Efficient constant factor approximation algorithms for stabbing line segments with equal disks

Konstantin Kobylkin

Abstract

Fast constant factor approximation algorithms are devised for an NP- and W[1]-hard problem of intersecting a set of \( n \) straight line segments with the smallest cardinality set of disks of fixed radii \( r > 0 \), where the set of segments forms a straight line drawing \( G = (V,E) \) of a planar graph without edge crossings. Exploiting tough connection of the problem with the geometric Hitting Set problem, an \((50 + 52\sqrt{\frac{1}{13}} + \varepsilon)\)-approximate \( O(n^4 \log n) \)-time and \( O(n^2 \log n) \)-space algorithm is given based on the modified Agarwal-Pan algorithm. More accurate \((34 + 24\sqrt{2} + \varepsilon)\)- and \((34 + 44\sqrt{6} + \varepsilon)\)-approximate algorithms are also proposed for cases where \( G \) is any subgraph of either an outerplane graph or a Delaunay triangulation respectively, which work within the same time and space complexity bounds, where \( \varepsilon > 0 \) is an arbitrary small constant. Moreover, an \( O(n^2 \log n) \)-time and \( O(n^2) \)-space 18-approximation is designed for the case where \( G \) is any subgraph of a Gabriel graph. To the best of our knowledge, related work only tackles the case where \( E \) consists of axis-parallel segments, resulting in an \( O(n \log n) \)-time and \( O(n \log n) \)-space 8-approximation.

Keywords: approximation algorithm, geometric Hitting Set problem, epsilon net, geometric data structure, Delaunay triangulation, Gabriel graph, line segments

1. Introduction

Design of approximation algorithms is a hot topic for many NP-hard combinatorial optimization problems related to geometric coverage and intersection (see e.g. works [3], [8], [12], [13], [17]). Some problems from this class can be

\[\text{Preprint submitted to Computational Geometry: Theory and Applications} \quad \text{July 31, 2018}\]
formulated in the following general form. Suppose a family $\mathcal{D}$ is given of simply shaped sets from $\mathbb{R}^2$, which can be e.g. disks, straight line segments, triangles etc. The problem is to find the smallest cardinality set $Q$ of translates of a given set $Q_0 \subset \mathbb{R}^2$ such that $\bigcup_{Q \in Q} Q$ either covers or intersects each $D \in \mathcal{D}$ in some prescribed way. In this paper fast constant factor approximation algorithms are constructed for a special NP-hard ([12], [14]) geometric intersection problem from this class related to networking and sensor placement.

**Intersecting Plane Graph with Disks (IPGD):** given a straight line drawing (or a plane graph) $G = (V, E)$ of an arbitrary simple planar graph without edge crossings and a constant $r > 0$, find the smallest cardinality set $C \subset \mathbb{R}^2$ of points such that each edge $e \in E$ is within Euclidean distance $r$ from some point $c = c(e) \in C$. Here each isolated vertex $v \in V$ is treated as a zero-length segment $e_v \in E$.

The IPGD abbreviation is used throughout our paper to denote the problem for simplicity of presentation.

To the best of our knowledge special settings of the IPGD problem are first considered in [12] for cases of (possibly overlapping) axis-parallel and bounded length straight line segments. From one hand, the IPGD problem can be expressed in the form of special geometric intersection problem which is to find the least cardinality set of radius $r$ disks whose union has nonempty intersection with each segment from $E$. From the other hand, it coincides with the well known geometric **Hitting Set** problem on the plane.

The classical geometric **Hitting Set** problem on the plane is to find the smallest cardinality subset $H$ of some given set $Y \subseteq \mathbb{R}^2$ such that $H \cap R \neq \emptyset$ for any $R \in \mathcal{R}$ where $\mathcal{R}$ is some given family of subsets of $\mathbb{R}^2$ which are called objects. Moreover, a pair $(Y, \mathcal{R})$ is associated with each instance of the **Hitting Set** problem called a range space. Obviously, the IPGD problem is equivalent to the **Hitting Set** problem for a specific range space $(\mathbb{R}^2, \mathcal{N}_r(E))$, where $\mathcal{N}_r(E) = \{ N_r(e) : e \in E \}$, $N_r(e) = \{ x \in \mathbb{R}^2 : d(x, e) \leq r \}$ and $d(x, e)$ is Euclidean distance between a point $x \in \mathbb{R}^2$ and a segment $e \in E$.

The IPGD problem could be of interest in network security, sensor network deployment and facility location. For example, in [12] its sensor deployment applications are reported for road networks. Namely, for a road network possible intrusion activity must be monitored by deploying sensor devices near its roads which have uniform circular sensing area. As full network surveillance might be costly, it may be sufficient to guarantee that the intruder movement is detected at least once somewhere along any its road. Minimizing total cost of the deployed identical sensors can be modeled in the form of the IPGD problem assuming that network roads are modeled by interior disjoint straight line segments. Moreover, another network security application may be of interest of the IPGD problem for optical fiber networks following the approach of [1].

---

1A subset $H \subset \mathbb{R}^2$ is named a hitting set for $\mathcal{R}$ if $H \cap R \neq \emptyset$ for every $R \in \mathcal{R}$.

2For $x \in \mathbb{R}^2$ $N_r(x)$ denotes a radius $r$ disk centered at $x$. 

---
The IPGD problem generalizes a classical NP-hard disk covering problem
on the case of non-zero length segments. In the disk covering problem one needs
to cover a given finite point set on the plane with the least cardinality set of
equal disks. Generally, the IPGD problem is quite different from the Hitting
Set problem for families of equal disks (which is equivalent to the disk covering
problem) and other bounded aspect ratio objects as segment lengths are not
assumed to be bounded from above by any linear function of $r$.

This impedes design of approximation algorithms for the IPGD problem
both via known grid shifting technique and through seeking maximal subsets
of non-overlapping objects within $N_r(E)$, recalling its equivalence to the Hitting
Set problem: these are the traditional techniques used to design modest
complexity PTAS and small constant factor approximations with reasonable
complexity both for disk covering and Hitting Set problems with bounded
aspect ratio objects.

In spite of all these difficulties a local search based PTAS exists [17] for the
IPGD problem (see also more general result in [19]). It has huge time complex-
ity though, being considered in the form of constant factor approximation for
some sufficiently small $\varepsilon$ (see e.g. introduction in [3]). Moreover, existence of an
$O(1)$-approximation is also guaranteed by general result from [20] of reasonable
time complexity but with extremely large absolute constant factor. In this pa-
per one is managed to design approximation algorithms for the IPGD problem
of reasonable time complexity whose factors are modest absolute
constants.

