i \in U\) without loss of generality. Then \(a_i \in S\) for some \(S \in S'\). Therefore, by our construction of the graph \(G_S\), there is a self-loop edge \(e = (S, S)\) with \(\text{lab}_i(e) = 1\) and \(\text{lab}_j(e) = 0\) for all \(i' \in [k] \setminus \{i\}\). The cycle \(\gamma\) consists of this single edge is a cycle in \(G_S[S']\) satisfying 
\[
\text{parity}_i(\gamma) = 1 \neq 0 = \text{parity}_j(\gamma).
\]
Now it suffices to consider the case \(\{a_i, a_j\} \cap U = \emptyset\). The boundary \(\partial U\) of \(U\) consists of arcs (each of which is a portion of the boundary of an obstacle in \(S'\)) and break points (each of which is an intersection point of the boundaries of two obstacles in \(S'\)). We can view \(\partial U\) as a planar graph \(G\) embedded in the plane, where the break points are vertices and the arcs are edges. Each face of (the embedding of) \(G\) is a connected component of \(\mathbb{R}^2 \setminus \partial U\), which is either contained in \(U\) (called in-faces) or outside \(U\) (called out-faces). Let \(F_i\) and \(F_j\) be the faces containing \(a_i\) and \(a_j\), respectively. Since \(\{a_i, a_j\} \cap U = \emptyset\), \(F_i\) and \(F_j\) are both out-faces. Furthermore, we have \(F_i \neq F_j\), for otherwise \(a_i, a_j \in F_i\) and there exists a plane curve inside the out-face \(F_i\) connecting \(a_i\) and \(a_j\), which contradicts the fact that \(S'\) separates \((a_i, a_j)\). Thus, there exists a simple cycle \(\hat{\gamma}\) in \(G\) (which corresponds to a simple closed curve in the plane) such that one of \(F_i\) and \(F_j\) is inside \(\hat{\gamma}\) and the other one is outside \(\hat{\gamma}\) (it is well-known that in a planar graph embedded in the plane, for any two distinct faces there exists a simple cycle in the graph such that one face is inside the cycle and the other is outside). Because \(a_i \in F_i\) and \(a_j \in F_j\), we know that \(\hat{\gamma}\) separates \((a_i, a_j)\) and hence \(\pi_{i,j}\) crosses \(\hat{\gamma}\) an odd number of times by the first statement of Fact 2. Let \(\sigma_1, \ldots, \sigma_r\) be the arcs of \(\hat{\gamma}\) given in the order along \(\hat{\gamma}\), and suppose they are contributed by the obstacles \(S_1, \ldots, S_r \in S'\), respectively (note that here \(S_1, \ldots, S_r\) need not be distinct). For convenience, we write \(\sigma_0 = \sigma_r\) and \(S_0 = S_r\). Let \(x_t\) be the connection point of the arcs \(\sigma_{t-1}\) and \(\sigma_t\) for \(t \in [r]\), then \(x_t \in S_{t-1} \cap S_t\). For each \(t \in [r]\), we fix a plane curve \(\tau_t\) inside the obstacle \(S_t\) with endpoints \(\text{ref}(S_t)\) and \(x_t\) (such a curve exists because \(S_t\) is connected). Again, we write \(\tau_0 = \tau_r\). See Figure 3 for an illustration of the arcs \(\sigma_1, \ldots, \sigma_r\), the points \(x_1, \ldots, x_r\), and the curves \(\tau_1, \ldots, \tau_r\). Now let \(\tau'_t\) be the plane curve with endpoints \(\text{ref}(S_{t-1})\) and \(\text{ref}(S_t)\) obtained by concatenating \(\tau_{t-1}\), \(\sigma_{t-1}\), and \(\tau_t\), and let \(l_t \in \{0, 1\}^k\) be the label whose \(i'\)-th bit is 0 (resp., 1) if \(\pi_{i'}\) crosses \(\tau'_t\) an even (resp., odd) number of times, for \(t \in [r]\). Note that \(\tau'_t \subseteq S_{t-1} \cup S_t\). Therefore, by our construction of \(G_S\), there should be an edge \(e_t = (S_{t-1}, S_t)\) with \(\text{lab}(e) = l_t\), for each \(t \in [r]\). Consider the cycle \(\gamma\) in \(G_S[S']\) with vertex sequence \((S_0, \ldots, S_r)\) and edge sequence \((e_1, \ldots, e_r)\). We claim that 
\[
\text{parity}_i(\gamma) \neq \text{parity}_j(\gamma).
\]
Let \( \gamma' \) be the closed plane curve obtained by concatenating the curves \( \tau'_1, \ldots, \tau'_r \). Observe that \( \gamma' \) consists of \( \gamma \) and two copies of \( \tau_1, \ldots, \tau_r \). It follows that for any plane curve \( \pi \), the parity of the number of times that \( \pi \) crosses \( \gamma' \) is equal to the parity of the number of times that \( \pi \) crosses \( \gamma \). In particular, \( \pi_{i,j} \) crosses \( \gamma' \) an odd number of times. Without loss of generality, we may assume that \( \pi_i \) crosses \( \gamma' \) an odd number of times and \( \pi_j \) crosses \( \gamma' \) an even number of times. Since \( \gamma' \) is the concatenation of \( \tau'_1, \ldots, \tau'_r \) and the parity of the number of times that \( \pi_i \) (resp., \( \pi_j \)) crosses \( \tau'_i \) is indicated by the \( i \)-th (resp., \( j \)-th) bit of \( t_i \), the \( i \)-th (resp., \( j \)-th) bit of \( \bigodot_{1}^{r} t_i \) is 1 (resp., 0). Because \( \text{parity}(\gamma) = \bigodot_{1}^{r} t_i \), we have \( \text{parity}_{i,j}(\gamma) \neq \text{parity}_{j,i}(\gamma) \).

**Definition 27.** Let \( G = (V_G, E_G) \) and \( H = (V_H, E_H) \) be two \( k \)-labeled graphs. A parity-preserving mapping (PPM) from \( H \) to \( G \) is a pair \( f = (f_V, f_E) \) consisting of two functions \( f_V : V_H \rightarrow V_G \) and \( f_E : E_H \rightarrow E_G \) such that for each edge \( e = (u, v) \in E_H \), \( f_E(e) \) is a path between \( f(u) \) and \( f(v) \) in \( G \) satisfying \( \text{parity}(f_E(e)) = \text{lab}(e) \). The cost of the PPM \( f \) is defined as \( \text{cost}(f) = |V_H| - |E_H| + \sum_{e \in E_H} |f_E(e)| \). The image of \( f \), denoted by \( \text{Im}(f) \), is the subgraph of \( G \) consisting of the vertices \( f_V(v) \) for \( v \in V_H \) and the vertices on the paths \( f_E(e) \) for \( e \in E_H \), and the edges on the paths \( f_E(e) \) for \( e \in E_H \).

**Fact 28.** For any PPM \( f \), the number of vertices of \( \text{Im}(f) \) is at most \( \text{cost}(f) \).

**Proof.** Let \( f = (f_V, f_E) \) be a PPM from \( H = (V_H, E_H) \) to \( G \). The number of vertices \( f_V(v) \) for \( v \in V_H \) is at most \( |V_H| \). The number of internal vertices on each path \( f_E(e) \) for \( e \in E_H \) is at most \( |f_E(e)| - 1 \). Note that a vertex of \( \text{Im}(f) \) is either \( f_V(v) \) for some \( v \in V_H \) or an internal vertex on the path \( f_E(e) \) for some \( e \in E_H \). Thus, the total number of vertices of \( \text{Im}(f) \) is at most \( |V_H| + \sum_{e \in E_H} (|f_E(e)| - 1) = |V_H| - |E_H| + \sum_{e \in E_H} |f_E(e)| = \text{cost}(f) \).

**Lemma 29.** Let \( H \) be a \( P \)-good \( k \)-labeled graph and \( f \) be a PPM from \( H \) to \( G_S \). Then \( \text{Im}(f) \) is also \( P \)-good. In particular, \( \text{cost}(f) \geq \text{opt} \).

**Proof.** To see \( \text{Im}(f) \) is \( P \)-good, what we want is that \( i \) and \( j \) belong to different parts of \( \Phi_{\text{Im}(f)} \) for all \( (i, j) \in P \). Consider a pair \( (i, j) \in P \). Since \( H \) is \( P \)-good, there exists a cycle \( \gamma \) in \( H \) such that \( \text{parity}_{i,j}(\gamma) \neq \text{parity}_{j,i}(\gamma) \). Let \( \gamma' \) be the image of \( \gamma \) under \( f \), which is a cycle in \( \text{Im}(f) \) obtained by replacing each vertex \( v \) of \( \gamma \) with \( f_V(v) \) and each edge \( e \) of \( \gamma \) with the path \( f_E(e) \). Because \( f \) is a PPM, we have \( \text{parity}(\gamma') = \text{parity}(\gamma) \). Therefore, \( \text{parity}_{i,j}(\gamma') \neq \text{parity}_{j,i}(\gamma') \). It follows that \( i \) and \( j \) belong to different parts of \( \Phi_{\text{Im}(f)} \), and hence \( \text{Im}(f) \) is \( P \)-good. To see \( \text{cost}(f) \geq \text{opt} \), let \( S' \subseteq S \) be the vertex set \( \text{Im}(f) \). Then \( \text{Im}(f) \) is a subgraph of \( G_S[S'] \), which implies \( G_S[S'] \) is also \( P \)-good by Lemma 26. \( S' \) is a \( P \)-separator, i.e., \( |S'| \geq \text{opt} \). Furthermore, by Fact 28, we have \( \text{cost}(f) \geq |S'| \geq \text{opt} \).

**Lemma 30.** There exists a \( P \)-good \( k \)-labeled graph \( H^* \) with at most \( 4k \) vertices and \( 5k \) edges and a PPM \( f^* \) from \( H^* \) to \( G_S \) such that \( \text{cost}(f^*) = \text{opt} \).

**Proof.** Let \( S_{\text{opt}} \subseteq S \) be a \( P \)-separator of the minimum size. By Lemma 26, the induced subgraph \( G_S[S_{\text{opt}}] \) is \( P \)-good. Let \( G \) be a minimal \( P \)-good subgraph of \( G_S[S_{\text{opt}}] \), that is, no proper subgraph of \( G \) is \( P \)-good. Note that \( G \) does not have degree-0 and degree-1 vertices, simply because deleting a degree-0 or degree-1 vertex (and its adjacent edge) from \( G \) does not change \( \Phi_G \). Suppose \( G \) has \( r \) connected components \( C_1, \ldots, C_r \). We fix a spanning tree \( T_i \) of \( C_i \) for each \( i \in [r] \). Let \( E_0 \) be the set of non-tree edges of \( G \), i.e., the edges not in \( T_1, \ldots, T_r \). We mark all vertices of \( G \) with degree at least 3. Furthermore, for each component \( C_i \) that has no vertex with degree at least 3 (which should be a simple cycle because \( G \) does not have degree-1 vertices), we mark a vertex of \( C_i \) that is adjacent to the (only) non-tree edge
of $C_t$. We notice that all unmarked vertices of $G$ are of degree 2 and each component $C_t$ of $G$ has at least one marked vertex. Therefore, $G$ consists of the marked vertices and a set $K$ of chains (i.e., paths consisting of degree-2 vertices) connecting marked vertices. See (the left and middle figures of) Figure 4 for an illustration of the marked vertices and chains.

