Comput. Geom. 90 (2020) 101650

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Improved approximation bounds for the minimum constraint removal problem [☆]

Sayan Bandyapadhyay [a,*,1], Neeraj Kumar [b], Subhash Suri [b], Kasturi Varadarajan [a]

[a] Department of Computer Science, University of Iowa, Iowa City, USA
[b] Department of Computer Science, University of California, Santa Barbara, USA

ARTICLE INFO

Article history:
Received 3 August 2019
Received in revised form 17 February 2020
Accepted 28 March 2020
Available online 2 April 2020

Keywords:
Minimum constraint removal
Minimum color path
Barrier resilience
Obstacle removal
Obstacle free path

ABSTRACT

In the minimum constraint removal problem, we are given a set of overlapping geometric objects as obstacles in the plane, and we want to find the minimum number of obstacles that must be removed to reach a target point t from the source point s by an *obstacle-free* path. The problem is known to be intractable and no sub-linear approximations are known even for simple obstacles such as rectangles and disks. The main result of our paper is an approximation framework that gives an $O(\sqrt{n\alpha(n)})$-approximation for polygonal obstacles, where $\alpha(n)$ denotes the inverse Ackermann's function. For pseudodisks and rectilinear polygons, the same technique achieves an $O(\sqrt{n})$-approximation. The technique also gives $O(\sqrt{n})$-approximation for the *minimum color path* problem in graphs. We also present some inapproximability results for the geometric constraint removal problem.

1. Introduction

Given a set $\mathcal{S}$ of geometric objects as obstacles in the plane, a path is called *obstacle-free* if it does not intersect the interior of any obstacle. In the *minimum constraint removal* (MCR) problem, the goal is to remove a minimum-sized subset $\mathcal{S}' \subseteq \mathcal{S}$ such that the remaining set $\mathcal{S} \setminus \mathcal{S}'$ admits an obstacle-free path between a source point s and the target point t. The problem is known to be NP-complete even when the obstacles have very simple geometry such as rectangles or line segments. The MCR problem is also related to the *minimum color path* (MCP) problem where the goal is to find a path in a graph using the minimum number of colors. In the vertex-colored version of the problem, each vertex v of a graph $G = (V, E)$ is associated with a set of colors $\chi(v) \subseteq \mathcal{C} = \{1, 2, \ldots, |\mathcal{C}|\}$, and the goal is to find a path between two fixed vertices s and t (s-t path) that minimizes the total number of colors along the path. Similarly, in the edge-colored version, each edge of G has an associated set of colors, and the s-t path must minimize the total number of colors along the path.

The geometric constraint removal problem can be cast as a minimum color path problem by constructing a graph on the *arrangement* formed by the obstacles. The arrangement $\mathcal{A}(\mathcal{S})$ of $\mathcal{S}$ is a partition induced by the obstacles, whose

☆ A preliminary version of this paper appeared in the Proceedings of the 21st International Conference on Approximation Algorithms for Combinatorial Optimization Problems (APPROX), 2018. The work of Bandyapadhyay, Kumar and Suri was supported in part by National Science Foundation grants CCF-1525817 and CCF-1814172; work of Bandyapadhyay and Varadarajan was supported by National Science Foundation grant CCF-1615845.

* Corresponding author.
E-mail addresses: sayan-bandyapadhyay@uiowa.edu (S. Bandyapadhyay), neeraj@cs.ucsb.edu (N. Kumar), suri@cs.ucsb.edu (S. Suri), kasturi-varadarajan@uiowa.edu (K. Varadarajan).
[1] The work was partially done when the author was visiting University of California, Santa Barbara.

https://doi.org/10.1016/j.comgeo.2020.101650
0925-7721/

Table 1
Table showing the best known approximation and inapproximability results after our work for different versions of MCP and MCR. Question mark denotes that result is unknown.

Problem	Version/Objects	Approximation	Inapproximability
MCP	Vertex-colored	$O(\sqrt{\lvert V\rvert})$ (Theorem 2)	$o(\log\lvert V\rvert)$ [3]
	Edge-colored	$\lvert\mathcal{C}\rvert$ (trivial)	
MCR	Polygons	$O(\sqrt{n\alpha(n)})$ (Corollary 5)	$2-\epsilon$ (Theorem 6)
	Convex polygons	$O(\sqrt{n\alpha(n)})$ (Corollary 5)	$2-\epsilon$ (Lemma 7)
	Rectilinear polygons	$O(\sqrt{n})$ (Corollary 5)	$2-\epsilon$ (Theorem 6)
	Rectangles	$O(\sqrt{n})$ (Corollary 5)	APX-hard (Theorem 7)
	Pseudodisks	$O(\sqrt{n})$ (Theorem 3)	?
	Unit disks	2 [5]	?

faces are the two-dimensional connected regions, and whose edges are segments of the obstacle boundaries. We define a planar graph $G_{\mathcal{A}}$ whose vertices are in one-to-one correspondence with the faces of the arrangement and whose edges join two neighboring faces. By associating each obstacle with a unique color, we obtain a version of the minimum color path problem—an s–t path has exactly as many colors along it as the number of obstacles it crosses. However, it is worth pointing out that the number of vertices in $G_{\mathcal{A}}$ can be *quadratic* in the size of the geometric input: a set of n geometric obstacles, each with a constant number of boundary edges, can create an $\Omega(n^2)$ size arrangement.

The minimum color path problem (both versions) is NP-complete and also NP-hard to approximate to a factor $o(\log|V|)$ even if the graph is planar [1–3]. In [4], Hassin et al. present an $O(\sqrt{|V|})$-approximation algorithm for the special case of the MCP problem where each edge has exactly one color. Indeed, their algorithm works for a more general weighted version. However, for the general MCP problem, no sub-linear approximation algorithm appears to be known.

The geometric minimum constraint removal problem has been studied under different names across multiple research communities, including sensor networks and robotics. In networks, the problem is called barrier resilience where obstacles represent coverage regions of sensors, and the network's resilience is measured by the minimum number of sensors whose removal creates a sensor-avoiding s–t path. Circular disks are a commonly used model for sensor coverage. When all disks have the same (unit) radius, a 2-approximation is known due to Chan and Kirkpatrick [5], who build and improve upon the earlier work of Bereg and Kirkpatrick [6]. However, even for this simple case, the complexity of minimum constraint removal is an unsolved open problem [5]. In [5,7], constant factor approximations are proposed for restricted versions of arbitrary radii disks. However, when disks have arbitrary radii, no sub-linear approximation with provable guarantee is known. The problem has also been studied for other types of obstacles, mainly from the perspective of time complexity. The problem has been shown to be NP-complete for convex obstacles [8], for line segments [9], even in the bounded density case [10,11], and for axis-parallel rectangles with aspect ratio close to one [7].