However, when segments of $E$ are allowed to intersect by their relative interi-
ors and admitted to have arbitrary large number of orientations, it is unlikely [3]
to achieve constant factor approximation at least by using known approaches.
In the case of a few orientations a reasonable constant factor approximation still
exists [12] which can be computed in modest time.

1.1. Our results and related work

The IPGD problem represents most general setting of geometric intersec-
tion problem with straight line segments and translates of a given disk in which
segments can have arbitrary orientations and lengths. An $O(n \log n)$-time $8$-
approximation algorithm is only known [12] in the restricted case, where seg-
ments are axis-parallel but are allowed to overlap. This algorithm is based on
specific properties of Euclidean $r$-neighbourhoods of segments having the same
orientation. Namely, if $E$ consists of, say, horizontal segments, only 8 points
are needed to hit the set $\{N \in N_r(E) : N \cap I \neq \emptyset\}$ for a fixed $I \in N_r(E)$.
Essentially the same idea is used to design an $O(n \log n)$-time 4-approximation
[6] for the situation where $E$ consists of points (i.e. zero length segments). This
is not the case for general setting of the IPGD problem in which segments are
allowed to have arbitrarily large number of orientations and arbitrarily large
length with respect to $r$.

Due to close links of the IPGD problem with the geometric Hitting Set
problem, recent results are also reviewed below for close settings of the latter
problem. Recently a remarkable breakthrough has been achieved in construct-
ing sharper and faster constant factor approximation algorithms for geometric
Hitting Set problems due to application of ideas of local search [9] as well as of statistical learning using epsilon nets [20] and some complexity measures [18]. In particular, local search approach has become quite competitive for designing low constant factor approximations for classes of geometric Hitting Set problems with families of disks [9] and pseudo-disks [4]. It can be proved that objects from $N_r(E)$ form a family of pseudo-disks without loss of generality.

Indeed, as straight line segments from $E$ intersect at most at their endpoints, segments of $E$ can be slightly shifted to become pairwise disjoint and non-parallel while keeping all nonempty intersections of subsets of objects from $N_r(E)$ with some slightly larger $r$. For two non-overlapping segments $e$ and $e'$ it can be understood that $|\text{bd } N_r(e) \cap \text{bd } N_r(e')| \leq 2$ because Euclidean distance grows strictly monotonically from $e$ (or from $e'$) to point of the curve $\chi(e,e') = \{ x \in \mathbb{R}^2 : d(x,e) = d(x,e') \}$ as it moves along $\chi(e,e')$ in any of two opposite directions starting from midpoint of the segment which joins closest points of $e$ and $e'$, where a set $\text{bd } N$ denotes the boundary of $N$ for $N \subset \mathbb{R}^2$. Therefore an algorithm from [4] yields 4-approximate solution for the IPGD problem in $O(n^{15})$-time, being too complicated.

In this paper an alternative approach is adopted based on epsilon nets [2] which results in faster algorithms at the expense of larger constants in their approximation factors. Namely, a simple to implement $(50 + 52\sqrt{13} + \varepsilon)$-approximation is proposed for the IPGD problem working in

$$O\left(\left(n^2 + \frac{n \log n}{\varepsilon^2} + \frac{\log n}{\varepsilon^3}\right)n^2 \log n\right)$$

time along with $(34 + 24\sqrt{2} + \varepsilon)$- and $(34 + 44\sqrt{11} + \varepsilon)$-approximations for special segment configurations $E$ defined by outerplane graphs and Delaunay triangulations (arising in network applications) running within the same time, where $\varepsilon > 0$ is an arbitrary small constant. Moreover, an $O(n^2 \log n)$-time 18-approximation is designed in the case where $E$ is subset of edge set of a Gabriel graph, which comes up in applications for wireless networks.

Within epsilon net approach the only relevant constant factor approximation is known for the Hitting Set problem for sets of pseudo-disks [20] with similar time complexity bounds which gives a huge constant approximation factor. Moreover, if $E$ consists of zero length segments, a $(13 + \varepsilon)$-approximation can be designed based on weak epsilon nets [3], which works in $O(n^5 \log^3 n)$ time [10]. Thus, our algorithms give a competitive tradeoff of approximation factor and time complexity being compared with known local search and epsilon net based approximation algorithms designed for similar geometric Hitting Set problems.

3A set of geometric objects on the plane is called a set of pseudo-disks if any two its objects have their boundaries intersecting at most twice.

4In this work the IPGD problem is also reduced to the Hitting Set problem with $|Y| = O(n^2)$. 

4
1.2. Brief description of main ideas and algorithms

As was mentioned in the introduction, the IPGD problem is equivalent for a set \( E \) of straight line segments to the \textit{Hitting Set} problem on a set \( N_r(E) \) of Euclidean \( r \)-neighbourhoods of segments of \( E \). To obtain constant factor approximations for the IPGD problem approaches from \cite{2} and \cite{20} are exploited in this paper which are designed for the general geometric \textit{Hitting Set} problem.

To give a short outline of our algorithms for the IPGD problem, we begin with the following idea which provides low constant factor approximations for cases of the IPGD problem with zero-length \cite{6} and axis-parallel \cite{12} segments. Say, for the case of zero-length segments the idea is to use a “divide-and-conquer” heuristic by finding a maximal (with respect to inclusion) independent set \( I \) of disks within \( N_r(E) \), i.e. a maximal subset of pairwise non-overlapping disks. As \( 7r \) radius disks are sufficient to cover \( 2r \) radius disk, a \( 7 \)-point set \( S \) can be constructed which has nonempty intersection with disks from \( N_{I} = \{ N \in N_r(E) : N \cap I \neq \emptyset \} \) for any \( I \in \mathcal{I} \). Therefore a set \( \bigcup_{I \in \mathcal{I}} S_I \) gives a \( 7 \)-approximate problem solution.

Instead, in this paper a similar but more relaxed and indirect approach is adopted which employs ideas of work \cite{20}. Indeed, direct application fails to work of the aforementioned heuristic in the general setting of the IPGD problem as one can not guarantee hitting sets for \( N_I \) to exist with constant number of points uniformly for all \( I \in \mathcal{I} \). Actually, it is not possible because, first, lengths are not assumed to be bounded from above of segments from \( E \) by any linear function of \( r \). Second, one can not cluster segments into constant number of groups with similar segment orientations and apply the heuristic in each group separately as done in \cite{12}: number can be arbitrarily large of distinct orientations of segments from \( E \).