We claim that $|E_0| < k$, the number of marked vertices in $G$ is bounded by $4k$, and $|K| \leq 5k$. For each $e = (u, v) \in E_0$, let $\gamma_e$ be the (simple) cycle consists of $e$ and the (unique) simple path between $u$ and $v$ in $T_1$, where $t \in [r]$ is the index such that $C_t$ contains $u$ and $v$. By Lemma 23 and 24, we have $\Phi_G = \bigcup_{e \in E_0} \Phi_{C_t} = \bigcup_{e \in E_0} \Phi(\gamma_e)$. By Fact 3, there exists $E_0' \subseteq E_0$ with $|E_0'| < k$ such that $\bigcup_{e \in E_0'} \Phi(\gamma_e) = \bigcup_{e \in E_0} \Phi(\gamma_e)$. Let $G'$ be the subgraph of $G$ obtained by removing all edges in $E_0 \setminus E_0'$. Using Lemma 23 and 24 again, we deduce that

$$\Phi_{G'} = \bigcup_{e \in E_0'} \Phi(\gamma_e) = \bigcup_{e \in E_0} \Phi(\gamma_e) = \Phi_G. \quad (1)$$

Therefore, $G'$ is also $P$-good. It follows that $G' = G$, since no proper subgraph of $G$ is $P$-good. This further implies $E_0' = E_0$ and $|E_0| < k$. Next, we consider the number of vertices in $G$ with degree at least 3. Since $G$ does not have degree-1 vertices, any leaf of the trees $T_1, \ldots, T_r$ must be adjacent to some edge in $E_0$. Since $|E_0| < k$, the number of leaves of $T_1, \ldots, T_r$ is at most $2k$, and hence there are at most $2k$ nodes in $T_1, \ldots, T_r$ whose degree is at least 3. Now observe that a marked vertex $v$ of $G$ is either adjacent to some edge in $E_0$ or of degree at least 3 in the tree $T_t$, where $C_t$ is the component containing $v$. Therefore, there can be at most $4k$ marked vertices in $G$. Finally, we bound $|K|$, the number of chains. Note that each edge of $G$ belongs to exactly one chain in $K$. Therefore, the number of chains containing at least one edge in $E_0$ is at most $k$, because $|E_0| < k$. All the other chains, i.e., the chains that do not have any edge in $E_0$, are contained in the trees $T_1, \ldots, T_r$. It follows that these chains do not form any cycle, and thus their number is less than the number of marked vertices in $G$ (which is at most $4k$). Thus, $G$ has at most $5k$ chains, i.e., $|K| \leq 5k$.

![Figure 4](image-url) An illustration of the marked vertices in $G$ and the resulting graph $H^*$ by path-contraction. The left figure shows the graph $G$ consisting of two connected components where the black edges are tree edges and the grey edges are non-tree edges in $E_0$. The middle figure shows the marked vertices in $G$ (and the chains in $K$ connecting the marked vertices). The right figure shows the graph $H^*$ obtained by path-contraction.

The desired $k$-labeled graph $H^*$ is defined via a path-contraction procedure on $G$ as follows. The vertices of $H^*$ are one-to-one corresponding to the marked vertices of $G$. The edges of $H^*$ are one-to-one corresponding to the chains in $K$: for each chain connecting
two marked vertices $u$ and $v$, we have an edge in $H^*$ connecting the two vertices of $H$ corresponding to $u$ and $v$. The label of each edge $e$ of $H^*$ is defined as $\text{lab}(e) = \text{parity}(\pi_e)$, where $\pi_e$ is the chain in $C$ corresponding to $e$. See Figure 4 for an illustration of how to obtain $H^*$ via path-contraction. Since there are at most $4k$ marked vertices in $G$ and $|K| \leq 5k$, $H^*$ has at most $4k$ vertices and $5k$ edges. Next, we define the PPM $f^* = (f_V^*, f_E^*)$ from $H^*$ to $G_S$. The function $f_V^*$ simply maps each vertex of $H^*$ to its corresponding marked vertex in $G$ (which is a vertex of $G_S$), and the function $f_E^*$ simply maps each edge of $H^*$ to its corresponding chain in $K$ (which is a path in $G_S$). The fact that $f^*$ is a PPM directly follows from the construction of $H^*$. Furthermore, we observe that $\text{cost}(f^*)$ is equal to the number of vertices in $G$, because the chains in $K$ are “interior-disjoint” in the sense that two chains can only intersect at their endpoints. Therefore, $\text{cost}(f^*) = |S_{\text{opt}}| = \text{opt}$. Finally, we show that $H^*$ is $P$-good. It suffices to show $\Phi_{H^*} = \Phi_G$. Consider two elements $i,j \in [k]$ belong to the same part of $\Phi_G$. We have $\text{parity}_i(\gamma) = \text{parity}_j(\gamma)$ for any cycle $\gamma$ in $G$. It follows that $\text{parity}_i(\gamma^*) = \text{parity}_j(\gamma^*)$ for any cycle $\gamma^*$ in $H^*$, because the image of $\gamma^*$ under $f^*$ is a cycle $\gamma$ in $G$ satisfying $\text{parity}(\gamma) = \text{parity}(\gamma^*)$. Thus, $i$ and $j$ belong to the same part of $\Phi_{H^*}$. Next consider two elements $i,j \in [k]$ belong to different parts of $\Phi_G$. By Equation 1, there exists some edge $e \in E_0$ such that $i$ and $j$ belong to different parts of $\Phi(\gamma_i)$, i.e., $\text{parity}_i(\gamma_i) \neq \text{parity}_j(\gamma_i)$. Since $\gamma_i$ is a simple cycle in $G$, it corresponds to a simple cycle in $H^*$, i.e., there is a simple cycle $\gamma^*$ in $H^*$ whose image under $f^*$ is $\gamma_i$. Because $f^*$ is a PPM, we have $\text{parity}(\gamma^*) = \text{parity}(\gamma_i)$. It then follows that $\text{parity}_i(\gamma^*) \neq \text{parity}_j(\gamma^*)$ and hence $i,j$ belong to different parts of $\Phi_{H^*}$. Therefore, $\Phi_{H^*} = \Phi_G$ and $H^*$ is $P$-good.

The above lemma already gives us an algorithm that runs in $2^{O(k^2)}n^{O(k)}$ time. First, we guess the $k$-labeled graph $H^*$ in Lemma 30. Since $H^*$ has at most $4k$ vertices and $5k$ edges, the number of possible graph structures of $H^*$ is $k^{O(k)}$ and the number of possible labeling of the edges of $H^*$ is bounded by $(2k)^{3k}$. Therefore, there can be $2^{O(k^2)}$ possibilities for $H^*$. We enumerate all possible $H^*$, and for every $H^*$ that is $P$-good, we compute a PPM from $H^*$ to $G_S$ with the minimum cost; later we will show how to do this in $n^{O(k)}$ time. Among all these PPMs, we take the one with the minimum cost, say $f^*$. By Lemma 29 and 30, we know that $\text{Im}(f^*)$ is $P$-good and $\text{cost}(f^*) = \text{opt}$. To find an optimal solution, let $S' \subseteq S$ be the set of vertices of $\text{Im}(f^*)$. Since $\text{Im}(f^*)$ is a subgraph of $G_S[S']$ and $\text{Im}(f^*)$ is $P$-good, we know that $G_S[S']$ is also $P$-good and hence $S'$ is a $P$-separator. Furthermore, Fact 28 implies that $|S'| \leq \text{cost}(f^*) = \text{opt}$. Therefore, $S'$ is an optimal solution for the problem instance. The entire algorithm takes $2^{O(k^2)}n^{O(k)}$ time.

Now we discuss the missing piece of the above algorithm, how to compute a PPM from $H^*$ to $G_S$ with the minimum cost in $n^{O(k)}$ time, given a $k$-labeled graph $H^* = (V_{H^*}, E_{H^*})$ with at most $4k$ vertices and $5k$ edges. For all $u,v \in S$ and $l \in \{0,1\}^k$, let $\pi_{u,v,l}$ be the shortest path (i.e., the path with fewest edges) between $u$ and $v$ whose parity is $l$. All these paths can be computed in $2^{O(k)}n^3$ time using Floyd’s algorithm. Suppose $f^* = (f_V^*, f_E^*)$ is the PPM from $H^*$ to $G_S$ we want to compute. Recall that $\text{cost}(f^*) = |V_{H^*}| - |E_{H^*}| + \sum_{e^* \in E_{H^*}} |f_E^*(e^*)|$. The terms $|V_{H^*}|$ and $|E_{H^*}|$ only depend on $H^*$ itself. Therefore, we want to choose $f^*$ that minimizes $\sum_{e^* \in E_{H^*}} |f_E^*(e^*)|$. We simply enumerate all possibilities of $f_V^*$. Since $H^*$ has at most $4k$ vertices, there are at most $n^{4k}$ possible $f_V^*$ to be considered. Once $f_V^*$ is determined, the endpoints of the paths $f_E^*(e^*)$ are also determined. This allows us to minimize $|f_E^*(e^*)|$ for each $e^* \in E_{H^*}$ independently. Let $e^* = (u^*, v^*) \in E_{H^*}$. Since $f^*$ is a PPM, $f_E^*(e^*)$ must be a path connecting $u = f_V^*(u)$ and $v = f_V^*(v)$ whose parity is $l = \text{lab}(e^*)$. By the definition of $\pi_{u,v,l}$, it follows that $|f_E^*(e^*)| \geq |\pi_{u,v,l}|$ and thus setting $f_E^*(e^*) = \pi_{u,v,l}$ will minimize $|f_E^*(e^*)|$. After trying all possible $f_V^*$, we can finally find the optimal PPM $f^*$ in $n^{O(k)}$ time.
6.1 Improving the running time to $2^{O(p)} n^{O(k)}$

To further improve the running time of the above algorithm to $2^{O(p)} n^{O(k)}$ requires nontrivial efforts. Without loss of generality, in this section, we assume $k \leq n$. Indeed, if $k > n$, the problem can be solved in $2^{O(k)}$ time by enumerating every subset $S' \subseteq S$ and checking if $S'$ is a $P$-separator (which can be done in polynomial time by first computing $\Phi_{S'}$ using Lemma 23 and 24 and then applying the criterion of Lemma 26).

As stated before, there are $2^{O(k)}$ possibilities for $H^*$. Thus, in order to improve the factor $2^{O(k)}$ to $2^{O(p)}$, we have to avoid enumerating all possible $H^*$. Instead, we only enumerate the graph structure of $H^*$ (but not the labels of its edges). There are $k^{O(k)}$ possible graph structures to be considered, because $H^*$ has at most $4k$ vertices and $5k$ edges. For each possible graph structure, we want to label the edges to make $H^*$ $P$-good and then find a $PPM$ from $H^*$ (with that labeling) to $G_S$ such that the cost of the PPM is minimized.

Formally, consider a graph structure $H^* = (V_{H^*}, E_{H^*})$ of $H^*$. A labeling-$PPM$ pair for $H^*$ refers to a pair $(\text{lab}, f^*)$ where $\text{lab} : E_{H^*} \to \{0,1\}^k$ is a labeling for $H^*$ and $f^* = (f^*_V, f^*_E)$ is a $PPM$ from $H^*$ to $G_S$ (with respect to the labeling lab). Our task is to find a labeling-$PPM$ pair $(\text{lab}, f^*)$ for $H^*$ with the minimum cost($f^*$) such that $H^*$ is $P$-good with respect to the labeling lab.