In robotics, the minimum constraint removal problem models the motion planning problem of multi-articulated robot [2,8]. Suppose we have a physical environment constraining a disjoint set of impenetrable obstacles in the plane, and a robot with two degrees of freedom. Then the configuration space approach to motion planning shrinks the robot to a point while simultaneously expanding the obstacle by taking their Minkowski sum with the robot's geometry. The result is our minimum constraint removal problem: a set of two-dimensional intersecting obstacles that may have no feasible path for the robot, and so some obstacles need to be removed.

Finally, the problem has also been studied through the lenses of parameterized complexity [11,7], and exact and heuristic algorithms [2,12]. It is also loosely related to a shortest path problem in the plane [13,14], where given a set of disjoint obstacles, the goal is to find an Euclidean shortest s-t path that intersects at most k obstacles.

1.1. Our results

In this work, we make progress on both the graph and the geometric versions of MCP and obtain improved polynomial-time approximation bounds. In particular, we present the following results for the minimum constraint removal problem.

- An $O(\sqrt{n\alpha(n)})$-approximation for polygonal objects, where $\alpha(n)$ is the inverse Ackermann's function and n denotes the total number of vertices of the polygons. We also present an $O(\sqrt{n})$-approximation for rectilinear polygons.
- An $O(\sqrt{n})$-approximation for disks, where n denotes the number of disks. For arbitrary disks, the only approximation results known are in the restricted cases, where either the crossing patterns of the paths are limited or the aspect ratio and the density are bounded. Our result can be easily extended to pseudodisks.

Our approximation framework also leads to an $O(\sqrt{|V|})$-approximation for MCP on vertex-colored graphs. Finally, we also present some results showing the hardness of approximating MCR. In particular, we show that the problem is NP-hard to approximate within a factor better than 2 for either rectilinear or convex polygons. We also prove the APX-hardness of the problem in a more restricted case, where the obstacles are axis-parallel rectangles. Table 1 shows the state of the art results after our work on MCP and MCR.

We describe our approximation framework in Section 2. The application of the framework to the MCR problem is discussed in Section 3. Finally we describe the hardness results in Section 4.

2. An algorithmic framework

We begin our discussion by describing the generic framework for approximating the minimum color path problems on graphs. In the later sections, we apply this framework to achieve similar approximation bounds for the MCR problem. Roughly speaking, the framework comprises of two main steps. In the first step, a 'small' subset of the colors are removed, and in the second stage, an approximation of the minimum color path is computed using a shortest path algorithm. We start with some basic definitions.

We assume that we are given a graph $G = (V, E)$, the source vertex s, the target vertex t, and a set of colors $\mathcal{C}$, such that each vertex $v \in V$ is assigned a subset $\chi(v) \subseteq \mathcal{C}$ of colors. We will refer to such a graph as a *colored graph* and denote it by $G = (V, E, \mathcal{C})$. We define the set of colors $\chi(\pi)$ used by π to be the union of the colors of vertices on this path. That is, $\chi(\pi) = \bigcup_{v\in\pi} \chi(v)$.

Definition 1. A path π in G is a *k-color path* if $|\chi(\pi)| = k$.

An algorithm is called an *r-approximation* algorithm for computing a k-color path if it satisfies the following two conditions: (1) if G has a k-color s-t path, then the algorithm returns a path π^* with $|\chi(\pi^*)| \le rk$, and (2) if G does not have a k-color s-t path, the algorithm returns an arbitrary s-t path. The following is straightforward.

Lemma 1. *If there exists an $r(k)$-approximation algorithm to compute a k-color path from s to t, then there also exists an $r(l)$-approximation algorithm for computing a minimum color path from s to t, where l is the number of colors used by a minimum color path.*

Proof. We try all possible values $k = 1, 2, \ldots, |\mathcal{C}|$ and let π_k be the path returned by the approximation algorithm for computing a k-color path for a given value of k. Let j be the value such that $\chi(\pi_j)$ has smallest cardinality over all $\chi(\pi_k)$. Recalling that l is the number of colors used by a minimum color path, we have that $|\chi(\pi_l)|$ is at most $r(l)l$. Clearly, $|\chi(\pi_j)| \le |\chi(\pi_l)| \le r(l)l$ and therefore π_j is an $r(l)$-approximation for computing a minimum color path. □

From Lemma 1 it follows that computing an approximation of a k-color path is sufficient, and therefore in the rest of our discussion, we work towards that goal. Next, we describe the details of our framework.

2.1. Approximation framework

We are given a colored graph $G = (V, E, \mathcal{C})$ and an integer k, and our goal is to develop an approximation algorithm for computing a k-color path in G. The key idea behind our approximation framework is to define a notion of *neighborhood* for the colors in $\mathcal{C}$, and 'discard' the colors that have dense neighborhoods.

Definition 2. Let $\mathcal{P}$ be an arbitrary set of objects and β be a parameter. We define *neighborhood* $\mathcal{N} : \mathcal{C} \to 2^{\mathcal{P}}$ to be a mapping from $\mathcal{C}$ to subsets of $\mathcal{P}$ that satisfies the following properties.

1. *(**Bounded-Size Property**)* The size of $\mathcal{N}$, defined to be the sum of cardinalities of all neighborhoods, $\sum_{C\in\mathcal{C}} |\mathcal{N}(C)|$, is $O(k\beta^2)$
2. *(**Bounded-Occurrence Property**)* If there exists a k-color path in G, then there also exists a k-color path π^* in G such that, for any color $C \in \mathcal{C}$, the number of times that C appears on π^* is at most $O(|\mathcal{N}(C)|)$.

The set $\mathcal{P}$ in the above definition can be any set of objects. For example, in MCP problem $\mathcal{P}$ is the set of vertices of the graph. In the geometric MCR, $\mathcal{P}$ is a set of points in the plane. We now describe our approximation algorithm which we will refer to as Approx-Core.

Algorithm Approx-Core

1. Construct the neighborhood $\mathcal{N}(C)$ for each color $C \in \mathcal{C}$.
2. For all $C \in \mathcal{C}$, remove all occurrences of the color C from the graph G if $|\mathcal{N}(C)| \ge \beta$. Let G' be the modified graph after removing all such colors.
3. For every vertex v in G', assign an integer weight $|\chi(v)|$ on v.
4. Compute a minimum weight path π from s to t in G' using Dijkstra's Algorithm. Return π.