Following approach of \cite{2}, one can get an approximate solution to the IPGD problem by a compound algorithm which combines two different subalgorithms, first of which exercises a similar “divide-and-conquer” approach for some special subsets of \( N_r(E) \).

1.2.1. First subalgorithm

First subalgorithm represents an improved version of the algorithm from \cite{20}. Given its parameters \( 0 < \varepsilon < 1 \) and a positive weight map \( w : Y_0 \rightarrow \mathbb{R}_+ \), defined on some finite subset \( Y_0 \subset \mathbb{R}^2 \) with \( \text{OPT} = \text{OPT}(\mathbb{R}^2, N_r(E)) = \text{OPT}(Y_0, \mathcal{N}_r(E)) \) and \( |Y_0| = O(n^2) \), it seeks a hitting set for a subset of those objects from \( N_r(E) \) whose weight is at least \( \varepsilon \)th fraction of \( w(Y_0) \), where \( w(N) = \sum_{y \in N \cap Y_0} w(y) \) for \( N \subset \mathbb{R}^2 \) and \( \text{OPT}(Y, \mathcal{R}) \) is the optimum of the \textit{Hitting Set} problem for a given range space \( (Y, \mathcal{R}) \). Hitting set has a special name \cite{13} of the kind, which is output by the first subalgorithm.

\footnote{Set \( Y_0 \) can be constructed by computing vertices of arrangement of boundaries of objects from \( \mathcal{N}_r(E) \).}
Definition 1. Let $0 < \varepsilon < 1$ and $w : Y_0 \to \mathbb{R}_+$. A subset $Y' \subseteq Y_0$ is called a (weighted) $\varepsilon$-net for a range space $(Y_0, \mathcal{N}_\varepsilon(E), w)$ if $Y' \cap N \neq \emptyset$ for any $N \in \mathcal{N}_\varepsilon(E)$ with $w(N) > \varepsilon w(Y_0)$. If $Y'$ is allowed to contain arbitrary points of the plane, it is called a weak $\varepsilon$-net.

More specifically, first subalgorithm seeks a maximal subset $I$ of nearly non-overlapping objects within the subset $\mathcal{N}_\varepsilon = \{N \in \mathcal{N}_\varepsilon(E) : w(N) > \varepsilon w(Y_0)\}$. Here one controls amount of overlap of objects from $I$ through a special parameter $\delta$ and a weight map $w$ on $Y_0$ as follows [20].

Definition 2. Given a subset $P \subseteq \mathcal{N}_\varepsilon(E)$, a parameter $\delta > 0$ and a weight map $w : Y_0 \to \mathbb{R}_+$, a subset $I = I(\delta) \subseteq P$ is called a maximal (with respect to inclusion) $\delta$-independent for a range space $(Y_0, P, w)$ if

$$w(I \cap I') \leq \delta w(Y_0)$$

for any distinct $I, I' \in I$ and for any $N \in P$ there is some $I = I(N) \in I$ with $w(N \cap I) > \delta w(Y_0)$.

Given a parameter $0 < \varepsilon < 1$ and a weight map $w : Y_0 \to \mathbb{R}_+$, first subalgorithm seeks a maximal $\delta$-independent set $I$ for $(Y_0, \mathcal{N}_\varepsilon, w)$ with $\delta = \theta_0 \varepsilon$, where $0 < \theta_0 < 1$ is some IPGD problem specific absolute constant (see subsection 3.1). Then it finds a short length hitting set $H_1$ (see subsection 3.2) for $\mathcal{N}_{\delta, I} \subseteq \{N \in \mathcal{N}_\varepsilon : w(N \cap I) > \delta w(Y_0)\} \cup \{I\}$ for each $I \in I$ to come up with a hitting set $H(\delta) = \bigcup_{I \in I} H_1$ for $\mathcal{N}_\varepsilon$. It turns out that $|H_1| \leq \frac{c_1 w(I)}{\delta w(Y_0)} + c_2$, resulting in the bound

$$|H(\delta)| \leq \frac{c_1 \sum_{I \in \mathcal{N}_\varepsilon} w(I)}{\delta w(Y_0)} + c_2 |I|,$$

where $c_1, c_2 > 0$ are some absolute constants. Due to [20] (see proof of theorem 4 from that paper and lemmas 2 and 3 below), $|I| = O\left(\frac{1}{\varepsilon}\right)$ and $\frac{c_1 \sum_{I \in \mathcal{N}_\varepsilon} w(I)}{\delta w(Y_0)} = O\left(\frac{1}{\varepsilon}\right)$. Therefore one gets the ($w$-independent) bound $|H(\delta)| = O\left(\frac{1}{\varepsilon}\right)$.

First subalgorithm is presented in its most general form in subsection 2.2 for an arbitrary range space $(Y, \mathcal{R})$. It further develops machinery from [20], which simplifies design of epsilon net seeking algorithms by reducing it to constructing a special algorithm, finding hitting sets for geometrically simpler subspaces of $(Y, \mathcal{R})$. This subalgorithm may be of interest due to improvements in approximation factor it gives over the original algorithm from [20].

1.2.2. Second subalgorithm

Second subalgorithm is a modification of the algorithm from [2]. It adjusts the parameter $\varepsilon$ and computes a weight map $w$ to get $|H(\delta)| = O(\text{OPT})$. Namely, it computes point weights $w(y), y \in Y_0$, to get the inequality

$$w(N) > \varepsilon w(Y_0)$$

hold for all $N \in \mathcal{N}_\varepsilon(E)$, where $\varepsilon$ is also found such that $\varepsilon = \frac{1}{\lambda_0 \text{OPT}}$ for some $\lambda_0 \approx 1$. 

6
1.2.3. Main algorithm

Finally, the compound main algorithm works as follows: second subalgorithm runs to compute a suitable weight map $w_0$ (i.e., satisfying the inequality (1) for all $N \in \mathcal{N}_r(E)$) and gets a value $\varepsilon_0 = \frac{1}{\lambda_0 \text{OPT}}$ of the parameter $\varepsilon$ whereas first (epsilon net seeking) subalgorithm constructs a hitting set for $\mathcal{N}_r(E)$ being applied for found $\varepsilon_0$ and $w_0$, thus, arriving at the constant factor approximate solution. As the IPGD problem admits its feasible solutions to contain arbitrary points of $\mathbb{R}^2$ and $\text{OPT} = \text{OPT}(Y_0, \mathcal{N}_r(E))$, first subalgorithm is allowed to return weak epsilon nets.