Let $C_1, \ldots, C_r$ be the connected components of $H^*$, and $T_1, \ldots, T_r$ be spanning trees of $C_1, \ldots, C_r$, respectively. Let $E_0 \subseteq E_{H^*}$ be the set of edges that are not in $T_1, \ldots, T_r$. For each $e \in E_0$, denote by $\gamma_e$ the cycle in $H^*$ consisting of the edge $e$ and the (unique) simple path between the two endpoints of $e$ in $T_i$, where $t \in [r]$ is the index such that $C_t$ contains $e$. By Lemma 23 and 24, we have $\Phi_{H^*} = \bigotimes_{i=1}^r \Phi_{C_t} = \bigcap_{i \in E_0} \Phi(\gamma_e)$. Therefore, a labeling makes $H^*$ $P$-good iff for every $(i, j) \in P$ there exists an edge $e \in E_0$ such that $\text{parity}_i(\gamma_e) \neq \text{parity}_j(\gamma_e)$ with respect to that labeling. We say a labeling lab : $E_{H^*} \to \{0,1\}^k$ respects a function $\xi : P \to E_0$ if for all $(i, j) \in P$, we have $\text{parity}_i(\gamma_e) \neq \text{parity}_j(\gamma_e)$ where $e = \xi(i, j)$ and parity is calculated with respect to the labeling lab. Then we immediately have the following fact.

**Fact 31.** A labeling makes $H^*$ $P$-good iff it respects some function $\xi : P \to E_0$.

Our first observation is that for any function $\xi : P \to E_0$, one can efficiently find the “optimal” labeling-$PPM$ pair $(\text{lab}, f^*)$ for $H^*$ satisfying the condition that lab respects $\xi$.

**Lemma 32.** Given $\xi : P \to E_0$, one can compute in $2^{O(p)} n^{O(k)}$ time a labeling-$PPM$ pair $(\text{lab}, f^*)$ for $H^*$ which minimizes cost($f^*$) subject to the condition that lab respects $\xi$.

**Proof.** Suppose $f^* = (f^*_V, f^*_E)$ is the $PPM$ we want to compute. We enumerate all possibilities of $f^*_V : V_{H^*} \to S$. Since $|V_{H^*}| \leq 4k$, there are $n^{O(k)}$ different $f^*_V$ to be considered. Fixing a function $f^*_V$, we want to determine the labeling lab and the function $f^*_E$ such that (i) lab respects $\xi$, (ii) $f^*$ is a $PPM$ with respect to the labeling lab, and (iii) cost($f^*$) is minimized. For an edge $e^* = (u^*, v^*) \in E_{H^*}$, and a label $l \in \{0,1\}^k$, we denote by $\text{len}(e^*, l) = |\pi_{u^*, l}|$, where $u = f^*_V(u^*)$, $v = f^*_V(v^*)$. As argued before, for a fixed labeling lab, an optimal function $f^*_E$ is the one that maps each edge $e^* = (u^*, v^*) \in E_{H^*}$ to the path $\pi_{u^*, l}$, where $u = f^*_V(u^*)$, $v = f^*_V(v^*)$, $l = \text{lab}(e^*)$; with this choice of $f^*_E$, we have $\text{cost}(f^*) = |V_{H^*}| - |E_{H^*}| + \sum_{e^* \in E_{H^*}} \text{len}(e^*, \text{lab}(e^*))$. Therefore, our actual task is to find a labeling lab that respects $\xi$ and minimizes $\sum_{e^* \in E_{H^*}} \text{len}(e^*, \text{lab}(e^*))$. Suppose $E_{H^*} = \{e_1, \ldots, e_m\}$ where $m = O(k)$. Let $\delta : [m] \times E_0 \to \{0,1\}$ be an indicator defined as $\delta(t, e) = 1$ if $e$ is an edge of the cycle $\gamma_e$ and $\delta(t, e) = 0$ otherwise. For a labeling lab : $E_{H^*} \to \{0,1\}^k$, we have $\text{parity}(\gamma_e) = \sum_{t=1}^m \delta(t, \xi(t, j)) \cdot \text{lab}_t(e_t)$ for any $e \in E_0$. Therefore, a labeling lab respects $\xi$ iff $\sum_{t=1}^m \delta(t, \xi(i, j)) \cdot \text{lab}_t(e_t) \neq \sum_{t=1}^m \delta(t, \xi(i, j)) \cdot \text{lab}_t(e_t)$ for all $(i, j) \in P$, or equivalently,
23:22 Algorithms for Point Separation and Obstacle Removal

\[ \sum_{i,j}^m \delta(t, \xi(i, j)) : (\text{lab}(e_t) \oplus \text{lab}^j(e_t)) = 1 \text{ for all } (i, j) \in P. \]

So our task is to find a labeling \( \text{lab} \) which minimizes \( \sum_{i=1}^m \text{len}(e_t, \text{lab}(e_t)) \) subject to \( \sum_{i=1}^m \delta(t, \xi(i, j)) : (\text{lab}(e_t) \oplus \text{lab}^j(e_t)) = 1 \text{ for all } (i, j) \in P. \)

Now consider the following problem: for a pair \((t', \phi)\) where \( t' \in [m] \) is an index and \( \phi : P \to \{0, 1\} \) is a function, compute a “partial” labeling \( \text{lab} : \{e_1, \ldots, e_{t'}\} \to \{0, 1\}^k \) such that \( \sum_{i=1}^{t'} \text{len}(e_t, \text{lab}(e_t)) \) is minimized subject to the condition \( \sum_{i=1}^{t'} \delta(t, \xi(i, j)) : (\text{lab}(e_t) \oplus \text{lab}^j(e_t)) = \phi(i, j) \text{ for all } (i, j) \in P. \)

We want to solve the problem for all pairs \((t', \phi)\). This can be achieved using dynamic programming as follows. For a label \( l \in \{0, 1\}^k \), we denote by \( \phi_l : P \to \{0, 1\} \) the function which maps \((i, j) \in P\) to 0 (resp., 1) if the \( i \)-th bit and the \( j \)-th bit of \( l \) is the same (resp., different). We consider the index \( t' \) from 1 to \( m \). Suppose now the problems for all pairs with index \( t' - 1 \) have been solved. To solve for a pair \((t', \phi)\), we enumerate the labeling \( \text{lab}(e_t) \) for \( e_{t'} \). Fixing \( \text{lab}(e_t') = l \), the remaining problem becomes to determine \( \text{lab} : \{e_1, \ldots, e_{t'-1}\} \to \{0, 1\}^k \) that minimizes \( \sum_{i=1}^{t'-1} \text{len}(e_t, \text{lab}(e_t)) \) subject to the condition \( \sum_{i=1}^{t'-1} \delta(t, \xi(i, j)) : (\text{lab}(e_t) \oplus \text{lab}^j(e_t)) = \phi(i, j) \text{ for all } (i, j) \in P. \)

Thus, provided that we already know the solution for the problem for all pairs with index \( t' - 1 \), we can solve the problem for \((t', \phi)\) in \( 2^p \cdot p^{O(1)} \) time. Since there are \( 2^p m \) pairs \((t', \phi)\) to be considered and \( m = O(k) \), the problem for all pairs can be solved in \( 2^{O(p)} \) time.

Now we see that for a fixed \( f^*_\text{lab} \), one can compute in \( 2^{O(p)} \) time the optimal \( \text{lab} \) and \( f^*_\text{lab} \). Since there are \( n^{O(k)} \) possible \( f^*_\text{lab} \) to be considered, the entire algorithm takes \( 2^{O(p)} n^{O(k)} \) time, which completes the proof.

The above lemma directly gives us a \( k^{O(p)} n^{O(k)} \)-time algorithm to compute the desired labeling-PPM pair. By Fact 31, it suffices to compute a labeling-PPM pair \((\text{lab}, f^*)\) for \( H^* \) with the minimum cost(\( f^* \)) such that \( \text{lab} \) respects some function \( \xi : P \to E_0 \). Note that the number of different functions \( \xi : P \to E_0 \) is at most \((5k)^{p}\) because \(|P| = p \) and \(|E_0| \leq 5k\). We simply enumerate all these functions, and for each function \( \xi : P \to E_0 \), we use Lemma 32 to compute in \( 2^{O(p)} n^{O(k)} \) time a labeling-PPM pair \((\text{lab}, f^*)\) for \( H^* \) with the minimum cost(\( f^* \)) such that \( \text{lab} \) respects \( \xi \). Among all the labeling-PPM pairs are computed, we then pick the pair \((\text{lab}, f^*)\) with the minimum cost(\( f^* \)).

To compute the desired labeling-PPM pair more efficiently, we observe that in fact, we do not need to try all functions \( \xi : P \to E_0 \). If a family \( \Xi \) of functions \( \xi : P \to E_0 \) satisfies that any labeling making \( H^* \) \( P \)-good respects some \( \xi \in \Xi \), then trying the functions in \( \Xi \) is already sufficient. We show the existence of such a family \( \Xi \) of size \( k^{O(k)} \).

**Lemma 33.** There exists a family \( \Xi \) of \( k^{O(k)} \) functions \( \xi : P \to E_0 \) such that any labeling making \( H^* \) \( P \)-good respects some \( \xi \in \Xi \). Furthermore, \( \Xi \) can be computed in \( k^{O(k)} \) time.

**Proof.** As the first step of our proof, we establish a bound on the number of sequences of “finer and finer” partitions of \([k]\). Let \( m \geq 1 \) be an integer. An \( m \)-sequence \((\Phi_1, \ldots, \Phi_m)\) of partitions of \([k]\) is finer and finer if \( \Phi_1 \supseteq \cdots \supseteq \Phi_m \). We show that the total number of finer and finer \( m \)-sequences is bounded by \((m + k)^{O(k)} \).

To this end, we first observe that the number of non-decreasing sequences \((z_1, \ldots, z_m)\) of integers in \([k]\) is \( \binom{m+k-1}{k} = (m + k)^{O(k)} \).

Therefore, it suffices to show that for any non-decreasing sequence \((z_1, \ldots, z_m)\) of integers in \([k]\), the number of finer and finer \( m \)-sequences \((\Phi_1, \ldots, \Phi_m)\) satisfying \( |\Phi_i| = z_i \) for all \( i \in [m] \) is bounded by \((m + k)^{O(k)} \). Fix a non-decreasing sequence \((z_1, \ldots, z_m)\) of integers in \([k]\). For convenience, define \( \Phi_{m+1} = \{\{1\}, \ldots, \{k\}\} \) as finest partition of \([k]\) and let \( z_{m+1} = |\Phi_{m+1}| = k \). Then we must have \( \Phi_m \supseteq \Phi_{m+1} \). By applying Fact 4, for a fixed \( \Phi_{i+1} \) with \(|\Phi_{i+1}| = z_{i+1} \), the number of partitions \( \Phi_i \supseteq \Phi_{i+1} \) with \(|\Phi_i| = z_i \) is \( z_{i+1}^{O(d_{i+1})} \).
where $d_{t+1} = z_{t+1} - z_t$. Therefore, by a simple induction argument we see that for an index $t \in [m]$, the number of the possibilities of the subsequence $(\Phi_t, \ldots, \Phi_m)$ is bounded by $\prod_{i=t}^{m} O(d_{i+1}) = k^{O(k-z_t)}$. In particular, the number of finer and finer $m$-sequences $(\Phi_t, \ldots, \Phi_m)$ satisfying $|\Phi_i| = z_i$ for all $i \in [m]$ is bounded by $k^{O(k)}$. Furthermore, we observe that these sequences can be computed in $O(m) + k^{O(k)}$ time by repeatedly using Fact 4. Indeed, by Fact 4, for a fixed subsequence $(\Phi_{t+1}, \ldots, \Phi_m)$, one can compute in $k^{O(d_{t+1})}$ time all $\Phi_t$ such that $|\Phi_t| = z_t$ and $\Phi_t \geq \Phi_{t+1}$ time, where $d_{t+1} = z_{t+1} - z_t$. Therefore, knowing all $k^{O(k-z_{t+i})}$ possible subsequences $(\Phi_{t+1}, \ldots, \Phi_m)$, one can compute all possible subsequences $(\Phi_t, \ldots, \Phi_m)$ in $k^{O(k-z_t)}$ time. In particular, all finer and finer $m$-sequences $(\Phi_t, \ldots, \Phi_m)$ satisfying $|\Phi_i| = z_i$ for all $i \in [m]$ can be computed in $O(m) + k^{O(k)}$ time. The $(m+1)^{O(k)}$ non-decreasing sequences $(z_1, \ldots, z_m)$ of integers in $[k]$ can be easily enumerated in $(m+k)^{O(k)}$ time, which implies that all finer and finer $m$-sequences of partitions of $[k]$ can be computed in $(m+k)^{O(k)}$ time.