Lemma 2. *Given the set $\mathcal{P}$ and a parameter β, the algorithm APPROX-CORE gives an $O(\beta)$-approximation for the k-color path in G.*

Proof. Assume that there exists a k-color path in G. Otherwise, the proof is trivial as the algorithm always returns a path. Let $\mathcal{C}_1$ be the set of colors removed during step 2 of the algorithm, and $\mathcal{C}_2$ be the set of colors in G' that appear on the path π returned by the algorithm. Then, the total number of colors in G that may appear on π is at most $|\mathcal{C}_1| + |\mathcal{C}_2|$.

First we compute a bound on the size of $\mathcal{C}_1$. Observe that the neighborhood of each color $C \in \mathcal{C}_1$ has size at least β. Therefore, we have:

$$\begin{aligned} \sum_{C \in \mathcal{C}_1} |\mathcal{N}(C)| &\leq \sum_{C \in \mathcal{C}} |\mathcal{N}(C)| \\ \implies |\mathcal{C}_1| \cdot \beta &\leq O(k\beta^2) && \text{By the bounded-size property of } \mathcal{N} \\ \implies |\mathcal{C}_1| &\leq ck\beta && \text{for some constant } c \end{aligned}$$

Next, we compute a bound on size of $\mathcal{C}_2$. Towards this end, observe that the neighborhood $\mathcal{N}(C)$ of every color C in G' has fewer than β colors. By the bounded-occurrence property of the neighborhood $\mathcal{N}$, there exists a k-color path π^* in G such that for each $C \in \mathcal{C}$, the number of times that C appears on π^* is at most $O(|\mathcal{N}(C)|)$. Therefore, it follows that any color C in G' appears on π^* at most $O(\beta)$ times. In other words, there exists a path in G' that has weight at most $c'k\beta$ for another constant c'. Therefore the number of colors used by the minimum weight path π is at most $c'k\beta$.

Hence, the total number of colors in $\mathcal{C}$ that appear on π is at most $|\mathcal{C}_1| + |\mathcal{C}_2| = (c + c') \cdot k\beta$, which is an $O(\beta)$-approximation. □

We obtain the following theorem.

Theorem 1. *Given a colored graph $G = (V, E, \mathcal{C})$ and integer k, suppose a neighborhood $\mathcal{N}$ for G can be constructed in polynomial time that satisfies the bounded-size and bounded-occurrence property. Then there exists a polynomial time algorithm that achieves an $O(\beta)$-approximation for computing a k-color path in G.*

Therefore, in order to achieve an approximation for the k-color path, it just suffices to construct a neighborhood $\mathcal{N}$, that satisfies the bounded-size and bounded-occurrence properties. In the next section, we illustrate this construction for MCP on vertex-colored graphs.

2.2. Application to minimum color path

In this section, we will apply the above framework to achieve $O(\sqrt{n})$-approximation for MCP on a vertex-colored graph $G = (V, E, \mathcal{C})$ with n vertices. Our goal is to simply compute a neighborhood $\mathcal{N}$ for a k-color path in G such that $\mathcal{N}$ has bounded-size $O(kn)$ and satisfies the bounded-occurrence property. Using Lemma 1 and $\beta = \sqrt{n}$ in Theorem 1, an $O(\sqrt{n})$-approximation for MCP follows.

We define neighborhood $\mathcal{N}(C)$ of each color C to be the set $\{v \in V \mid C \in \chi(v) \text{ and } |\chi(v)| \leq k\}$. The bounded-occurrence property is easily satisfied because a k-color path π_k will never visit vertices that contain more than k colors, and since π_k is simple, each occurrence of a color C on the path can be uniquely charged to a vertex in $\mathcal{N}(C)$. To see that the bounded-size property is satisfied, we note the following.

$$\sum_{C \in \mathcal{C}} |\mathcal{N}(C)| = \sum_{v \in V : |\chi(v)| \leq k} |\chi(v)| \leq kn.$$

Theorem 2. *There exists an $O(\sqrt{n})$-approximation algorithm for MCP on vertex-colored graphs.*

Application to minimum label path As another example application for the framework, we consider a special case of MCP when each edge has exactly one color (called its label). This problem has been well studied [4,15,16] under the name *minimum label path*. Hassin et al. [4] gave an $O(\sqrt{n})$-approximation for this problem on general graphs.[2] Using our framework and the following simple definition of neighborhood, we can achieve an $O(\sqrt{\frac{n}{OPT}})$-approximation if the number of edges in G is $O(n)$. Here *OPT* is the number of labels used by any minimum label s-t path.

For the sake of applying the framework, we transform the input edge-colored graph $G = (V, E, \mathcal{C})$ into a vertex-colored graph H by adding a vertex corresponding to each edge that subdivides the edge. The color corresponding to an old edge is moved to the new vertex. Now, for each new vertex v that has color C, we include both neighbors (old vertices) of v in

[2] Indeed their algorithm works for a more general version, where each label has a non-negative cost and the goal is to find an s-t path minimizing the combined cost of its labels.

H to the neighborhood of C. The bounded-occurrence property is straightforward. For the bound on size, observe that an old vertex v can be in at most *degree*(v) neighborhoods, so sum of cardinality of all neighborhoods is at most $2|E|$. Since $|E| = O(n)$, the size of $\mathcal{N}$ is $O(n) = O(\frac{n}{k} \cdot k)$. With $\beta = \sqrt{n/k}$, Theorem 1 and Lemma 1 give an $O(\sqrt{\frac{n}{OPT}})$-approximation.

3. Application to geometric objects

We now show how to approximate the MCR problem when the obstacles are geometric objects such as constant-complexity polygons or disks. We point out that we cannot directly apply Theorem 2 to the graph of the arrangement of obstacles since the latter can have size $\Theta(n^2)$. Instead, we first construct a colored graph G such that an s–t path in the plane that removes the minimum number of obstacles corresponds to a path in G that uses the minimum number of colors. We then construct the neighborhood $\mathcal{N}$ for colors in G such that it satisfies the bounded-size and bounded-occurrence properties. For technical reasons, the graph G we construct for the geometric instances has colors assigned on edges—one can easily transform it into a vertex-colored graph by adding a vertex corresponding to each edge.

Throughout this section, we assume that the obstacles are in *general position*, namely, no three obstacle boundaries intersect at a common point, and the boundaries of any two objects intersect transversally.

Any arrangement of obstacles in the plane can be partitioned into two distinct regions namely the obstacles, and *free space*, that is the region of the plane not occupied by obstacles. Without loss of generality, we assume that the points s and t lie in free space, as we must remove all the obstacles that are incident to either s or t in order to find an obstacle free s–t path. We say that a path π *crosses* an obstacle S if π intersects the interior of S. Note that, as s and t lie in free space, if π crosses S, π must intersect the boundary of S transversally.