1.2.4. Our geometric ideas and algorithms

Within described algorithmic framework there are two basic ingredients that need to be specified for the main algorithm to return approximate solutions for the IPGD problem. The first ingredient consists in a special procedure which seeks hitting sets for sets $\mathcal{N}_{\delta,I}$ of size at most $c_1 \delta w(I) + c_2$ with small constants $c_1$ and $c_2$ for an arbitrary object $I \in \mathcal{I}$ of maximal $\delta$-independent set $\mathcal{I}$ for $(Y_0, \mathcal{N}_r, w)$. In subsection 3.2 several fast variants are given of the procedure with small constants $c_1$ and $c_2$ for different segment configurations including general case of interior-disjoint segments and cases where segments form edge sets of subgraphs of special plane graphs. Proofs for their performances heavily exploit geometry of Euclidean $r$-neighbourhoods of segments.

The second ingredient consists in choosing a suitable value of the parameter $\theta_0$, governing the degree of intersection of objects from $\mathcal{I}$. In subsection 3.1 a special IPGD problem specific geometric approach is used to adjust $\theta_0$, which finally leads to small constants in upper bounds $|\mathcal{I}| = O \left( \frac{1}{\varepsilon} \right)$, $\frac{\sum_{l \in \mathcal{I}} w(l)}{\delta w(Y_0)} = O \left( \frac{1}{\varepsilon} \right)$ and $|H(\delta)| = O \left( \frac{1}{\varepsilon} \right)$. The approach is based on obtaining upper bounds of the form $\sum_{l \in \mathcal{I}} w(l) \leq c_3 w(Y_0)$ with small constant $c_3 \geq 1$, using geometric properties of intricate mutual location of objects from $\mathcal{I}$ and points from $Y_0$, which are partly $r$-independent: some of those properties depend only on mutual location of straight line segments from $E(\mathcal{I})$, where $E(\mathcal{N}) \subseteq E$ is the segment set with $\mathcal{N} = \mathcal{N}_r(E(\mathcal{N}))$ for $\mathcal{N} \subseteq \mathcal{N}_r(E)$.\footnote{Let also $e(\mathcal{N})$ be a segment such that $N = \mathcal{N}_r(e(\mathcal{N}))$ for $N \in \mathcal{N}_r(E)$.}

From one hand, this IPGD problem specific method may be of interest in establishing accuracy bounds for constant factor approximation algorithms for wide class of HITTING SET problems on sets of convex objects defined by a well-behaved metric, say, on sets of $l_1$- and $l_\infty$-induced $r$-neighbourhoods of straight line segments. From the other hand, this method represents an interesting application of the abstract approach adopted in [20].

2. Refined algorithmic framework: epsilon nets and independent sets

In this preliminary section known abstract Pyrga-Ray and Agarwal-Pan algorithms from [20] and [2] are briefly reviewed from which above two subalgo-
rithms are originated. Improved versions are also presented of these abstract algorithms in most general form for the Hitting Set problem on a range space \((Y, \mathcal{R})\), achieving the approximation factor in their combination more than 20 times smaller than the approximation factor for combination of the original algorithms. Being rather technical in the light of ideas from [2] and [20], these improvements reveal explicit constants though, which can be achieved in approximation factors of the algorithms based on those ideas. A similar improvement is achieved in [10] of the original Agarwal-Pan algorithm in order to design a low constant factor approximation for the geometric Hitting Set problem on a set of disks.

2.1. Parametric Agarwal-Pan algorithm for general Hitting Set problems

In this subsection a modified Agarwal-Pan algorithm is given for computing a suitable weight map \(w\) and parameter \(\varepsilon\) which generalizes the original Agarwal-Pan algorithm [2]. It introduces small parameters into this known algorithm adjusting which reduces the resulting approximation factor by at least factor of \(4e\) of its combination with any epsilon net seeking algorithm.

Briefly, the original Agarwal-Pan algorithm, being applied for the range space \((Y, \mathcal{R})\), iterates through an integer parameter \(k\), aiming to get \(\text{OPT}(Y, \mathcal{R}) \in (k/2, k]\) (or \(k \leq \text{OPT}(Y, \mathcal{R})\)) and a weight map \(w\) at once such that the inequality \(w(R) > \frac{w(Y)}{2e}\) holds true for all \(R \in \mathcal{R}\). Suppose that an epsilon net seeking algorithm is used for \((Y, \mathcal{R}, w)\), returning \(\varepsilon\)-nets of size at most \(C\varepsilon\), where \(C\) is some constant. Then a combination of these two algorithms is an algorithm whose approximation factor is the product of factors \(4e\) and \(C\). These two factors represent multiplicative errors introduced by the Agarwal-Pan and chosen epsilon net seeking algorithm.

2.1.1. Parametric Agarwal-Pan algorithm

To reduce the first factor to \(1 + \nu\) for an arbitrarily small \(\nu > 0\), positive parameters \(\delta, \eta, \lambda_1\) and \(\mu\) are introduced into the original Agarwal-Pan algorithm, resulting in a special Parametric Agarwal-Pan algorithm. Let \(\kappa = 2\eta - \lambda_1\).

In accordance with the original Agarwal-Pan algorithm, at the core of the Parametric Agarwal-Pan algorithm the following subalgorithm lies called the Iterative reweighting subalgorithm. Given a range space \((Y, \mathcal{R})\) and an integer \(k\) as its input, this subalgorithm updates a weight map \(w\) on \(Y\) (i.e. modifies \(w(y)\) for each \(y \in Y\)) aiming to arrive at the case where all sets from \(\mathcal{R}\) have their weights exceeding \(\frac{w(Y)}{\lambda_0k}\) for some small \(\lambda_0 > 0\), which depends on the introduced parameters.

Namely, the subalgorithm performs a series of passes, where a single pass processes sets from \(\mathcal{R}\) one by one which is called a round. When processing a single set \(R \in \mathcal{R}\) within a round, it is checked if \(w(R) > \frac{w(Y)}{\lambda_0k}\). If it is not, weight of each \(y \in Y \cap R\) is multiplied by \(1 + \lambda_1\). In each round at most \(\lceil \mu k \rceil\) weight updates are done for all sets from \(\mathcal{R}\) in total. Given \(k\), at most \(\lceil \frac{2\lambda_0 \ln(|Y|/k)}{\mu k} \rceil\) rounds are performed.