With the above result, we are now ready to prove the lemma. Suppose $E_0 = \{e_1, \ldots, e_m\}$ where $m = O(k)$. We construct a family $\Xi$ of functions $\xi : P \to E_0$ as follows. For every finer and finer $m$-sequence $(\Phi_1, \ldots, \Phi_m)$ of partitions of $[k]$ satisfying that $i$ and $j$ belong to different parts in $\Phi_m$ for all $(i, j) \in P$, we include in $\Xi$ a corresponding function $\xi : P \to E_0$ defined by setting $\xi(i, j) = e_i$ where $t \in [m]$ is the smallest index such that $i$ and $j$ belong to different parts in $\Phi_t$. By the above result, we have $|\Xi| = k^{O(k)}$ and $\Xi$ can be computed in $k^{O(k)}$ time. It suffices to prove that $\Xi$ satisfies the desired property. Let $\text{lab} : E_H \to \{0, 1\}^k$ be a labeling that makes $H^* - P$-good. Recall that we have $\Phi_H = \bigcap_{i=1}^{k} \Phi(\gamma_{e_i})$. Now we define a finer and finer $m$-sequence $(\Phi_1, \ldots, \Phi_m)$ of partitions of $[k]$ by setting $\Phi_t = \bigcap_{i=1}^{k} \Phi(\gamma_{e_i})$ for all $t \in [m]$.

Then we have $\Phi_m = \Phi_H$. Since $H^*$ is $P$-good, we know that $i$ and $j$ belong to different parts in $\Phi_m$ for all $(i, j) \in P$. Let $\xi \in \Xi$ be the function corresponding to the sequence $(\Phi_1, \ldots, \Phi_m)$. We shall show that $\text{lab}$ respects $\xi$. Consider a pair $(i, j) \in P$ and suppose $\xi(i, j) = e_i$ for some $t \in [m]$. We want to verify that $\text{parity}_i(\gamma_{e_i}) \neq \text{parity}_j(\gamma_{e_i})$. If $t = 1$, then $i$ and $j$ belong to different parts in $\Phi_1 = \Phi(\gamma_{e_i}) = \Phi(\gamma_{e_j})$, i.e., $\text{parity}_i(\gamma_{e_i}) \neq \text{parity}_j(\gamma_{e_j})$. If $t > 1$, then $i$ and $j$ belong to different parts in $\Phi_t$ but belong to the same parts in $\Phi_{t-1}$, which implies that $i$ and $j$ belong to different parts in $\Phi(\gamma_{e_i})$, i.e., $\text{parity}_i(\gamma_{e_i}) \neq \text{parity}_j(\gamma_{e_j})$. This completes the proof.

With the above lemma in hand, we simply construct the family $\Xi$ in $k^{O(k)}$ time, and only try the functions in $\Xi$. This improves the running time to $2^{O(p)} k^{O(k)}$, which is $2^{O(p) + O(k)}$ because $k \leq n$ by our assumption.

**Theorem 34. Generalized Point-Separation for connected obstacles in the plane can be solved in $2^{O(p)} k^{O(k)}$ time, where $n$ is the number of obstacles, $k$ is the number of points, and $p$ is the number of point-pairs to be separated.**

**Corollary 35. Point-Separation for connected obstacles in the plane can be solved in $2^{O(k^2)} k^{O(k)}$ time, where $n$ is the number of obstacles and $k$ is the number of points.**

### 7 An Improved Algorithm for Pseudo-disk Obstacles

In this section, we study Generalized Points-separation for pseudo-disk obstacles and obtain an improved algorithm. To this end, the key observation is the following analog of Lemma 26 for pseudo-disk obstacles.

**Lemma 36.** Suppose $S$ consists of pseudo-disk obstacles. Then a subset $S' \subseteq S$ is a $P$-separator iff there is a subgraph of the induced subgraph $G_S[S']$ that is planar and $P$-good.
Proof. The “if” part follows immediately from Lemma 26. So it suffices to show the “only if” part. Let \( S' \subseteq S \) be a \( P \)-separator and \( U = \bigcup_{S \subseteq S} S \). Recall that two obstacles \( S, S' \in S \) contribute to \( U \) if an intersection point of the boundaries of \( S \) and \( S' \) is a break point on the boundary of \( U \) (see Section 2). By Fact 5, the graph \( G' = (S', E) \) where \( E = \{ (S, S'): S, S' \in S \) contribute to \( U \}) \) is planar. We define a subgraph \( G \) of the induced subgraph \( G_S[S'] \) as follows. The vertex set of \( G \) is \( S' \). For each edge \( e = (S, S') \) of \( G_S[S'] \), if \( S, S' \) contribute to \( U \) or \( S = S' \), then we include \( e \) in \( G \), otherwise we discard it. We observe that \( G' \) is planar. Indeed, \( G \) can be obtained from \( G' \) by adding parallel edges and self-loops. Since \( G' \) is planar and adding parallel edges and self-loops does not change planarity, \( G \) is also planar. It now suffices to prove that \( G \) is \( P \)-good. Consider a pair \((i, j) \in P \) and we want to show the existence of a cycle \( \gamma \) in \( G \) such that \( \text{parity}_i(\gamma) \neq \text{parity}_j(\gamma) \). In the proof of Lemma 26, we constructed a cycle \( \gamma \) in \( G_S[S'] \) satisfying \( \text{parity}_i(\gamma) \neq \text{parity}_j(\gamma) \). In that construction, \( \gamma \) also satisfies the following property: for each pair \((S, S') \) of two consecutive vertices in \( \gamma \), there are two adjacent arcs in the boundary of \( U \) contributed by \( S \) and \( S' \) respectively, which implies that \( S, S' \) contribute to \( U \). Therefore, \( \gamma \) is also a cycle in \( G \). It follows that \( G \) is \( P \)-good, completing the proof. \( \square \)

With the above lemma in hand, we are now ready to prove an analog of Lemma 30 for pseudo-disk obstacles. The only difference is that here we can require \( H^* \) to be planar.

\[ \textbf{Lemma 37.} \text{ Suppose } S \text{ is a set of pseudo-disk obstacles. Then there exists a } P \text{-good } k \text{-labeled planar graph } H^* \text{ with at most } 4k \text{ vertices and } 5k \text{ edges and a PPM } f^* \text{ from } H^* \text{ to } G_S \text{ such that } \text{cost}(f^*) = \text{opt}. \]

Proof. Recall that in the proof of Lemma 30, we first took a minimal \( P \)-good subgraph \( G \) of the induced subgraph \( G_S[S'] \), and then obtained \( H^* \) by applying a path-contraction procedure on \( G \). The choice of \( G \) is arbitrary as long as it is a minimal \( P \)-good subgraph of \( G_S[S'] \). Furthermore, if \( G \) is planar, then the resulting \( H^* \) is also planar because the path-contraction procedure preserves planarity. Therefore, it suffices to show that \( G_S[S'] \) has a minimal \( P \)-good subgraph that is planar. By Lemma 36, there exists a \( P \)-good subgraph of \( G_S[S'] \) that is planar. Since subgraphs of a planar graph are also planar, there exists a minimal \( P \)-good subgraph of \( G_S[S'] \) that is planar, which completes the proof. \( \square \)

Now we explain how the planarity of \( H^* \) in Lemma 37 helps us solve the problem more efficiently. Recall how our algorithm in Section 6.1 works. We first enumerate the graph structure \( H^* = (V_H, E_H) \) of \( H^* \). For a fixed graph structure, let \( C_1, \ldots, C_r \) be the connected components of \( H^* \), and \( T_1, \ldots, T_r \) be spanning trees of \( C_1, \ldots, C_r \), respectively. Let \( E_0 \subseteq E_{H^*} \) be the set of edges that are not in \( T_1, \ldots, T_r \). We then create the family \( \Xi \) of functions \( \xi : P \rightarrow E_0 \) in Lemma 33. For each \( \xi \in \Xi \), we use the algorithm of Lemma 32 to efficiently compute the “optimal” labeling-PPM pair \((\text{lab}, f^*)\) for \( H^* \) satisfying the condition that \( \text{lab} \) respects \( \xi \). Here we apply the same framework, but replace Lemma 32 with an improved algorithm which works for the case that \( H^* \) is planar. The key ingredient of this improved algorithm is the planar separator theorem, which allows us to solve the problem of Lemma 32 more efficiently using divide-and-conquer when \( H^* \) is planar.

\[ \textbf{Lemma 38.} \text{ Suppose } H^* \text{ is planar. Given } \xi : P \rightarrow E_0, \text{ one can compute in } 2^{O(n \sqrt{k})} \text{ time a labeling-PPM pair } (\text{lab}, f^*) \text{ for } H^* \text{ which minimizes } \text{cost}(f^*) \text{ subject to the condition that } \text{lab} \text{ respects } \xi. \]

Proof. As in the proof of Lemma 32, suppose \( E_{H^*} = \{ e_1, \ldots, e_m \} \) where \( m = O(k) \). Let \( \delta : [m] \times E_0 \rightarrow \{0, 1\} \) be an indicator defined as \( \delta(t, e) = 1 \) if \( e_t \) is an edge of the cycle
\( \gamma_c \) and \( \delta(t, e) = 0 \) otherwise. Consider a triple \((H, V', f'_V)\), where \( H = (V_H, E_H) \) is a subgraph of \( H^* \), \( V' \subseteq V_H \) is a subset of the vertex set of \( H \), and \( f'_V : V' \to S \) is a mapping. For such a triple, we define a corresponding problem: for every function \( \phi : P \to \{0, 1\} \), computing a labeling-PPM pair \((\text{lab}, f)\) for \( H \) (i.e., \( \text{lab} : E_H \to \{0, 1\}^k \)) is a labeling for the edges of \( H \) and \( f \) is a PPM from \( H \) to \( G_S \) with respect to the labeling \( \text{lab} \) that minimizes \( \text{cost}(f) \) subject to (i) \( f \) is compatible with \( f'_V \), i.e., \( f \) maps every \( v \in V' \) to \( f'_V(v) \) and (ii) \( \sum_{e_t \in E_H} \delta(t, (i, j)) \cdot (\text{lab}_i(e_t) \oplus \text{lab}_j(e_t)) = \phi(i, j) \).