Consider an optimal path π that removes the minimum number of obstacles. It is easy to see that π will cross an obstacle S if and only if S was removed from input. Therefore, removing an obstacle is equivalent to crossing it. In the following, we introduce the notion of a *k-crossing path*.

Definition 3. A path π in the plane is called a *k-crossing path* if it crosses exactly k obstacles.

It is easy to see that if each obstacle is assigned a unique color and we assign color to a path whenever it enters an obstacle, then a k-crossing path π uses exactly k colors. Observe that although the space of k-crossing paths is infinite, we want to establish a one to one correspondence between the path in the plane that crosses minimum number of obstacles and a path in G that uses the minimum number of colors.

Towards this end, we simply let G to be the "dual" graph of $G_{\mathcal{A}}$ induced by the input arrangement $\mathcal{A}$: each cell C_i of $G_{\mathcal{A}}$ is associated with a vertex v_i of G that is contained in C_i, and any pair of neighboring cells C_i, C_j are joined by the edge $v_i v_j$ that only intersects the shared boundary ∂C_{ij} between the cells C_i and C_j. Note that the edge $v_i v_j$ is not necessarily a straight line segment, it could be a curve segment in some cases. Due to the general position assumption, ∂C_{ij} is part of the boundary of a unique obstacle in $\mathcal{S}$. Therefore, we have that each edge of G intersects the boundary of a unique obstacle and no two obstacles share an edge. Additionally, we make G directed by replacing each edge $\{v_i, v_j\}$ with two directed edges $v_i \to v_j$ and $v_j \to v_i$. Next, we assign colors to edges of G. Each obstacle in $\mathcal{S}$ corresponds to a color in $\mathcal{C}$, so for any edge $e = v_i \to v_j \in E$, we assign to e the set of colors corresponding to all obstacles S such that v_j lies in the interior of S and v_i does not lie in the interior of S. From our general position assumptions, it follows that $|S|$ is either 1 or 0. Roughly speaking, we assign a color when the edge *enters* into the corresponding obstacle.

Note that the way G is defined, it is a plane graph and we consider its natural embedding which is also planar. Since we assign colors when an edge of G enters an obstacle, it is easy to see that a k-color path π in G corresponds to a k-crossing path π' in the plane. For the other direction, without loss of generality we can assume that there exists a k-crossing path π' that visits a cell in the arrangement at most once. This holds because if not, we can always find a shortcut between two consecutive visits to the same cell. Therefore given such a path π' we can easily construct a path π in G by simply concatenating the vertices corresponding to each arrangement cell intersected by π' in order. Thus, we have the following immediate observation.

Lemma 3. *Given a set $\mathcal{S}$ of obstacles in the plane, we can build an edge colored graph $G = (V, E, \mathcal{C})$ with two fixed vertices v_s, v_t such that:*

1. *if there is a k-color v_s-v_t path in G, then there is also a j-crossing s-t path in the plane for some $j \le k$, and*
2. *if there is a k-crossing s-t path in the plane, then there is also a j-color path from v_s to v_t in G for some $j \le k$.*

We refer to such a graph $G = (V, E, \mathcal{C})$ as a *valid* edge colored graph for the arrangement. From the above discussion and using Lemma 1 and Theorem 1, we have the following.

Lemma 4. *Suppose we are given a valid edge colored graph $G = (V, E, \mathcal{C})$ for an arrangement of the set $\mathcal{S}$ of input obstacles in the plane. Moreover, given integer k, suppose we can construct the neighborhoods $\mathcal{N}(S)$ for all obstacles $S \in \mathcal{S}$ in polynomial time such* *that $\mathcal{N}$ has a total size of $O(k\beta^2)$ and also satisfies the bounded-occurrence property, where β is independent of k. Then, there exists a polynomial time algorithm that achieves an $O(\beta)$-approximation.*

In the following, given a set $\mathcal{S}$ of obstacles in the plane, we will show how to build a sparse neighborhoods $\mathcal{N}(S)$ for all $S \in \mathcal{S}$, such that the term β^2 in total size of the neighborhoods is close to linear in n.

3.1. Building a sparse neighborhood

Consider an edge $e = v_i \to v_j$ of G. We say that e *enters* the obstacle S if e intersects the boundary of S and the cell C_j corresponding to the destination vertex v_j of e is contained in S. This suggests a natural way of defining the neighborhood $\mathcal{N}(S)$: we simply *include all edges e that* enter *the obstacle S in the neighborhood $\mathcal{N}(S)$ of S*. It is easy to see that every occurrence of the color corresponding to an obstacle S can be uniquely charged to an edge in $\mathcal{N}(S)$. Therefore $\mathcal{N}$ satisfies the *bounded-occurrence* property.

Note that *every edge of the arrangement* is part of the boundary of a cell and due to general position assumption, belongs to the boundary of exactly one obstacle. Therefore, *every edge of our graph G* intersects the boundary of exactly one obstacle, and thus we have the following observation.

Observation 1. *Every edge of G is included in the neighborhood of at most one obstacle.*

For the *bounded-size* property, by Observation 1, we have that the total neighborhood size is $|E|$. However, $|E|$ can be $\Omega(n^2)$, so this directly is not useful for us. Therefore, the primary challenge is to sparsify this neighborhood and in the rest of the discussion, we work towards this goal.

One approach is to only consider the "shallow" edges $e = v_i \to v_j$ such that the destination cells C_j have depth at most k (that is, C_j is contained in at most k obstacles). That is, we define the neighborhood $\mathcal{N}(S)$ of obstacle S to be the set of all shallow edges that enter S. The *bounded-occurrence* property still holds because a k-crossing path cannot enter cells of depth more than k. By planarity, the number of shallow edges is within a multiplicative constant of the number of cells with depth at most k. Indeed, if obstacles are simpler shapes such as disks or pseudodisks, the number of cells with depth at most k is known to be $O(kn)$ [17]. Thus, the total neighborhood size is $O(kn)$. Hence, by Lemma 4, with $\beta = \sqrt{n}$, we readily obtain an $O(\sqrt{n})$-approximation for MCR when the obstacles are pseudodisks.

Theorem 3. *There exists an $O(\sqrt{n})$-approximation for MCR when the obstacles are pseudodisks.*

However, if the obstacles are more general such as rectangles, similar bounds on the number of cells with depth at most k do not hold. Nevertheless, we will use a different notion, called the *level* of a cell, to obtain a more refined neighborhood for the obstacles.

3.2. Approximation for general objects

In this section, we obtain similar approximation bounds when obstacles are polygonal. We begin by formally defining *level* of a cell.