8
Following its original version, the Parametric Agarwal-Pan algorithm itself is simple and consists of calling the Iterative reweighting subalgorithm within a binary search loop for $k$ along with applying a special subalgorithm to seek $\varepsilon$-net for space $(Y, R, w)$ of size at most $\frac{C}{\varepsilon}$. Namely, as a preprocessing step, a $\frac{1}{\lambda k}$-net $H_1$ is constructed for $(Y, R)$ with $w \equiv 1$ for each tried $k$. Then a suitable weight map $w_{\delta}$ and an integer $k \leq \text{OPT}$ are computed using the Iterative reweighting subalgorithm, applied for $Y_{\delta} = Y \setminus H_1$ and $R_{\delta} = R \setminus \{R \in R : R \cap H_1 \neq \emptyset\}$ with uniform initial weight map on $Y_{\delta}$. After that a $\frac{1}{\lambda k \varepsilon^s(\mu)}$-net $H_2$ is found for $(Y_{\delta}, R_{\delta}, w_{\delta})$, where $s$ is the number of weight updates performed for sets from $R_{\delta}$ at the final round of the Iterative reweighting subalgorithm. Finally, a set $H := H_1 \cup H_2$ is returned as an output of the Parametric Agarwal-Pan algorithm.

Of course, one can get the original Agarwal-Pan algorithm from the Parametric Agarwal-Pan algorithm, setting $\delta = \lambda = \mu := 2$ and $\lambda_1 = 1$. Pseudocode is given in the appendix for the Parametric Agarwal-Pan algorithm and the Iterative reweighting subalgorithm.

### 2.1.2. Performance of the Parametric Agarwal-Pan algorithm

To summarize on the performance of the Parametric Agarwal-Pan algorithm some notations and definitions are needed which reflect basic operations done within it along with their complexity. Let $0 < \varepsilon < 1$ and $\varphi = \varphi(|Y|, |R|, \frac{1}{\varepsilon})$ be (non-decreasing with respect to its arguments) complexity of computing an $\varepsilon$-net of size at most $\frac{C}{\varepsilon}$ for space $(Y, R, w)$. Its space cost is assumed to be linear with respect to the cost of storing the range space $(Y, R, w)$. Moreover, the Iterative reweighting subalgorithm (namely, its step 4) requires the following data structures to be implemented.

**Definition 3.** A data structure for a range space $(Y, R, w)$ is called a range counting data structure if it can perform the following two operations:

1. return $w(R)$ for any $R \in R$;
2. update weight $w(y)$ of any $y \in Y$.

**Definition 4.** A data structure for a range space $(Y, R)$ is called a range reporting data structure if it can report all points of $Y \cap R$ for any $R \in R$.

Let $\tau_s = \tau_s(|Y|, |R|)$ be complexity of performing $s$th operation within range counting data structure, $s = 1, 2$ and $\omega = \omega(|Y|, |R|)$ be complexity such that points can be reported of $Y \cap R$ within time of the order $O(\omega + |Y \cap R|)$ for any $R \in R$. Moreover, it is assumed that preprocessing times spent by range counting and reporting structures (i.e. times required to create them) are of the order $O(\tau^{(0)})$ and $O(\omega^{(0)})$ respectively. Their space cost is again assumed to be linear with respect to cost of storing of the respective (possibly weighted) range space.

The lemma below summarizes on the performance of the Parametric Agarwal-Pan algorithm.

---
Lemma 1. For any small constants $\delta, \mu, \lambda_1, \eta > 0$ such that $\kappa > 0$, the Parametric Agarwal-Pan algorithm is $C(\delta + \lambda e^{\lambda_1 \mu/\lambda})$-approximate, works in
\[
O \left( \tau(0) + \omega(0) + \left( \frac{u_{\mu \delta}}{\lambda_1 \kappa} + \varphi_{\delta} \right) \log \text{OPT} (Y_\delta, R_\delta) + \varphi_{\lambda e^{\lambda_1 \mu/\lambda}} \right)
\]
time and
\[
O \left( \frac{|Y \setminus Y_\delta|}{\mu} + \frac{|Y_\delta| \log |Y_\delta|}{\mu} \left( \frac{1}{\kappa} + \frac{1}{\lambda_1} \right) + |R| \right)
\]
space, where $\lambda = 1 + \eta$,
\[
u_{\mu \delta} = \left[ \frac{|R_\delta| \tau_1}{\mu} + \frac{|Y| \tau_2}{\mu \delta} + \left( \text{OPT} (Y_\delta, R_\delta) + \frac{1}{\mu} \right) \omega \right] \log |Y|
\]
and $\varphi_\theta = \varphi(|Y|, |R|, \theta \text{OPT} (Y_\delta, R_\delta))$.

Proof of the lemma is left for Appendix B being rather technical in view of lemmas 2.1 and 3.1 from [2].

2.2. Improved independent sets for building small epsilon nets

From the lemma 1 of previous subsection, it is clear that approximation factor of the Parametric Agarwal-Pan algorithm depends on the constant $C$ appearing in the $O \left( \frac{1}{\epsilon} \right)$ bound on size of epsilon net returned by a chosen epsilon net seeking algorithm. Therefore, to achieve better approximation, the epsilon net seeking algorithm should be applied within the Parametric Agarwal-Pan algorithm, which returns $\epsilon$-nets of size at most $C$. In this subsection we come up with improvements of the Pyrga-Ray epsilon net seeking algorithm from [20] which result in smaller constant $C$ than that for the original approach.

The original approach of [20] splits the problem of seeking small sized epsilon net for $(Y, R, w)$ into several independent subproblems each of which is to find a short length hitting set for some special subset of
\[
R^s_\varepsilon = \{ R \in R : 2^{s+1} \varepsilon w(Y) \geq w(R) > 2^s \varepsilon w(Y) \}
\]
for $s = 0, \ldots, \log_2 \frac{1}{\epsilon} - 1$. Within this approach one extracts a maximal $\theta_0 2^s \varepsilon$-independent subset $I_s$ for $(Y, R^s_\varepsilon, w)$ for some problem specific constant $0 < \theta_0 < 1$. Then it splits $R^s_\varepsilon$ into subsets $R^s_{\varepsilon, I}$, $I \in I_s$, where
\[
R^s_{\varepsilon, I} \subseteq \{ R \in R^s_\varepsilon : w(R \cap I) > \theta_0 2^s \varepsilon w(Y) \} \cup \{ I \}.
\]
Indeed, computing a small hitting set for $R^s_{\varepsilon, I}$ is a much easier task than getting that set for $R_\varepsilon$, where $R_\varepsilon = \{ R \in R : w(R) > \varepsilon w(Y) \}$. Moreover, such tasks can be executed in parallel for distinct sets from $I_s$.