We show how to solve the problem instance \((H, V', f'_V)\) efficiently using divide-and-conquer. Let \( c \) be a sufficiently large constant. If \( |V_H| \leq c \), we simply solve the instance using brute-force in \( O(1) \) time. Assume \( |V_H| > c \). Since \( H^* \) is planar, \( H \) is also planar. Thus, by the planar separator theorem, we can find in \( |V_H|^{O(1)} \) time a partition of \( V_H \) into three sets \( V_1, V_2, X \) such that (i) there is no edge in \( E_H \) between \( V_1 \) and \( V_2 \), (ii) \( |X| \leq 3\sqrt{|V_H|} \), and (iii) \(|V_1| \leq \frac{2}{3}|V_H|\) and \(|V_2| \leq \frac{2}{3}|V_H|\). We define two subgraphs \( H_1 \) and \( H_2 \) of \( H \) as follows. The graph \( H_1 = (V_{H_1}, E_{H_1}) \) is the induced subgraph \( H[V_1 \cup X] \), and the graph \( H_2 = (V_{H_2}, E_{H_2}) \) is defined as \( V_{H_2} = V_2 \cup X \) and \( E_{H_2} = E_{H \setminus E_H} \). Observe that \( H_1 \) and \( H_2 \) cover all the vertices and edges of \( H \). In addition, \( H_1 \) and \( H_2 \) share the common vertices in \( X \) and do not share any common edges. Let \( V'_1 = (X \cup V') \cap V_H \) and \( V'_2 = (X \cup V') \cap V_{H_2} \). We enumerate all functions \( g : X \to S \) that compatible with \( f'_V \), i.e., \( g(v) = f'_V(v) \) for all \( v \in X \cap V' \). The number of such functions is \( n^{O(\sqrt{|V_H|})} \) because \( |X| = O(\sqrt{|V_H|}) \). For a fixed function \( g : X \to S \), let \( g' : X \cup V' \to S \) be the function obtained by gluing \( g \) and \( f'_V \), i.e., \( g'(v) = g(v) \) on \( X \) and \( g'(v) = f'_V(v) \) on \( V' \). We then recursively solve the two problem instances \( \text{Prob}_{g,1} = (H_1, V'_1, g'_1) \) and \( \text{Prob}_{g,2} = (H_2, V'_2, g'_2) \) where \( g'_1 \) (resp., \( g'_2 \)) is the function obtained by restricting \( g' \) to \( V'_1 \) (resp., \( V'_2 \)). After all functions \( g : X \to S \) are considered, we collect all the solutions for the problem instances \( \text{Prob}_{g,1} \) and \( \text{Prob}_{g,2} \).

We are going to use these solutions to obtain the solution for the problem instance \((H, V', f'_V)\). Recall that for every function \( \phi : P \to \{0, 1\} \), we want to compute a labeling-PPM pair \((\text{lab}, f)\) for \( H \) that minimizes \( \text{cost}(f) \) subject to (i) \( f \) is compatible with \( f'_V \) and (ii) \( \sum_{e_t \in E_H} \delta(t, (i, j)) \cdot (\text{lab}_i(e_t) \oplus \text{lab}_j(e_t)) = \phi(i, j) \). We first guess how the desired PPM \( f \) maps the vertices in \( X \), which can be described as a function \( g : X \to S \). There are in total \( n^{O(\sqrt{|V_H|})} \) guesses we need to make. Now suppose our guess for \( g \) is correct. As before, we define \( g' : X \cup V' \to S \) as the function obtained by gluing \( g \) and \( f'_V \). Note that \( f \) is compatible with \( g' \). Let \((\text{lab}_1, f_1)\) and \((\text{lab}_2, f_2)\) denote labeling-PPM pairs for \( H_1 \) and \( H_2 \), respectively, obtained by restricting \((\text{lab}, f)\) to \( H_1 \) and \( H_2 \). Define \( \phi_1 : P \to \{0, 1\} \) as \( \phi_1(i, j) = \sum_{e_t \in E_{H_1}} \delta(t, (i, j)) \cdot (\text{lab}_1(e_t) \oplus \text{lab}_2(e_t)) \) and \( \phi_2 : P \to \{0, 1\} \) as \( \phi_2(i, j) = \sum_{e_t \in E_{H_2}} \delta(t, (i, j)) \cdot (\text{lab}_1(e_t) \oplus \text{lab}_2(e_t)) \). We observe that \( f_1 \) and \( f_2 \) must be compatible with \( g'_1 \) and \( g'_2 \), respectively, where \( g'_1 \) (resp., \( g'_2 \)) is the function obtained by restricting \( g' \) to \( V'_1 \) (resp., \( V'_2 \)), because \( f \) is compatible with \( g' \). Also, we have \( \phi = \phi_1 \oplus \phi_2 \) and \( \text{cost}(f) = \text{cost}(f_1) + \text{cost}(f_2) - |X| \), because \( V_{H_1} \cap V_{H_2} = X \) and \( \{E_{H_1}, E_{H_2}\} \) is a partition of \( E_H \). On the other hand, as long as \( f_1 \) and \( f_2 \) are compatible with \( g'_1 \) and \( g'_2 \) respectively and \( \phi = \phi_1 \oplus \phi_2 \), we can always glue the two labeling-PPM pairs \((\text{lab}_1, f_1)\) and \((\text{lab}_2, f_2)\) to obtain a labeling-PPM pair \((\text{lab}, f)\) for \( H \) satisfying \( \text{cost}(f) = \text{cost}(f_1) + \text{cost}(f_2) - |X| \) such that (i) \( f \) is compatible with \( g' \) and (ii) \( \sum_{e_t \in E_H} \delta(t, (i, j)) \cdot (\text{lab}_1(e_t) \oplus \text{lab}_2(e_t)) = \phi(i, j) \). Therefore, we can solve the problem as follows. We simply guess the functions \( \phi_1 \) and \( \phi_2 \) satisfying \( \phi_1 \oplus \phi_2 = \phi \). There are in total \( 2^p \) guesses we need to make. Suppose our guess is correct. We retrieve the solution \((\text{lab}_1, f_1)\) of the problem instance \( \text{Prob}_{g,1} \) for the function \( \phi_1 \) and the solution \((\text{lab}_2, f_2)\) of the problem instance \( \text{Prob}_{g,2} \) for the function \( \phi_2 \), which have already been computed. We know that \((\text{lab}_1, f_1)\) (resp., \((\text{lab}_2, f_2)\)) minimizes \( \text{cost}(f_1) \) (resp., \( \text{cost}(f_2) \)) subject to (i) \( f_1 \) is compatible with \( g'_1 \) (resp., \( f_2 \) is...
 Algorithms for Point Separation and Obstacle Removal

compatible with \( g' \)) and (ii) \( \sum_{e \in E_1} \delta(t, \xi(i,j)) \cdot (\text{lab}_i(e) \oplus \text{lab}_j(e)) = \phi_1(i,j) \) (resp., \( \sum_{e \in E_2} \delta(t, \xi(i,j)) \cdot (\text{lab}_i(e) \oplus \text{lab}_j(e)) = \phi_2(i,j) \)). By gluing \((\text{lab}_1, f_1)\) and \((\text{lab}_2, f_2)\), we obtain a labeling-PPM pair \((\text{lab}, f)\) for \( H \), which is what we want because of the optimality of \((\text{lab}_1, f_1)\) and \((\text{lab}_2, f_2)\). In the previous sections, we gave an \( O(\sqrt{h}) \)-time algorithm for \( H \) to find a coloring mapping \( G \rightarrow H \) which solves the generalized point-separation problem in \( O(n^{O(\sqrt{h})}) \) time. Besides the recursive calls, all work can be done in \( 2^{O(p)}n^{O(\sqrt{h})} \) time.

Finally, we analyze the running time of the above algorithm. Let \( T(h) \) denote the time cost for solving a problem instance \((H, V', f'_V)\) with \(|V_H| = h\). We have \( T(h) = O(1) \) for \( h \leq c \), because we use brute-force for the case \( h \leq c \). Suppose \( h > c \). In this case, we have recursive calls on the subgraphs \( H_1 \) and \( H_2 \) of \( H \). Note that \(|V_1| + |X| \leq \frac{3}{4}h + 3\sqrt{h} \leq \frac{3}{4}h\), because \( h > c \) and \( c \) is sufficiently large. Similarly, we have \( H_2 \leq \frac{3}{4}h \). The number of recursive calls is \( n^{O(\sqrt{h})} \). Besides the recursive calls, all work can be done in \( 2^{O(p)}n^{O(\sqrt{h})} \) time. Therefore, we have the recurrence \( T(h) = n^{O(\sqrt{h})} \cdot T\left(\frac{3}{4}h\right) + 2^{O(p)}n^{O(\sqrt{h})} \), which solves to \( T(h) = 2^{O(p)}n^{O(\sqrt{h})} \). To solve the problem of the lemma, the initial call is for the problem instance \((H^*, \text{null})\), which takes \( 2^{O(p)}n^{O(\sqrt{h})} \) time since \(|V_{H^*}| = O(k)\).

Replacing Lemma 32 with Lemma 38, we can apply the algorithm in Section 6.1 to solve the generalized point-separation problem in \( 2^{O(p)}k^{O(1)}n^{O(\sqrt{h})} \) time.

Theorem 39. Generalized Point-Separation for pseudo-disk obstacles in the plane can be solved in \( 2^{O(p)}k^{O(1)}n^{O(\sqrt{h})} \) time, where \( n \) is the number of obstacles, \( k \) is the number of points, and \( p \) is the number of point-pairs to be separated.

Corollary 40. Point-Separation for pseudo-disk obstacles in the plane can be solved in \( 2^{O(k^2)}n^{O(\sqrt{h})} \) time, where \( n \) is the number of obstacles and \( k \) is the number of points.

8 ETH-Hardness of Points-Separation

In the previous sections, we gave an \( f(k) \cdot n^{O(1)} \)-time algorithm for \( k \)-Point-Separation with general (connected) obstacles and an \( f(k) \cdot n^{O(\sqrt{h})} \)-time algorithm with pseudo-disk obstacles. In this section, we show that assuming Exponential Time Hypothesis (ETH), both of our algorithms are almost tight and significant improvement is unlikely. We begin by describing our reduction for general obstacles.

8.1 Hardness for General Obstacles

We give a reduction from Partitioned Subgraph Isomorphism (PSI) problem which is defined as follows. Recall that in the Subgraph Isomorphism problem, we are given two graphs \( G \) and \( H \) and we want to find an injective mapping \( \psi : V(G) \rightarrow V(H) \) such that if \((u, v) \in E(G)\), then \((\psi(u), \psi(v)) \in E(H)\). In the Partitioned Subgraph Isomorphism problem, we want to find a colorful mapping \( G \rightarrow H \). Formally, we are given undirected graphs \( H \) and \( G \) with maximum degree 3, and a coloring function \( \text{col} : V(H) \rightarrow V(G) \) that partitions vertices of \( H \) into \(|V(G)| \) classes. We say that an injective mapping \( \psi : V(G) \rightarrow V(H) \) is a colorful mapping of \( G \) into \( H \), if for every \( v \in V(G) \), \( \text{col}(\psi(v)) = v \), and for every \((u, v) \in E(G)\), we have \((\psi(u), \psi(v)) \in E(H)\). Then in the Partitioned Subgraph Isomorphism, we want to find if there exists a colorful mapping of \( G \) into \( H \).

We will use the following well-known result of Marx [23] relevant to our reduction.

Theorem 41. [23, Corollary 6.3] Unless ETH fails, PSI cannot be solved in \( f(k)n^{o(k/ \log k)} \) time for any function \( f \) where \( k = |E(G)| \) and \( n = |V(H)| \).
Our Construction.

Given an instance of PSI as graphs $G,H$ and coloring $\text{col} : V(H) \to V(G)$, we want to construct an instance of Points-separation, namely a set of obstacles $S$ and a set of points $A$ such that all point pairs in $A$ are separated. For the ease of exposition, we will first discuss a reduction from PSI to an instance $(S,A,P)$ of Generalized Points-separation where the set $P$ of request pairs is specified. Later we extend the construction to show that the same bounds also hold for Points-separation.

The set of obstacles $S$ used in our construction mainly consists of an obstacle $S_{pq}$ for every edge $(u_p,u_q) \in E(H)$. In addition, we also use an additional auxiliary obstacle denoted by $S_0$. All the obstacles and request pairs will be contained in a rectangle $R$ with bottom-left corner $(0,0)$ and top-right corner $(z,3)$, where $z$ is the total number of request pair groups. Each group can have at most two request pairs. We split the rectangle $R$ into $z$ blocks, each of width one. The $r$-th block $B_r$ is bounded by the vertical lines $x = r - 1$ and $x = r$, contains the $r$-th request pair group. Initially all obstacles are horizontal line segments of length $z$ occupying the part of $x$-axis from $x = 0$ to $x = z$ and coincident to the bottom side of $R$. Moreover, let $\ell_1,\ell_2$ be two horizontal line segments coincident with $y = 1$ and $y = 2$ respectively and starting from $x = 0$ (left boundary of $R$) and ending at $x = z$ (right boundary of $R$). These line segments will serve as guardrails for obstacle growth. Specifically obstacles can only grow vertically at $x = r$ (for some integer $r$) or horizontally along the lines $\ell_1,\ell_2$. (See also Figure 5.)