Definition 4. The *level of a cell C* in an arrangement, denoted by $L(C)$, is the minimum number of objects one needs to cross to reach C from the source s by a path in the plane. For a collection of cells, we say they are at level at most k if the level of each cell in this collection is at most k.

With the above definition in place, we can use it to construct the neighborhood $\mathcal{N}$. Recall that the neighborhood $\mathcal{N}(S)$ of an object S (in our earlier definition) consists of all edges $e = v_i \to v_j$ of the graph G that enter the obstacle S into a "shallow" cell C_j. We can now simply use the level of a cell to define its shallowness. Specifically, we *include the edge e to the neighborhood $\mathcal{N}(S)$ if e enters S and the level of its source cell C_i and destination cell C_j are both at most k*. Since a k-crossing path would never enter a cell that has level more than k, bounded occurrence property is easily satisfied. Next, we need to bound the total size of this neighborhood.

Towards that end, suppose we define the level of a vertex of the arrangement $\mathcal{A}(\mathcal{S})$ in the same way. That is, the level of a vertex of the arrangement $\mathcal{A}(\mathcal{S})$ is the minimum number of obstacles crossed by any path from s to the vertex. We define $V_l(n)$ to be the maximum number of level l vertices in an arrangement of polygons with *input complexity n*. Therefore, $V_0(n)$ upper bounds the number of vertices in $\mathcal{A}(\mathcal{S})$ that can be reached from s without crossing any obstacle. It is important to note that n here is the input complexity (total number of polygon vertices) and not the number of edges of the arrangement, which could be quadratic in n. We now need a technical definition.

Definition 5. A function $f : \mathbb{Z}^+ \to \mathbb{R}^+$ is *well-behaved* if for any $n \geq 1$ and any random variable X with support $\{0, 1, 2, \ldots, n\}$ and expectation n/m, where $m \geq 1$, we have

$E[f(X)] \leq f(n)/m.$

We will require that $V_0(n)$ is upper bounded by some well-behaved function. Examples of well-behaved functions are n, $n \log n$, and n^2. We now upper bound the number of cells with level at most k, using standard machinery [17].

Lemma 5. *Suppose for a class T of objects, $V_0(n) \leq f(n)$ for some well-behaved function f. Then for any $k \geq 1$, the number of cells with level at most k in an arrangement of the objects from T with input complexity n is $O(k \cdot f(n))$.*

Proof. Let U_k be the set of vertices in the arrangement $\mathcal{A}(\mathcal{S})$ that are at level at most k. We will prove that $|U_k| = O(k \cdot f(n))$. Observe that for every cell that has level at most k, all vertices on its boundary must also be at level at most k. This holds because every point inside the cell is contained in the same set of obstacles, so if we can reach the cell by crossing at most k obstacles, we can also reach the boundary vertices by crossing no more than k obstacles. Therefore by planarity, the number of cells with level at most k will also be bounded by $O(k \cdot f(n))$.

We will now bound the number of vertices in the set U_k. Our approach is based on the well-known probabilistic method due to Clarkson and Shor [17]. Specifically, we sample each object in $\mathcal{S}$ independently with a probability $p = 1/m$, where $m = 2k$. Let $\mathcal{S}' \subseteq \mathcal{S}$ be the set of sampled objects. Fix one of the vertices $v \in U_k$, and assume its level is $l \leq k$; fix a path from s to v that crosses l objects. There are two cases: either v is an intersection point of two obstacle polygons S_i, S_j or v is a corner of some obstacle S_i. In the first case, v will show up as a level 0 vertex in $\mathcal{A}(\mathcal{S}')$ if $S_i, S_j \in \mathcal{S}'$ and none of the l obstacles from $\mathcal{S}$ that 'block' the path from s to v are sampled in $\mathcal{S}'$. This happens with a probability $p^2(1-p)^l \geq p^2(1-p)^k$. In the second case, v is corner of some polygon S_i. Similar to the earlier argument, v shows up as a level 0 vertex in $\mathcal{A}(\mathcal{S}')$ only if $S_i \in \mathcal{S}'$ and none of the l blocking obstacles in $\mathcal{S}$ are chosen to be in $\mathcal{S}'$. This happens with a probability at least $p(1-p)^k$. By linearity of expectation, the expected number of vertices of U_k that show up as a level 0 vertex in $\mathcal{A}(\mathcal{S}')$ is at least $|U_k| p^2 (1-p)^k = |U_k| \left(\frac{1}{m}\right)^2 \left(1 - \frac{1}{m}\right)^k \geq \alpha |U_k|/k^2$, for some constant $\alpha > 0$.

Now we will upper bound the expected number of level 0 vertices in $\mathcal{A}(\mathcal{S}')$. Since each polygon is sampled with an independent probability of $1/m$, if n was the number of vertices of polygons in $\mathcal{S}$, the expected number of vertices of polygons in $\mathcal{S}'$ is n/m. As f is well-behaved, the expected number of level 0 vertices in $\mathcal{A}(\mathcal{S}')$ is bounded above by $f(n)/m = f(n)/2k$.

It follows that $\alpha |U_k|/k^2 \leq f(n)/2k$. This gives $|U_k| = O(k \cdot f(n))$ and therefore we achieve the claimed bound. □

Combining Lemma 5 with Lemma 4, we get the following approximation for MCR.

Theorem 4. *Suppose for a class of objects, the maximum number $V_0(n)$ of level 0 vertices is upper bounded by $f(n)$ for some well-behaved function f. Then, there exists an $O(\sqrt{f(n)})$-approximation for MCR with this class of objects.*

From the work due to Edelsbrunner et al. [18] it follows that, for arbitrary n segments, $V_0(n) = O(n\alpha(n))$, where $\alpha(n)$ is the functional inverse of Ackermann's function. Moreover, if the segments are axis-parallel, it follows that $V_0(n) = O(n)$ [18, 19]. It is easy to see that both upper bounds are in terms of well-behaved functions. For polygonal objects with input complexity n, the number of underlying segments is also $O(n)$, so we obtain the same bounds. In particular, for general polygons $V_0(n) = O(n\alpha(n))$ and for rectilinear polygons, we have $V_0(n) = O(n)$. Hence, we have the following corollary.

Corollary 5. *There exists an $O(\sqrt{n\alpha(n)})$-approximation for MCR with polygonal obstacles. This improves to an $O(\sqrt{n})$-approximation when the polygons are rectilinear.*

4. Hardness of approximation

In this section, we describe the 2-inapproximability and the APX-hardness results for rectilinear polygons and axis-parallel rectangles, respectively.