Finding hitting sets for each piece $R^s_{\varepsilon, I}$ of this fine splitting of $R_\varepsilon$ leads to introducing factor of 2 into upper bounds on size of the resulting epsilon net. We get over this problem by splitting $R_\varepsilon$ into larger blocks $R^s_{\varepsilon, I} \subseteq \{ R \in R_\varepsilon :$
$w(R \cap I) > \theta_0 \varepsilon w(Y) \cup \{I\}$ generated by some maximal $\theta_0 \varepsilon$-independent subset $\mathcal{I}$ for $(Y, R, w)$. This at least halves bounds on the length of returned epsilon net at the expense of dealing with more difficult subproblems of seeking small sized hitting sets for sets $\mathcal{R}_{\varepsilon, I}$ called sets of dependent objects in the sequel. In this paper it is shown that these subproblems can be solved quickly and accurately in the case of the IPGD problem (see the subsection 3.2).

Below an algorithm is presented improving on the algorithm from [20] by using this rougher splitting to perform epsilon net search. Its time complexity depends on the complexity of an auxiliary procedure and a data structure. Given its parameter $0 < \delta < 1$ and a subset $\mathcal{R}_I(\delta) \subseteq \{R \in \mathcal{R} : w(R \cap I) > \delta w(I)\}$ for some weight map $w : Y \to \mathbb{R}_+$ and an object $I \in \mathcal{R}$, auxiliary procedure (when exists) generates points of $Y \cap I$ (if necessary) and seeks a hitting set of size at most $\frac{c_1 \delta}{\varepsilon} + c_2$ for $\mathcal{R}_I(\delta)$, where $c_1, c_2 > 0$ are some absolute constants. Let $\xi = \xi(|Y \cap I|, |\mathcal{R}_I(\delta)|)$ be its time complexity.

**Assumption 1.** Given some maximal $\delta$-independent set $\mathcal{I}$ for $(Y, R, w)$, the following bound holds:

$$\sum_{I \in \mathcal{I}} \xi(|Y \cap I|, |\mathcal{R}_I(\delta_I)|) \leq L \xi(|Y|, |\mathcal{R}|)$$  \hspace{1cm} (3)

if $\mathcal{R}_I(\delta_I) \cap \mathcal{R}_{I'}(\delta_{I'}) = \emptyset$ for any distinct $I, I' \in \mathcal{I}$, where $\delta_I = \frac{\delta w(Y)}{w(I)}$ and $L > 0$ is some absolute constant.

To perform queries, checking if $w(R_1 \cap R_2) > \delta w(Y)$ for distinct $R_1, R_2 \in \mathcal{R}$ and $0 < \delta < 1$, a data structure is applied whose query and preprocessing times are denoted by $\psi = \psi(|Y|, |\mathcal{R}|)$ and $\gamma = \gamma(|Y|, |\mathcal{R}|)$ respectively. Finally, it is assumed that the overall space cost is at most linear for both the auxiliary procedure and the data structure with respect to cost of storing of range space $(Y, R, w)$.

Let $\alpha, \beta, \tau > 0$ and $0 < \theta_0 < 1$ be some Hitting Set problem specific parameters to be defined later. The following general algorithm returns an $\varepsilon$-net for $(Y, R, w)$.\footnote{for $\varepsilon \geq 1$ it returns an empty set by default}
Lemma 2. Suppose there are absolute constants \( y \) at most \( \varepsilon \). Then, under the assumption 1, an \( (\alpha, \beta, \tau) \) procedure with its parameter equal to \( \frac{\delta}{\omega(Y)} \) is applied to guarantee the bound \(|H_{\theta_0}| = O \left( \frac{1}{\varepsilon} \right) \) for the epsilon net \( H_{\theta_0} \) it returns. It inherits main ideas of work \([20]\) but establishes better upper bounds for \(|H_{\theta_0}|\) by at least factor of 2 assuming relative simplicity of the range space \((Y \cap I, \mathcal{R}_{e,I}, w)\).

Lemma 2. Suppose there are absolute constants \( \alpha, \beta, \tau \) and a graph \( G_I = (I, U) \) for any \( I \subseteq \mathcal{R} \) such that \( |U| \leq \beta |I| \) and \( m_I(y) \geq \alpha n_I(y) - \tau \) for every \( y \in Y \), where \( n_I(y) = |\{ I \in I : y \in I \}| \) and \( m_I(y) = |\{ I, I' \} \in U : y \in I \cap I' \}| \). Then, under the assumption 1, an \( \varepsilon \)-net can be constructed for a range space \((Y, \mathcal{R}, w)\) by the Epsilon Net Finder subalgorithm for any \( 0 < \varepsilon < 1 \) of size at most

\[
\left( 1 + \frac{1}{1 + \frac{c_2 \alpha}{c_1 \beta}} \right) \left( \frac{2c_1 \tau \beta}{\alpha^2} + \frac{c_2 \tau}{\alpha} \right) + \frac{c_2 \tau}{\alpha \sqrt{1 + \frac{c_2 \alpha}{c_1 \beta}}} \frac{1}{\varepsilon} \]

in \( O \left( \frac{\varepsilon}{\alpha^2 \omega(|Y|)} + \gamma + \xi \right) \) time, where \( \xi = \xi(|Y|, |\mathcal{R}_e|) \) and

\[
\theta_0 = \frac{\delta}{\alpha \sqrt{1 + \frac{c_2 \alpha}{c_1 \beta}}} \quad (4)
\]

Proof of the lemma can be found in Appendix C.

Definition 5. Given a space \((Y, \mathcal{R})\), a map is called a structural map for \((Y, \mathcal{R})\) which assigns a graph \( G_I \) for each \( I \subseteq \mathcal{R} \) as defined in the lemma \( \delta \) where the constants \( \alpha, \beta, \gamma \) are referred to as structural parameters for that space.

Remark 1. Range spaces \((Y \cap I, \mathcal{R}_{e,I})\) might have structural maps with smaller ratios \( \frac{\beta}{\alpha} \) and \( \frac{\beta}{\alpha} \) than those for the whole space \((Y, \mathcal{R})\). Thus, one might design those special procedures to seek hitting sets for \( \mathcal{R}_{e,I} \) of size at most \( \frac{c_2 \omega(Y)}{\omega(Y)} + c_2 \) with small constants \( c_1 \) and \( c_2 \), which are based on seeking small independent sets for \((Y \cap I, \mathcal{R}_{e,I}, w)\).