![Figure 5](image-url) An example construction. (a) Graphs $G$ and $H$ with the $\text{col} : V(H) \to V(G)$ shown by dotted boxes around nodes. (b) Block with Type-1 request pair for edge $(v,w) \in E(G)$. (c) Block with Type-2 request pair group for vertex $v$ and its adjacent edges $(v,w),(v,x) \in E(G)$. Obstacles $S_{1,2}$ and $S_{2,3}$ separate both pairs $(a,a')$ and $(a',a'')$ whereas $S_{1,2}$ and $S_{4,5}$ does not.

The $r$-th request pair group is contained in block $B_r$ and may consist of points $a_r, a'_r, a''_r$ where $a_r = (r - \frac{1}{2}, \frac{5}{2})$, $a'_r = (r - \frac{1}{2}, \frac{3}{2})$ and $a''_r = (r - \frac{1}{2}, \frac{1}{2})$. We have two types of groups: Type-1 request pair group consisting of one request pair $(a_r,a'_r)$ and Type-2 request pair group consisting of two request pairs $p_r = (a_r,a'_r)$ and $p'_r = (a'_r,a''_r)$. Depending on the type of the group, we will now grow the obstacles in a systematic manner so that they interact in the neighborhood of request pairs.

1. **Type-1 request pair group** For every edge $e_i = (v,w) \in E(G)$, we add a request pair $p_r = (a_r,a'_r)$ to $P$. Next we grow the obstacles around $p_r$ as follows. (See also Figure 5b.)
   - Extend the auxiliary obstacle $S_0$ vertically along $x = r$ until $y = 2$.
   - For every $(u_p,u_q) \in E(H)$ such that $(\text{col}(u_p),\text{col}(u_q)) = e_i$, extend the obstacle $S_{pq}$ vertically along $x = r - 1$ until $y = 2$ and then rightwards along $\ell_2$ until it touches $S_0$.

Observe that to separate Type-1 request pair $p_r$, we must select $S_0$ and one obstacle corresponding to an edge of $H$. 


2. **Type-2 request pair group** For a vertex $v \in V(G)$ and pair of edges $e_i, e_j \in E(G)$ adjacent to $v$ with $i < j$, we add two request pairs $p_r = (a_r, a'_r)$ and $p'_r = (a'_r, a''_r)$ to $P$. In order to grow the obstacles, consider the unit length intervals along lines $\ell_1, \ell_2$ contained in $B_r$. We subdivide these intervals by adding $n$ markers each separated by a small distance $\epsilon = \frac{1}{n+1}$. Here $n = |V(H)|$. We will use these markers to define the precise boundary of obstacles in block $B_r$. (See also Figure 5c.)

Let $e_i = (v, w)$ and $S_{pq} = (u_p, u_q)$ be an obstacle such that $(\text{col}(u_p), \text{col}(u_q)) = e_i$. Without loss of generality, assume that $\text{col}(u_p) = v$ and $\text{col}(u_q) = w$. First we extend $S_{pq}$ along the left boundary of $B_r$ along $x = r - 1$ until $y = 2$. Then we connect $S_{pq}$ to marker $p$ along line $\ell_1$ and to marker $n - p + 1$ along line $\ell_2$, moving from left to right.

Similarly, let $e_j = (v, x)$ and $S_{gh} = (u_g, u_h)$ be an obstacle such that $(\text{col}(u_g), \text{col}(u_h)) = e_j$. Without loss of generality, assume that $\text{col}(u_g) = v$ and $\text{col}(u_h) = w$. We extend $S_{gh}$ along the right boundary of $B_r$ along $x = r$ until $y = 2$. Then we connect $S_{gh}$ to marker $g$ along line $\ell_1$ and to marker $n - g + 1$ along line $\ell_2$, moving from right to left. Observe that to separate both Type-2 request pairs $p_r$ and $p'_r$, we must select two obstacles corresponding to edges of $H$.

It is easy to verify that all the obstacles are simple and connected. Observe that since each vertex has maximum degree 3, the total number of request pairs added is $z \leq |E(G)| + 2 \cdot 3|V(G)| = O(k)$ where $k = V(G)$. The total number of obstacles $|S| = |E(H)| + 1 = O(n^2)$ where $n = |V(H)|$.

**Observation 42.** For the Generalized Points-separation instance $(S, A, P)$ constructed above, we have $|S| = O(n^2)$, $|A| = O(k)$ and $|P| = O(k)$.

We prove the following lemma which will be useful later.

**Lemma 43.** Let $p_r = (a_r, a'_r)$ and $p'_r = (a'_r, a''_r)$ be a Type-2 request pair group corresponding to vertex $v$ and its two adjacent edges $e_i = (v, w)$ and $e_j = (v, x)$ such that $i < j$. Then two obstacles $S_{pq}$ defined by the edge $(u_p, u_q)$ and $S_{gh}$ defined by $(u_g, u_h)$ separate both $p_r$ and $p'_r$ if and only if $p = g$ and $\text{col}(u_p) = \text{col}(u_g) = v$, $\text{col}(u_q) = w$, $\text{col}(u_h) = x$.

**Proof.** The reverse direction is easy to verify. Specifically, if $\text{col}(u_p) = \text{col}(u_g) = v$, $\text{col}(u_q) = w$, $\text{col}(u_h) = x$ then the obstacles $S_{pq}$ and $S_{gh}$ are respectively coincident with left and right boundary of block $B_r$. Moreover, since $p = g$, both the obstacles overlap precisely at marker $p$ along $\ell_1$ and $n - p + 1$ along $\ell_2$, forming a closed curve containing only point $a'_r = (r - 1, \frac{3}{2})$. Therefore, both the pairs $p_r$ and $p'_r$ are separated.

For the other direction, from the way obstacles $S_{pq}$ and $S_{gh}$ interact in block $B_r$: they may overlap along $\ell_1$ or $\ell_2$ or both or none. If the obstacles overlap only along $\ell_1$, they cannot separate pair $p_r$. Similarly, if they overlap only along $\ell_2$, they cannot separate the pair $p'_r$. Since both pairs are separated, obstacles $S_{pq}$ and $S_{gh}$ must overlap along both $\ell_1$, $\ell_2$ and form a closed curve containing point $a'_r$. This can only happen if $S_{pq}, S_{gh}$ overlap in block $B_r$ approaching $\ell_1, \ell_2$ from opposite sides. Without loss of generality, we can assume that $S_{pq}$ is coincident with left boundary of $B_r$ and $S_{gh}$ is coincident with the right boundary of $B_r$. This can happen only if $\text{col}(u_p) = \text{col}(u_g) = v$, $\text{col}(u_q) = w$, $\text{col}(u_h) = x$. It remains to show that $p = g$. Observe that since $S_{pq}, S_{gh}$ overlap on $\ell_1$, we must have that marker $p$ is to the right of marker $g$. That is $p \geq g$. Similarly, since $S_{pq}, S_{gh}$ overlap on $\ell_2$, we have $n - p + 1 \geq n - g + 1$ which gives $p \leq g$. Combining these, we get $p = g$. ▶

We now prove the following lemma that establishes the correctness of our reduction.
Lemma 44. Given an instance of PSI as graphs $G, H$ and coloring, $\text{col}: V(H) \rightarrow V(G)$, there exists a colorful mapping $\psi: V(G) \rightarrow V(H)$ if and only if the point pairs $P$ can be separated by a set of $m = |E(G)| + 1$ obstacles $S^* \subseteq S$.

Proof. ($\Rightarrow$) Given a colorful mapping $\psi$ we construct the set of obstacles $S^*$ as follows. For every edge $e = (v, w) \in E(G)$, include the obstacle $(\psi(v), \psi(w))$ to $S^*$ — such an obstacle always exists because $(\psi(v), \psi(w)) \in E(H)$. Next, we add $S_0$ to $S^*$. It is easy to verify that $S^*$ separates the Type-1 request pairs. For a Type-2 request pair group $p_r, p'_r$, at vertex $v$ and edges $e_i = (v, w), e_j = (v, x)$, let $u_p = \psi(v), u_q = \psi(w)$ and $u_h = \psi(x)$. Since $\psi$ is a colorful mapping, we have $\text{col}(u_p) = \text{col}(\psi(v)) = v$. Similarly, $\text{col}(u_q) = w$ and $\text{col}(u_h) = x$.

Therefore, it follows from Lemma 43 that $S^*$ separates request pairs $p_r, p'_r$, for all $1 \leq r \leq z$.

($\Leftarrow$) Given a set $S^*$ of $m$ obstacles that separates all request pairs, we will first construct an injective function $M : E(G) \rightarrow E(H)$ that uniquely maps every edge of $G$ to an edge of $H$. Consider the set $P_1$ of Type-1 request pairs. Since $S^*$ separates $P_1$, it must include $S_0$ and a unique obstacle $S_{pq} = (u_p, u_q)$ for every edge $e_i = (v, w) \in E(G)$ such that $(\text{col}(u_p), \text{col}(u_q)) = e_i$. The uniqueness of $S_{pq}$ follows from the fact that there are $|E(G)|$ Type-1 request pairs and $|S^*| = |E(G)| + 1$ obstacles. We assign $M(e_i) = (u_p, u_q)$.

Next, we build a colorful mapping $\psi$ that is consistent with the mapping $M$ of edges. For this, we use the fact that $S^*$ also separates Type-2 request pair groups. Consider the Type-2 request pair group corresponding to vertex $v \in V(G)$ and edges $e_i = (v, w)$ and $e_j = (v, x)$ with $i < j$. We apply Lemma 43 over this group with obstacles defined by edges $(u_p, u_q) = M(e_i)$, and $(u_p, u_q) = M(e_j)$. This gives $u_p = u_q$ and $\text{col}(u_p) = v$. Since this holds for every pair of edges $e_i, e_j$ adjacent to vertex $v$, we can assign $\psi(v) = u_p$, which also satisfies $\text{col}(\psi(v)) = \text{col}(u_p) = v$ required for a colorful mapping. Repeating this for every $v$ gives the complete mapping $\psi: V(G) \rightarrow V(H)$. It remains to show that if $(v, w) \in E(G)$, then $(\psi(v), \psi(w)) \in E(H)$. To see this, observe that for every $e_i = (v, w) \in E(G)$ the edge $(u_p, u_q) = M(e_i)$ exists in $E(H)$, or else we would not be able to separate the Type-1 request pair for $e_i$. From the way we assign $\psi(v)$, it follows that $\psi(v) = u_p$ and $\psi(w) = u_q$. Therefore, $(\psi(u_p), \psi(u_q)) \in E(H)$.

We will now extend the above construction $(S, A, P)$ to the special case when $P$ consists of all pairs of points in $A$. We do this by adding $z$ special obstacles called barriers (one for each block $B_r$) and one master point $a_0 = (0, 4)$ that lies to the outside of rectangle $R$ enclosing all obstacles. Each barrier $S_r$ around block $B_r$ is an inverted U-shaped obstacle that is coincident with the left, top and bottom boundaries of $B_r$. More precisely, obstacle $S_r$ consists of three segments: a vertical segment from $(r - 1, 0)$ to $(r - 1, 3)$, a horizontal segment from $(r - 1, 3)$ to $(r, 3)$ and then a vertical segment from $(r, 3)$ to $(r, 0)$.

Let $S_0$ be the set of all barrier obstacles added above, we prove the following lemma.