4.1. 2*-inapproximability for rectilinear polygons*

We reduce Vertex Cover to MCR with rectilinear polygons. Recall that as input to the Vertex Cover problem we are given a graph $G = (V, E)$ with n vertices, and the goal is to find a minimum size subset $V' \subseteq V$ such that for any $(u, v) \in E$, either u or v is in V'. Let $e_1, \ldots, e_m$ be the edges of G. Now we describe the reduction. The constructed instance of MCR contains a region called *barrier* formed by a subset of the obstacles. Each point in the barrier is contained in at least $2n$ obstacles and thus if an s-t path intersects the barrier, it intersects at least $2n$ obstacles. We would ensure that any optimal path of the instance intersects at most n obstacles and thus no such path intersects the barrier region. Intuitively, the barrier region forces any optimal path to lie in a certain region, which we refer to as *corridor*.

The construction is the following. We place an obstacle corresponding to each vertex. For each edge (u, v) there is two possible pathlets (or subpaths of an s-t path) - one that intersects the obstacle corresponding to u and the other that

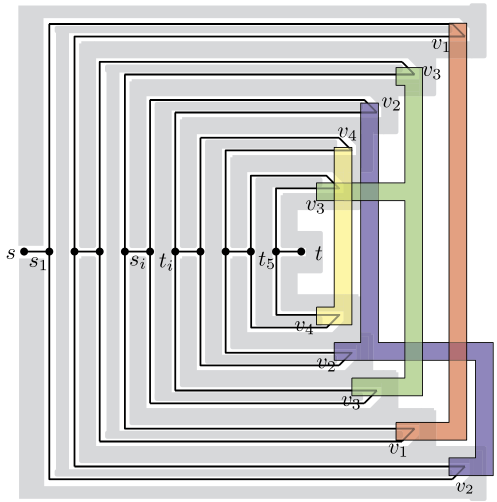

Fig. 1. An example of the construction. The barrier region is shown in gray.

intersects the obstacle corresponding to v. The start points of the two pathlets are same. The end points of the two pathlets are also same. Moreover, the m pairs of pathlets corresponding to the edges are placed one after the other in a series. This ensures that the endpoints of a pair are same as the start points of the next pair. Note that the selection of the pathlet corresponding to u (resp. v) for making an s-t path is equivalent to the selection of u (resp. v) for covering the edge (u, v). Thus, the s-t path formed by the chosen pathlets intersects only k obstacles if and only if all the edges can be covered by k vertices.

Next, we give more details about the layout of the pathlets and the obstacles. One of a pair of pathlets corresponding to each edge lies above x-axis and the other lies below x-axis. To ensure this, all the start and the end points of the pathlets are placed on the x-axis. Let s_i and t_i be the respective start and end points of the pathlets corresponding to the edge e_i. These points are placed on x-axis in the order $s_1, t_1, s_2, t_2, \ldots, s_m, t_m$. For each $1 \le i < m$, we connect the point t_i with s_{i+1} using a segment that joins the ith and $i+1$th pathlets. The point s is placed on x-axis before s_1 and t is placed on x-axis after t_m. s and s_1 are connected by a segment. Similarly, t_m and t are connected by a segment. Now to ensure that the pathlets cross the correct obstacles they are laid out in a fashion as shown in Fig. 1. Each pathlet contains exactly one point (tip) having the maximum x-coordinate. Moreover, all such tips corresponding to the pathlets are in convex position. Thus one can connect the tips of any subset of pathlets using segments to form a rectilinear polygon that does not intersect any other pathlets (see Fig. 1). Recall that each pathlet of an edge corresponds to a vertex incident on that edge. For each vertex $u \in V$, we connect the tips of the pathlets corresponding to u to form an obstacle whose shape is a rectilinear polygon. Note that the total number of possible s-t paths we constructed is 2^m. Now to make sure that any optimal s-t path is one of these 2^m paths we place the barrier around these paths. In other words, the barrier region forms a corridor for the paths. Any optimal path always stays inside the corridor, as it is expensive to cross the “wall” of the barrier. As the pathlets consist of a polynomial number of segments in total, a polynomial number of rectilinear polygons is sufficient to place avoiding the 2^m s-t paths. We make $2n$ copies of each such polygon to ensure the density. Lastly, each obstacle corresponding to a vertex is expanded sufficiently to ensure that it blocks the respective portion of the corridor. Note that the barrier can be placed in a way so that the corridor is arbitrarily thin, and thus this expansion can be done such that the obstacle do not cross any additional pathlets. From the above discussion, we obtain the following lemma.

Lemma 6. *For any $1 \le k \le n$, there is a size k vertex cover for G iff there is an s-t path that intersects k obstacles.*

As Vertex Cover is hard to approximate within a factor of $2 - \epsilon$ for any constant $\epsilon > 0$, assuming the Unique Games conjecture [20], we get the following theorem.

Theorem 6. *Minimum constraint removal with rectilinear polygons is hard to approximate within a factor of $2 - \epsilon$ for any constant $\epsilon > 0$, assuming the Unique Games conjecture.*

It is easy to see that the same idea can easily be extended for convex polygons. Basically, one can connect the tips of any subset of pathlets using segments to form a convex polygon that does not intersect any other pathlets.

Lemma 7. *Minimum constraint removal with convex polygons is hard to approximate within a factor of $2 - \epsilon$ for any constant $\epsilon > 0$, assuming the Unique Games conjecture.*

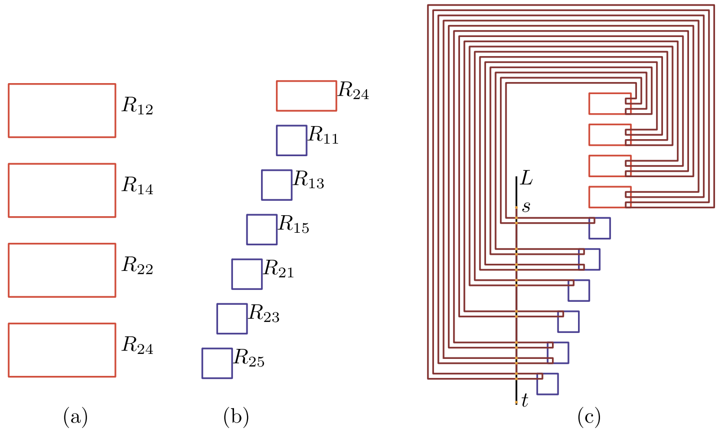

Fig. 2. (a) The stack of the class 1 rectangles for $m = 2$. (b) The initial configuration of the class 2 rectangles (shown by squares) for $m = 2$. (c) Drawing of the pathlets for the class 1 edges.

4.2. APX-hardness for axis parallel rectangles

We reduce a restricted version of vertex cover, which is referred to as SPECIAL-3VC, to our problem. Chan et al. [21] introduced this version for the sake of proving APX-hardness of several geometric optimization problems.