Adjusting the overlapping parameter \( \theta_0 \) demonstrates a tradeoff between the obtained upper bounds on both \( |I| \) and \( \frac{\sum_{H_I}}{|I|} \). In practical situation it might be better to try different values of \( \theta_0 \) to minimize \(|H_{\theta_0}|\).
3. Constant factor approximation algorithms for the IPGD problem

At the core of our algorithms for the IPGD problem two algorithms lie which are the Parametric Agarwal-Pan algorithm and the Epsilon Net Finder subalgorithm, given in subsections 2.1 and 2.2 respectively. According to lemmas 1 and 2 to transform combination of these abstract algorithms into a working approximation algorithm for the IPGD problem, one needs to design algorithms which implement their basic procedures initially considered black boxes by exploiting geometric specifics of the range space \((Y_0, N_r(E))\).

Among those black boxes are algorithms for computing hitting sets for sets of dependent objects within the Epsilon Net Finder subalgorithm and those for maintaining of geometric data structures to perform range searching queries throughout the Parametric Agarwal-Pan algorithm. Moreover, according to the lemma 2, one needs to identify IPGD problem specific structural parameters in order to guarantee upper bounds for approximation factor of the resulting compound algorithm. In subsequent subsections 3.1 and 3.2 these IPGD problem specific algorithms and parameters are identified, giving favourable performance bounds for this compound algorithm. They use some interesting insights in geometry of Euclidean \(r\)-neighbourhoods of straight line segments.

Some additional assumptions are introduced to simplify our work below. Namely, it can be assumed that

\[
N_0 = \{ N \in N_r(E) : \forall M \in N_r(E \setminus \{e(N)\}) M \cap N = \emptyset \} = \emptyset.
\]

Indeed, let \(E' = \{ e \in E : N_r(e) \notin N_0 \}\). One can easily transform any solution \(H'\) of the IPGD problem for \(E'\) into a solution \(H\) of the IPGD problem for \(E\) by setting \(H := H' \cup \{ u_0(e(N)) : N \in N_0 \}\), where \(u_0(e)\) denotes midpoint of segment \(e \in E\). Moreover, all objects can be easily identified from \(N_0\) in \(O(n^2)\) time if \(N_0 \neq \emptyset\).

3.1. Estimating structural parameters \(\alpha, \beta\) and \(\tau\)

To guarantee \(O(\frac{1}{\varepsilon})\) bounds for size of epsilon nets, obtained from the Epsilon Net Finder subalgorithm, a structural map should be identified for the respective range space \((Y_0, N_r(E))\) according to the lemma 2. In other words, one needs to build a structural map by assigning a graph for each subset \(N \subseteq N_r(E)\) or, equivalently, for each subset \(E' \subseteq E\). Below it is established that Delaunay triangulation graph of the segment set \(E'\) turns out to be the sought graph for which ratios \(\frac{\beta}{\alpha}\) and \(\frac{\tau}{\alpha}\) are small.

An elaborate upper bound is presented for \(\beta\). It depends on the parameter, which is equal to the minimum relative complexity of bd conv \(E'\) over all subsets \(E' \subseteq E\). An example is provided in the lemma 4 below where this parameter can be accurately estimated.

Delaunay triangulations can be defined \([8]\) for planar segment sets of non-overlapping straight line segments in assumption of their general position:

1. no quadruple exists of segments from \(E\) which is touched by any single disk;
2. the set is in general position of endpoints of segments from $E$.

**Definition 6.** Let $F$ be the maximal set of open non-overlapping triangles each of which has its endpoints lying on 3 distinct segments from $E$ and its open circumscribing disk does not intersect any segment from $E$. The complement

$$
\text{conv} \left( \bigcup_{e \in E} e \right) \setminus \left( \bigcup_{f \in F} f \cup \bigcup_{e \in E} e \right)
$$

is a union of a set $U$ of relatively open connected components, where closure of each component intersects exactly two segments from $E$. A triple $T_E = (E, U, F)$ is called a Delaunay triangulation of the segment set $E$. A graph $G_E = (E, U_2)$ is called a graph for $T_E$, where $U_2$ consists of those unordered pairs $e, e' \in E$ for which there exists $u \in U$ with $e \cap \text{cl} u \neq \emptyset$ and $e' \cap \text{cl} u \neq \emptyset$.

It is shown in section 4 of [8] that a Delaunay triangulation $T_E$ is uniquely defined by a set $E$ of non-overlapping segments in general position. Moreover, its graph $G_E$ is planar and dual to the graph of Voronoi diagram for $E$.

Below a parameter $\sigma$ is introduced which governs magnitude of $\beta$. Let

$$m(E') = \left| \left\{ e \in E' : e \cap \text{bd conv} \left( \bigcup_{e \in E'} e \right) \neq \emptyset \right\} \right| \text{ for } E' \subseteq E.$$ Let also

$$\sigma = \min_{N \subseteq N_t(E)} \frac{m(E(N))}{|N|}.$$

**Lemma 3.** For range space $(Y_0, N_t(E))$ there is a structural map with $\beta = 3 - \sigma$ and $\alpha = \tau = 1$.

**Proof.** It can be assumed that $E$ contains pairwise non-intersecting segments. Indeed, we can consider the IPGD problem for the same segment set $E$ with radius $r + \rho$ instead of $r$, where a small constant $\rho > 0$ guarantees meeting the following conditions:

1. $\{N \in N_t(E) : y \in N\} = \{N \in N_{r+\rho}(E) : y \in N\}$ for every $y \in Y_0$;
2. a subset of $N_t(E)$ has empty intersection iff the respective subset of $N_{r+\rho}(E)$ has no common points.

Then each segment $[v_1, v_2] \in E$ is replaced by the segment $[v_1 + \kappa(v_2 - v_1), v_2 - \kappa(v_2 - v_1)]$ (denote the set of segments thus obtained by $E_\kappa$), where a small $\kappa = \kappa(v_1, v_2) > 0$ guarantees that the same conditions are met for $N_{r+\rho}(E_\kappa)$ instead of $N_{r+\rho}(E)$, $y \in \text{int} \bigcap_{N \in N_{r+\rho}(E_\kappa):y \in N} N$ for each $y \in Y_0$ and the parameter $\sigma$ is kept unchanged, where $\text{int} N$ denotes the set of interior points of a set $N \subset \mathbb{R}^2$.