Lemma 45. There exists a solution with $|E(G)| + 1$ obstacles for the Generalized Points-separation instance $(S, A, P)$ constructed before if and only if there exists a solution with $|E(G)| + 1 + |S_0|$ obstacles for the Points-separation instance $(S \cup S_0, A \cup a_0)$.

Proof. ($\Rightarrow$) Add all the barriers $S_0$ to the solution for Generalized Points-separation. All points that lie in the same block are already separated. Any pair of points that lie in different blocks are separated due to the barrier obstacles $S_0$, which also separate every point in $A$ from the master point $a_0$.

($\Leftarrow$) The only way to separate the master point $a_0$ from point $a_r$ in block $B_r$ is to select the corresponding barrier $S_r$. Therefore, every solution must select all obstacles in $S_0$. Since the set $S_0$ does not separate any within-block request pair, the remaining set of $|E(G)| + 1$ non-barrier obstacles must separate all request pairs in $P$. 

CVIT 2016
Using Lemma 45 along with Lemma 44, Observation 42 and applying Theorem 41, we obtain the following result for Points-separation.

> **Theorem 46.** Unless ETH fails, a Points-separation instance \((S, A)\) cannot be solved in \(f(k)n^{o(k/\log k)}\) time where \(n = |S|\) and \(k = |A|\).

### 8.2 Hardness for Pseudodisk Obstacles

For the case of pseudodisk obstacles, we will give a reduction from Planar Multiway Cut problem: given an undirected planar graph \(G\) with a subset of \(k\) vertices specified as terminals, the task is to find a set of edges having minimum total weight whose deletion pairwise separates the \(k\) terminal vertices from each other. We will use another result by Marx [24] which showed that unless ETH fails, Planar Multiway Cut cannot be solved in \(f(k) \cdot n^{o(\sqrt{k})}\) time. The result also holds when each edge has unit weight, which is the case we will reduce from.

**Our Construction**

We first fix an embedding of the planar graph \(G\) and consider its dual graph \(G^*\). Then we create an instance \((\hat{S}, A)\) of Points-separation as follows. (See also Figure 6.)

![Figure 6](image_url) An example construction with pseudodisks. (a) The primal graph \(G\) and dual graph \(G^*\) are shown. The obstacles \(S\) move along the dual edges and overlap at the square markers. The terminals of \(G^*\) which form the point set \(A\) are shown in bold. (b) An illustration of how the two obstacles for the dual edge \((v_1^*, v_2^*)\) overlap is shown enlarged for clarity.

- **Adding obstacles.** For every edge \(e_{ij}^* = (v_i^*, v_j^*) \in E(G^*)\), we add two obstacles \(S_i^j, S_j^i\) such that \(S_i^j\) encloses the dual vertex \(v_i^*\) and extends halfway along \(e_{ij}^*\). Similarly, \(S_j^i\) encloses the dual vertex \(v_j^*\) and extends halfway along \(e_{ij}^*\) until it meets obstacle \(S_i^j\).

- **Adding points.** For each terminal \(t_i\), which is a vertex of the primal graph \(G\), add a point \(a_i\) with same coordinates as that of \(t_i\) in the embedding.

Observe that any pair of obstacles either overlap at their source vertex or at the middle of an edge, but not at both places. Therefore, no pair of obstacles intersect more than once and the construction can be realized with only pseudodisk obstacles. The following lemma establishes the correctness of our reduction.

> **Lemma 47.** There exists a solution to Planar Multiway Cut with \(m\) edges if and only if the Points-separation instance constructed above has a solution of size \(2m\).

**Proof.** For the forward direction, consider any pair of terminals \(t_x, t_y\) – since they are separated by the cut edges \(E_c\), there must be a cycle in the dual graph separating \(t_x, t_y\) and only consisting of dual of cut edges \(E^*_c\). Repeating this for every pair of terminals gives a family of separating cycles consisting only of edges \(E^*_c\). It is easy to verify that replacing each
dual edge $e_{ij}^*$ with its obstacle pair $S_i^i, S_j^j$ will also separate every point pair corresponding to the terminals.

For the other direction, given a solution $S'$ for Points-separation, we can draw curves in the plane that separate every point pair and lie in the union of $S'$. We can assume that the solution is exclusion-wise minimal, so every time we arrive inside an obstacle at vertex $v_i^*$, we must continue along an edge $e_{ij}^*$ where we must transfer to the other sibling obstacle $S_j^j$ for $e_{ij}^*$. Using these dual edges, we can construct a solution to Planar Multiway Cut of cost $|S'|/2$.

Since Planar Multiway Cut cannot be solved in $f(k)n^{o(\sqrt{k})}$ time assuming ETH, we obtain the following result.

**Theorem 48.** Unless ETH fails, a Points-separation instance $(S, A)$ with pseudodisk obstacles cannot be solved in $f(k)n^{o(\sqrt{k})}$ time where $n = |S|$ and $k = |A|$.

It is not difficult to see that the above construction can also be realized using only unit disks. In particular, we can replace each pseudodisks with a chain of unit disks and achieve the same result.

### 9 Hardness of Approximation

We will now switch our focus from exact algorithms to approximation algorithms for Points-separation with obstacles $S$ and input points $A$. Gibson et al. [15] gave a constant factor approximation algorithm for Points-separation when obstacles are pseudodisks. However, not much is known for more general obstacle shapes, other than a factor $O(|A|)$-approximation that readily follows from the natural extension of their algorithm for pseudodisks. In this section, we show that assuming the so-called Dense vs Random conjecture, Points-separation is significantly harder to approximate for general obstacle shapes. In particular, we show that assuming Dense vs Random, it is not possible to approximate Points-separation within a factor $|A|^{1/2-\epsilon}$ or $|S|^{3-2\sqrt{d}-\epsilon}$ for any $\epsilon > 0$.

We begin by first stating Dense vs Random, a well-known complexity-theoretic assumption about the hardness for the densest $k$-subgraph problems.

**Conjecture 49 (Dense vs Random [9]).** For all $0 < \alpha, \beta < 1$ with $\beta < \alpha - \epsilon$ for sufficiently small $\epsilon > 0$, and function $k : \mathbb{N} \rightarrow \mathbb{N}$ so that $k(n)$ grows polynomially with $n$, $(k(n))^{1+\beta} \leq n^{(1+\alpha)/2}$, there does not exist an algorithm ALG that takes as input an $n$-vertex graph $G$, runs in polynomial time, and outputs either dense or sparse, such that:

- For every graph $G$ that contains an induced subgraph on $k = k(n)$ vertices and $k^{1+\beta}$ edges, ALG($G$) outputs dense with high probability.
- If $G$ is drawn from $G(n, p)$ with $p = n^{\alpha-1}$ then ALG($G$) outputs sparse with high probability.

The conjecture was originally stated in [9] but the formalization of the conjecture as stated above is borrowed from [25]. In order to obtain hardness guarantees for our problem using Conjecture 49, we will describe a reduction that given a graph $G$ constructs an instance of Points-separation. Then we show that the images of dense instances under this reduction will have (with high probability) optimum at most $x_2^*$, whereas the images of random instances will have optimum at least $x_1^*$, where $x_1^*$ is much bigger than $x_2^*$. Let $\rho = x_1^*/x_2^*$ be the distinguishing ratio of the reduction, then an approximation algorithm for Points-separation with ratio smaller than $\rho$ can now (with high probability) distinguish between the images of dense and random instances, thereby refuting Conjecture 49. This gives us the following lemma.
Lemma 50. If there exists a reduction with distinguishing ratio \( \rho \), then, assuming DENSE vs RANDOM, there is no polynomial time approximation algorithm for POINTS-SEPARATION with approximation ratio less than \( \rho \).

Our construction is inspired from a similar construction using DENSE vs RANDOM for the related MIN-COLOR PATH problem from [25]. Specifically, we borrow the idea of partitioning the edges of graph \( G = (V,E) \) into \( z \) groups \( E_1,E_2,\ldots,E_z \), by assigning every edge to one of the groups with probability \( 1/z \) independent of other edges. We have the following lemma.

Lemma 51 (Lemma 7.3 [25]). For any graph \( G = (V,E) \), there exists a partitioning of edges into \( z = \frac{q}{2ln n} \) groups such that for any set \( E^* \subseteq E \) of \( q \) edges, every group \( E_i \in \{E_1,E_2,\ldots,E_z\} \) contains an edge from \( E^* \).

We will also need the following bound on the size of a subgraph of \( G(n,p) \).

Lemma 52 (Lemma 7.2 [25]). Let \( G \) be drawn from \( G(n,p) \). Then, with high probability, every subgraph of \( G \) with \( q = n^{\Omega(1)} \) edges contains \( \Omega(\min\{q,\sqrt{(q/p)}\}) \) vertices. Here \( \Omega \) ignores logarithmic factors.

Our Construction

Given a graph \( G = (V,E) \) and fixed \( \alpha,\beta \) and function \( k : \mathbb{N} \to \mathbb{N} \) satisfying conditions of Conjecture 49, we will construct an instance of POINTS-SEPARATION as follows.

1. Fix \( q = k^{1+\beta} \) and \( z = \frac{q}{2ln n} \). Using Lemma 51, partition the set of edges of \( G \) into \( z \) groups \( \{E_1,E_2,\ldots,E_z\} \).
2. Similar to the hardness construction in Section 8, all the request pairs and obstacles are contained in an enclosing rectangle \( R \) with bottom left corner \((0,0)\) and top-right corner \((z,4)\).
3. For every \( v_i \in V \), add an obstacle \( S_i \) to \( S \). Initially, all obstacles are horizontal line segments occupy the part of \( x \)-axis from \( x = 0 \) to \( x = z \).
4. Define two set of horizontal lines \( \ell_1^h : y = 1 + \frac{h}{|E|+1} \) and \( \ell_2^h : y = 3 + \frac{h}{|E|+1} \) to be a horizontal line that will serve as guardrails for obstacle growth corresponding to edge \( e_h \in E \). Here \( 1 \leq h \leq |E| \). We will refer to the group \( \ell_1^h, \ell_2^h \) lines as \( \ell_1\)-channel and \( \ell_2\)-channel respectively.
5. For each group \( E_r \), define a request pair block \( B_r \), which is a unit-width sub-rectangle of \( R \) bounded by vertical sides \( x = r-1 \) and \( x = r \). Let \( mid_r = (r-\frac{1}{2}) \) and add the pair of points \( a_r = (\text{mid}_r,\frac{1}{2}) \) and \( a_r' = (\text{mid}_r,\frac{1}{2}) \) to \( A \). These points will be contained in block \( B_r \).

Now for every edge \( e_h = (v_i,v_j) \in E_r \) with \( i < j \), we grow the obstacles along \( \ell_1, \ell_2 \)-channels as follows. (See also Figure 7.)

- Grow the obstacle \( S_i \) corresponding to vertex \( v_i \) vertically along left boundary \( x = r-1 \) of \( B_r \) until \( y = 4 \). Similarly grow \( S_j \) along right boundary \( x = r \) of \( B_r \) until \( y = 4 \).
- Moving along the horizontal line \( \ell_1^h \) from left to right, extend obstacle \( S_i \) from \( x = r-1 \) to \( x = \text{mid}_r \). Repeat the same for \( \ell_2^h \).
- Similarly, moving along the horizontal line \( \ell_1^h \) from right to left, extend obstacle \( S_j \) from \( x = r \) to \( x = \text{mid}_r \). Repeat the same for \( \ell_2^h \).

Lemma 53. Let \( S^* \subseteq S \) be a solution to the POINTS-SEPARATION instance \((S,A)\) constructed above. Then all point pairs in \( A \) are separated if and only if for every request pair block \( B_r \), there exists two obstacles \( S_i, S_j \in S^* \) such that \((v_i,v_j)\) is an edge assigned to group \( E_r \).
Figure 7: An an group of edges $E_1$ and the resulting Points-separation request pair block $B_1$. The $\ell_1$-channel is shown enlarged in the rightmost figure. As an example, observe that point pair $(a, a')$ is separated if obstacles $S_1, S_2$ are selected (because $(v_1, v_2) \in E_1$) but not separated if obstacles $S_2, S_3$ are selected (because $(v_2, v_3) \notin E_1$).