Definition 6. In a SPECIAL-3VC instance, we are given a graph $G = (V, E)$, where V contains $5m$ vertices $\{v_{ij} \mid 1 \le i \le m, 1 \le j \le 5\}$. E contains $4m + n$ edges - $4m$ of type 1 and n of type 2, where $2n = 3m$. Type 1 edges are of the form $\{(v_{ij}, v_{i,j+1}) \mid 1 \le i \le m, 1 \le j \le 4\}$. Type 2 edges are of the form $\{(v_{pq}, v_{xy}) \mid 1 \le p < x \le m,$ and q, y are odd numbers$\}$ such that any vertex v_{ij} with odd index j appears in exactly one such edge.

As each vertex v_{ij} with odd index j contributes exactly once in the type 2 edges, the number of type 2 edges is $3m/2 = n$. Chan et al. [21] proved that SPECIAL-3VC is APX-hard. Now we describe our reduction. The reduction is similar to the reduction for rectilinear polygons. We will have one obstacle corresponding to each vertex. Moreover, we construct two pathlets corresponding to each edge (u, v) such that one pathlet intersects the obstacle corresponding to u and the other intersects the obstacle corresponding to v. However, due to the simpler structure of the obstacles, here it is more complicated to ensure that the pathlets intersect the correct obstacles. The construction of the instance of MCR is as follows.

We denote the rectangles corresponding to v_{ij} by R_{ij}. First we place the rectangles corresponding to the vertices in $\{v_{ij} \mid 1 \le i \le m$ and j is even$\}$ in a way so that they form a stack like structure (see Fig. 2(a)). Also the rectangles are placed from top to bottom in the lexicographic order of the indexes (i, j): R_{ab} is considered before R_{cd} if $a < c$, and R_{a2} is considered before R_{a4}. We refer to these rectangles as the class 1 rectangles. Thereafter we place the rectangles corresponding to the remaining vertices. All these rectangles are placed in lexicographic order of the indexes (i, j). The first one is placed below R_{m4} (the last rectangle of the stack) in a way so that its left side is aligned with the left side of R_{m4}. Thereafter every rectangle is placed below the already placed ones and a little aligned towards the left w.r.t. the previous one (see Fig. 2(b)). We refer to these rectangles as the class 2 rectangles. We note that initially every class 2 rectangle is a square. Later each such rectangle might be expanded suitably towards right and below to ensure the correctness of the intersections with the pathlets.

Now let L be a vertical line such that all the rectangles are placed strictly to the right of it. All the endpoints of the pathlets we draw lie on L. Each pathlet is a curve consisting of rectilinear segments. The start (resp. end) points of the two pathlets corresponding to an edge are the same. We place s right above the topmost start point of the pathlets and connect s with this point by a vertical segment. Similarly, the point t is placed below the bottommost end point and joined with it by a vertical segment. At first we draw the pathlets for type 1 edges $\{(v_{ij}, v_{i,j+1}) \mid 1 \le i \le m, 1 \le j \le 4\}$ in the dictionary order of the indexes $(i, j, i, j+1)$, i.e. at first (v_{11}, v_{12}), then (v_{12}, v_{13}) and so on. The pairs of start and end points of the pathlets corresponding to these edges appear in the same order on L from top to bottom. For each type 1 edge $(v_{ij}, v_{i,j+1})$, let $s(i, j, j+1)$ and $t(i, j, j+1)$ be the respective start and end points of the pathlets. Note that for a v_{ij} with odd j, v_{ij} appears in two type 1 edges only if j is 3. Otherwise, it appears only once. Let P_{ij} be the horizontal projection (an interval) of R_{ij} on L. Then the start and endpoints of the pathlets of the type 1 edges with a vertex v_{ij} lie on P_{ij}. Now consider a type 1 edge $(v_{ij}, v_{i,j+1})$. Then either j or $j+1$ is odd. WLOG let j is odd. We draw the two points $s(i, j, j+1)$ and $t(i, j, j+1)$ on P_{ij} such that $s(i, j, j+1)$ lies above $t(i, j, j+1)$. One pathlet of $(v_{ij}, v_{i,j+1})$ lies on the right of L. It consists of three orthogonal segments and the only rectangle it intersects is R_{ij} (see Fig. 2(c)). The other pathlet is also drawn in a way so that the only rectangle it intersects is $R_{i,j+1}$ (see Fig. 2(c)). We repeat the process for all type 1 edges and each consecutive pairs of end and start points are joined with a vertical segment.

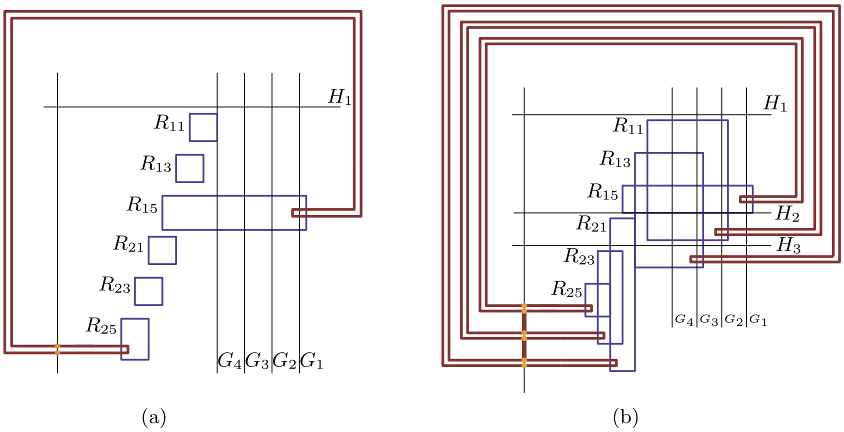

Fig. 3. (a) Drawing of the pathlets for the edge (v_{pq}, v_{m5}) where $m = 2, p = 1, q = 5$. (b) Drawing of the pathlets for the type 2 edges $(v_{15}, v_{25}), (v_{11}, v_{23}), (v_{13}, v_{21})$ where $m = 2$.

Now we draw the pathlets corresponding to the type 2 edges $\{(v_{pq}, v_{xy}) \mid 1 \le p < x \le m,$ and q, y are odd numbers$\}$. Note that, there are n such edges in G. We process all these edges in the reverse lexicographic order of the indexes (x, y) of the vertices v_{xy}. Thus at first we consider the edge that contains v_{m5}, then the edge that contains v_{m3} (if not considered already), then the edge that contains v_{m1}, then the edge that contains $v_{m-1,5}$ (if not considered already), and so on. We take $n+1$ vertical lines $G_1, \ldots, G_{n+1}$ such that G_{n+1} intersects the right vertical side of R_{11}, G_n is on the right of G_{n+1}, G_{n-1} is on the right of G_n, and in general G_i is on the right of G_{i+1}. Also let G_i and G_{i+1} are unit distance apart for $1 \le i \le n$. In every iteration $1 \le i \le n$, we define a horizontal line H_i. Denote by Q_i the region that lies below H_i and inside the strip defined by G_i and G_{i+1}. The drawing procedure is the following. Consider the first edge (v_{pq}, v_{m5}) corresponding to the vertex v_{m5}. Let H_1 be a horizontal line such that all the class 1 rectangles lie above it and all the class 2 rectangles lie below it. At first we expand R_{m5} sufficiently towards below such that one can place a pathlet with the following properties - the only rectangle it intersects is R_{m5}, it consists of two horizontal segments and one vertical segment, and its start and end points lie on L. Note that the expansion of R_{m5} do not create any new intersections with the existing pathlets. Thereafter R_{pq} is expanded sufficiently towards below and right to ensure that it has non-empty intersection with Q_1. Then the other pathlet can be drawn in a way so that it intersects the portion of R_{pq} that is in Q_1, and as Q_1 is empty the pathlet does not intersect any other rectangle (see Fig. 3(a)). Now consider the ith type 2 edge (v_{pq}, v_{xy}) in this order. Let all the edges before it in the ordering are already taken care of. It is easy to see that one can expand R_{xy} towards below for drawing a pathlet with the desired properties. Now to make sure that the other pathlet intersects only R_{pq}, set H_i to be a horizontal line such that the region Q_i, as defined above, is empty of previously drawn pathlets and expanded rectangles. Then we can expand R_{pq} towards below and right so that it has non-empty intersection with Q_i. As Q_i is empty one can draw the other pathlet as well with the desired properties (see Fig. 3(b)).

Lastly, we draw the barrier region around the paths. As the pathlets are orthogonal and consisting of a polynomial number of segments in total, the barrier region can be simulated using a polynomial number of rectangles and thus the construction can be realized in polynomial time. From the construction, it is straightforward to see the following lemma.

Lemma 8. *For any $1 \le k \le |V|$, there is a size k vertex cover for G iff there is an s-t path that intersects k rectangles.*

As Special-3VC is APX-hard we obtain the following theorem.

Theorem 7. *Minimum constraint removal with rectangles is APX-hard.*

5. Conclusion

In this paper, we study the minimum constraint removal problem and achieve close to $O(\sqrt{n})$ approximation for both graph and geometric versions. These results apply to pseudodisks and any polygonal shape without any additional restrictions. A natural open question to consider is to see if these bounds can be improved for simpler shapes such as rectangles or squares.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] I. Dinur, D. Steurer, Analytical approach to parallel repetition, in: Symposium on Theory of Computing, STOC 2014, 2014, pp. 624–633.
[2] K. Hauser, The minimum constraint removal problem with three robotics applications, in: Tenth Workshop on the Algorithmic Foundations of Robotics, WAFR 2012, 2012, pp. 1–17.
[3] S. Yuan, S. Varma, J.P. Jue, Minimum-color path problems for reliability in mesh networks, in: 24th INFOCOM 2005, vol. 4, IEEE, 2005, pp. 2658–2669.
[4] R. Hassin, J. Monnot, D. Segev, Approximation algorithms and hardness results for labeled connectivity problems, J. Comb. Optim. 14 (4) (2007) 437–453.
[5] D.Y.C. Chan, D.G. Kirkpatrick, Multi-path algorithms for minimum-colour path problems with applications to approximating barrier resilience, Theor. Comput. Sci. 553 (2014) 74–90.
[6] S. Bereg, D.G. Kirkpatrick, Approximating barrier resilience in wireless sensor networks, in: 5th ALGOSENSORS 2009, 2009, pp. 29–40.
[7] M. Korman, M. Löffler, R.I. Silveira, D. Strash, On the complexity of barrier resilience for fat regions, in: 9th ALGOSENSORS 2013, 2013, pp. 201–216.
[8] L. Erickson, S. LaValle, A simple, but NP-hard, motion planning problem, in: Proceedings of AAAI, AAAI Press, 2013.
[9] K.R. Tseng, D.G. Kirkpatrick, On barrier resilience of sensor networks, in: 7th ALGOSENSORS 2011, 2011, pp. 130–144.
[10] H. Alt, S. Cabello, P. Giannopoulos, C. Knauer, On some connection problems in straight-line segment arrangements, in: 27th EuroCG, 2011, pp. 27–30.
[11] E. Eiben, I. Kanj, How to navigate through obstacles?, in: Proceedings of ICALP, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.
[12] E. Eiben, J. Gemmell, I. Kanj, A. Youngdahl, Improved results for minimum constraint removal, in: Proceedings of AAAI, AAAI Press, 2018.
[13] J. Hershberger, N. Kumar, S. Suri, Shortest paths in the plane with obstacle violations, in: 25th Annual European Symposium on Algorithms (ESA 2017), 2017, pp. 49:1–49:14.
[14] P. Agarwal, N. Kumar, S. Sintos, S. Suri, Computing shortest paths in the plane with removable obstacles, in: 16th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2018), 2018, pp. 5:1–5:15.
[15] M.R. Fellows, J. Guo, I. Kanj, The parameterized complexity of some minimum label problems, J. Comput. Syst. Sci. 76 (8) (2010) 727–740.
[16] H.J. Broersma, X. Li, G. Woeginger, S. Zhang, Paths and cycles in colored graphs, Australas. J. Comb. 31 (1) (2005) 299–311.
[17] K. Clarkson, P. Shor, Application of random sampling in computational geometry, II, Discrete Comput. Geom. 4 (1989) 387–421.
[18] H. Edelsbrunner, L.J. Guibas, J. Hershberger, J. Pach, R. Pollack, R. Seidel, M. Sharir, J. Snoeyink, On arrangement of Jordan arcs with three intersection per pair, Discrete Comput. Geom. 4 (1989) 523–539, https://doi.org/10.1007/BF02187745.
[19] K. Kedem, R. Livne, J. Pach, M. Sharir, On the union of Jordan regions and collision-free translational motion amidst polygonal obstacles, Discrete Comput. Geom. 1 (1986) 59–70, https://doi.org/10.1007/BF02187683.
[20] S. Khot, O. Regev, Vertex cover might be hard to approximate to within 2-ϵ, J. Comput. Syst. Sci. 74 (3) (2008) 335–349.
[21] T.M. Chan, E. Grant, Exact algorithms and APX-hardness results for geometric packing and covering problems, Comput. Geom. 47 (2) (2014) 112–124.