Proof. For the forward direction, suppose we start moving vertically in block $B_\epsilon$ along $x = \text{mid}$, starting from $a'_r$ towards $a_r$. Before we reach point $a_r$, we must cross the lines $\ell_h$ for all $h$ such that $e_h \in E_r$. Whenever we arrive at $\ell_h$, which is the guardrail corresponding to edge $e_h = (v_i, v_j)$, if either $S_i \notin S^*$ or $S_j \notin S^*$, then we can cross over $\ell_h$ without intersecting an obstacle in $S^*$ by shifting infinitesimally to the left (or right) from $x = \text{mid}$. Since $S^*$ separates $a_r, a'_r$, there must be some $e_h = (v_i, v_j)$ with $i < j$, such that both $S_i, S_j \in S^*$.

For the other direction, if obstacles $S_i, S_j \in S^*$ such that $(v_i, v_j) \in E_r$, then the union of $S_i, S_j$ forms a closed curve enclosing both $a_r$ and $a'_r$ and therefore separates $a_r, a'_r$ from each other as well as from other points in $A$.

Using the discussion preceding Lemma 50, we can obtain a lowerbound for the distinguishing ratio $\rho$ of the above reduction as follows.

Lemma 54. Let $(S, A)$ be the resulting Points-separation instance obtained by applying the above reduction to a graph $G$. Then we have distinguishing ratio:

1. $\rho \geq \min \left\{ k^\beta, \sqrt{k^{\beta - 1} \cdot n^{1 - \alpha}} \right\}$ in terms of $n, k$
2. $\rho \geq \frac{\min \{ q, \sqrt{n^{1-\alpha}} \}}{q^{1/(\beta + 1)}}$ in terms of $n, q$.

Proof. We have the following two cases for the instance $(S, A)$ depending on graph $G$.

- $G$ contains a subgraph on $k$ vertices and $q = k^{\beta + 1}$ edges. Let $E^*$ be the set of these edges. Using Lemma 51, it follows that every group $E_r$ contains an edge $e_h \in E^*$. Using the obstacles corresponding to vertices in $E^*$ and applying Lemma 53, we obtain a set of at most $k$ obstacles that separate the request pair $(a_r, a'_r)$ in every block $B_r$. Therefore, the number of obstacles used in this case $x_d^* \leq k$.

- $G$ is drawn from $G(n, p)$ with $p = n^{\alpha - 1}$. From Lemma 53, it follows that to separate $(a_r, a'_r)$ in any block $B_r$, any solution must select both obstacles corresponding to at least one edge in $B_r$. Choosing one edge from each block, we obtain a subgraph of $G$ with $z$ edges. Applying Lemma 52 on this subgraph and observing that $z = \tilde{O}(q)$ gives the number of obstacles used in this case $x_u^* \geq \tilde{O}(\min \{ q, \sqrt{q/p} \})$.

Taking the ratio of solution sizes in both cases and substituting the values $p = n^{\alpha - 1}$ and $q = k^{\beta + 1}$, we obtain:

$$\rho = \frac{x_r^*}{x_d^*} \geq \frac{\min \left\{ k^{\beta + 1}, \sqrt{\frac{k^{\beta + 1}}{n^{1-\alpha}}} \right\}}{k} = \min \left\{ k^\beta, \sqrt{k^{\beta - 1} \cdot n^{1-\alpha}} \right\}$$
Similarly, in terms of \(q,n\), we obtain the following:

\[
\rho \geq \frac{x_t}{x_d} \geq \min \left\{ \frac{q}{k}, \frac{\sqrt{q \cdot n^{1-\alpha}}}{k} \right\} = \frac{\min \left\{ q, \frac{\sqrt{q \cdot n^{1-\alpha}}}{k} \right\}}{q^{1/(\beta+1)}}
\]

We will now fix the choice of parameters \(\alpha, \beta, k\) such that they satisfy the requirements of Conjecture 49 and obtain a bound on the distinguishing ratio in terms of number of obstacles \(|S| = n\) and number of points \(|A| = 2z = \Theta(q)\). The parameters are carefully chosen so that the maximize the distinguishing ratio and therefore obtain best possible lowerbound on the hardness of approximation.

\[\text{Lemma 55. Assuming Dense vs Random, a Points-separation instance } (S, A) \text{ cannot be approximated to a factor better than } |A|^{1/2 - \epsilon} \text{ in polynomial time, for any } \epsilon > 0.\]

\[\text{Proof. Let } \alpha = 1 - \epsilon \text{ and } \beta = \alpha - \epsilon \text{ and } q = n^{1-\alpha} = n^\epsilon. \text{ Since, } k^{\beta+1} = q, \text{ we have } k^{\beta+1} = n^\epsilon < n^{(1+\alpha)/2}. \text{ Therefore the parameters } \alpha, \beta, k \text{ satisfy the conditions of Conjecture 49. Since } q = n^{1-\alpha}, \text{ we have } \min \left\{ q, \frac{\sqrt{q \cdot n^{1-\alpha}}}{k} \right\} = q. \text{ Substituting this to the equation for } \rho \text{ in terms of } n,q \text{ from Lemma 54, we obtain:}
\]

\[
\rho \geq \frac{q}{q^{1/(\beta+1)}} = q^{-\epsilon} = q^{(1-2\epsilon)/(2\epsilon)} = q^{\frac{1-\epsilon}{2\epsilon} - \frac{\epsilon}{2\epsilon}} = q^{1/2 - \epsilon'}
\]

where \(\epsilon' = \frac{\epsilon}{2\epsilon}\). Since \(|A| = \Theta(q)|, applying Lemma 50, we achieve the claimed bound.\]

\[\text{Lemma 56. Assuming Dense vs Random, a Points-separation instance } (S, A) \text{ cannot be approximated to a factor better than } |S|^{3-2\sqrt{2} - \epsilon} \text{ in polynomial time, for any } \epsilon > 0.\]

\[\text{Proof. For this case, we set both } \alpha = \sqrt{2} - 1 \text{ and } k = n^{3-1}. \text{ With } \beta = \alpha - \epsilon, \text{ we have } k^{\beta+1} = n^{3-\sqrt{2}} < n^{(1+\alpha)/2} \text{ which satisfies the requirements of Conjecture 49.}
\]

Therefore, we have:

\[
k^{\beta} = n^{(\sqrt{2}-1)(\beta-1)} = n^{3-2\sqrt{2} - \epsilon'} \quad \text{for some } \epsilon' > 0
\]

\[
\sqrt{k^{\beta-1} \cdot n^{1-\alpha}} = \left( n^{(\sqrt{2}-1)(\beta-1)}, n^{(\sqrt{2}-1)(\sqrt{2}-1)} \right)^{1/2} = n^{\frac{(\sqrt{2}-1)(2\sqrt{2}-2-\epsilon)}{2}} \quad \text{for some } \epsilon'' > 0
\]

Substituting this to the equation for \(\rho\) in terms of \(n,k\) from Lemma 54 and applying Lemma 50, we achieve the claimed bound.

We conclude with the main result for this section.

\[\text{Theorem 57. Assuming Dense vs Random [9], one cannot approximate Points-separation within ratio } n^{3-2\sqrt{2} - \epsilon} \text{ or } m^{1/2 - \epsilon} \text{ in polynomial time, for any } \epsilon > 0, \text{ where } n \text{ is the number of obstacles and } m \text{ is the number of points.}\]

\[\text{References}\]

1. A. Agarwal, M. Charikar, K. Makarychev, and Y. Makarychev. O (\(\sqrt{\log n}\)) approximation algorithms for Min UnCut, Min 2CNF Deletion, and directed cut problems. In Proc. of 37th STOC, pages 573–581, 2005.
Amit Agarwal, Moses Charikar, Konstantin Makarychev, and Yury Makarychev. O(\sqrt{\log n}) approximation algorithms for min uncut, min 2cnf deletion, and directed cut problems. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, Baltimore, MD, USA, May 22-24, 2005, pages 573–581, 2005.

Paul Balister, Zizhan Zheng, Santosh Kumar, and Prasun Sinha. Trap coverage: Allowing coverage holes of bounded diameter in wireless sensor networks. In *IEEE INFOCOM 2009*, pages 136–144. IEEE, 2009.

Sayan Bandyapadhyay, Neeraj Kumar, Subhash Suri, and Kasturi Varadarajan. Improved approximation bounds for the minimum constraint removal problem. *Computational Geometry*, 90:101650, 2020.

Sergey Bereg and David G. Kirkpatrick. Approximating barrier resilience in wireless sensor networks. In *Proc. of 5th ALGOSENSORS*, volume 5804, pages 29–40, 2009.

S. Cabello and P. Giannopoulos. The complexity of separating points in the plane. *Algorithmica*, 74(2):643–663, 2016.

David Yu Cheng Chan and David G. Kirkpatrick. Approximating barrier resilience for arrangements of non-identical disk sensors. In *Proc. of 8th ALGOSENSORS*, pages 42–53, 2012.

David Yu Cheng Chan and David G. Kirkpatrick. Multi-path algorithms for minimum-colour path problems with applications to approximating barrier resilience. *Theor. Comput. Sci.*, 553:74–90, 2014.

Eduard Eiben and Iyad Kanj. How to navigate through obstacles? In *Proc. of 45th ICALP*, 2018.

Eduard Eiben, Jonathan Gemmell, Iyad A. Kanj, and Andrew Youngdahl. Improved results for minimum constraint removal. In *Proc. of 32nd AAAI*, pages 6477–6484, 2018.

Eduard Eiben and Iyad Kanj. A colored path problem and its applications. *ACM Trans. Algorithms*, 16(4):47:1–47:48, 2020.

Eduard Eiben and Daniel Lokshtanov. Removing connected obstacles in the plane is FPT. In *Proc. of 36th SoCG*, volume 164, pages 39:1–39:14, 2020.

Lawrence H. Erickson and Steven M. LaValle. A simple, but NP-Hard, motion planning problem. In *Proc. of 27th AAAI*, 2013.

Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

Klara Kedem, Ron Livne, János Pach, and Micha Sharir. On the union of jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete & Computational Geometry*, 1(1):59–71, 1986.

Matias Korman, Maarten Löffler, Rodrigo I. Silveira, and Darren Strash. On the complexity of barrier resilience for fat regions and bounded ply. *Comput. Geom.*, 72:34–51, 2018.

Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. *Journal of the ACM (JACM)*, 67(3):1–50, 2020.

Santosh Kumar, Ten-Hwang Lai, and Anish Arora. Barrier coverage with wireless sensors. *Wirel. Networks*, 13(6):817–834, 2007.

James R. Lee. Separators in region intersection graphs. In *Proc. of 8th ITCS*, volume 67, pages 1–8, 2017.

Daniel Lokshtanov, NS Narayanaswamy, Venkatesh Raman, MS Ramanujan, and Saket Saurabh. Faster parameterized algorithms using linear programming. *ACM Transactions on Algorithms (TALG)*, 11(2):1–31, 2014.

Danieł Marx. Can you beat treewidth? In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS’07)*, pages 169–179. IEEE, 2007.
24 Dániel Marx. A tight lower bound for planar multiway cut with fixed number of terminals. In *International Colloquium on Automata, Languages, and Programming*, pages 677–688. Springer, 2012.

25 Saket Saurabh Neeraj Kumar, Daniel Lokshtanov and Subhash Suri. A constant factor approximation for navigating through connected obstacles in the plane. In *Proc. 32nd SODA*, 2021.