Suppose a structural map is build for $(Y_0, N_{r+\rho}(E_\kappa))$ with $\alpha = \tau = 1, \beta = 3 - \sigma$ and a graph $G_{\tau,\kappa}$ corresponds to a subset $\mathcal{I}_{\rho,\kappa}$ under this map, where segments from $E_\kappa(\mathcal{I}_{\rho,\kappa})$ are shortened segments from $E(\mathcal{I})$ for $\mathcal{I} \subseteq N_{r+t}(E)$. It is obvious that the same graph can be assigned for the set $\mathcal{I}$ taking into account that
\{N \in \mathcal{I} : y \in N\} = \{N \in \mathcal{I}_{\infty} : y \in N\} \text{ for any } y \in Y_0. \text{ Thus, it is assumed with slight abuse of terminology that } y \in \text{int } \bigcap_{N \in \mathcal{I}, y \in N} N \text{ for each } y \in Y_0 \text{ and } E \text{ consists of non-overlapping segments. Moreover, it can be assumed that the segment set } E \text{ is in general position.}

Let \mathcal{I} \subseteq N_r(E) \text{ and } G_\mathcal{I} \text{ be the maximal graph which is obtained from a Delaunay triangulation graph for } E(\mathcal{I}) \text{ by removing redundant multiple edges. Due to the theorem 3 from } \cite{8}, \text{} G_\mathcal{I} \text{ contains at most } 3|E(\mathcal{I})| - k - 3 \text{ edges, where } k \text{ denotes number of those edges of conv } \left( \bigcup_{e \in E(\mathcal{I})} e \right), \text{ which are not segments of } E(\mathcal{I}). \text{ As segments from } E(\mathcal{I}) \text{ are non-intersecting, } m(E(\mathcal{I})) \leq k \text{ and } G_\mathcal{I} \text{ has at most } \beta|E(\mathcal{I})| \text{ edges.}

Let \mathcal{I}(y) = \{I \in \mathcal{I} : y \in I\} \text{ and } G_\mathcal{I}(y) \text{ be a subgraph of } G_\mathcal{I} \text{ induced by the subset } E(\mathcal{I}(y)) \text{ as its set of } n_\mathcal{I}(y) \text{ vertices. Let us prove that } \alpha = \tau = 1 \text{ by induction on } n_\mathcal{I}(y) \text{ for every } y \in Y_0. \text{ The case } n_\mathcal{I}(y) = 1 \text{ is obvious. Let us assume that any graph } G_\mathcal{I}(y) \text{ with } n_\mathcal{I}(y) \leq k \text{ vertices contains at least } n_\mathcal{I}(y) - 1 \text{ edges (for any } r) \text{ and suppose that } n_\mathcal{I}(y) = k + 1.

By perturbing } y \text{ within } \bigcap_{I \in \mathcal{I}(y)} I \text{ it can be achieved that segments from } E(\mathcal{I}(y)) \text{ are at distinct distances from } y. \text{ Besides, let } e_0(y) \in E(\mathcal{I}(y)) \text{ be the farthest (among segments of } E(\mathcal{I}(y))) \text{ segment from } y. \text{ Denote by } y_0 \text{ Euclidean projection of } y \text{ onto } e_0(y). \text{ There is a point } y_1 \in [y, y_0] \text{ which is equidistant from } e_0(y) \text{ and some segment } e(y) \in E(\mathcal{I}(y)) \setminus \{e_0(y)\} \text{ whereas none of segments of } E(\mathcal{I}) \setminus \{e_0(y), e(y)\} \text{ is within the distance } \|y_0 - y_1\|_2 \text{ from } y_1 \text{ (again may be after a small perturbation of } y). \text{ Due to duality between Delaunay triangulations and Voronoi diagrams considered over the same segment set (see the theorem 4 from } \cite{8}, \text{ we get that } G_\mathcal{I}(y) \text{ contains an edge which connects } e_0(y) \text{ and } e(y). \text{ Obviously, each segment of } E(\mathcal{I}(y)) \text{ has nonempty intersection with the radius } r \text{ disk centered at } y. \text{ Let } \gamma > 0 \text{ be so small such that } r_0 = \|y - y_0\|_2 - \gamma \text{ radius disk centered at } y \text{ intersects all } n_\mathcal{I}(y) - 1 \text{ segments from } E(\mathcal{I}(y)) \setminus \{e_0(y)\}. \text{ Let } G_\mathcal{I}(y, \gamma) \text{ be the subgraph of } G_\mathcal{I}(y) \text{ induced by segments of } E(\mathcal{I}(y)) \setminus \{e_0(y)\}. \text{ Applying inductive assumption, we have that } G_\mathcal{I}(y, \gamma) \text{ has at least } n_\mathcal{I}(y) - 2 \text{ edges. Thus, the graph } G_\mathcal{I}(y) \text{ contains at least } n_\mathcal{I}(y) - 1 \text{ edges.}

\textbf{Remark 2.} If } E \text{ consists of non-overlapping segments, a structural map is } r-\text{ and } Y_0\text{-independent from the proof above.}

\textbf{Lemma 4.} \text{ If each segment from } E \text{ has nonempty intersection with } \text{bd conv } \left( \bigcup_{e \in E} e \right), \text{ there is a structural map for } (Y_0, N_r(E)) \text{ with } \alpha = \tau = \sigma = 1 \text{ and } \beta = 2.

\textbf{3.2. Hitting set finders for sets of dependent objects}

In order for the Epsilon Net Finder subalgorithm to work, an auxiliary procedure should be designed to perform its step 2. The procedure must return...
hitting sets of length at most $\frac{c_1}{d} + c_2$ for sets $N_I(\delta) \subseteq \{N \in N_r(E) : w(N \cap I) > \delta w(I)\} \cup \{I\}$, where $I \in N_r(E)$ and $c_1$, $c_2 > 0$ are some absolute constants.

Five procedures are constructed below resulting in small constants $c_1$ and $c_2$ for different configurations of sets of straight line segments. More specifically, first procedure treats general case of the IPGD problem with segments intersecting at most at their endpoints, giving $c_1 = 8$ and $c_2 = 2$. Second procedure is designed for the case where $E$ is such that either $d(e, e') > r$ or $d(e, e') = 0$ for any distinct $e, e' \in E$, where $d(e, e')$ denotes Euclidean distance between $e$ and $e'$. It reports $c_1 = 1$ and $c_2 = 6$.

The other three procedures tackle cases in which $E$ is being a subset of edge set of either a Gabriel graph or a Delaunay triangulation. Such plane graphs arise in network routing and modelling applications. Namely, third procedure provides $c_1 = 0$ and $c_2 = 18$ in the case of Gabriel graphs. Fourth and fifth procedures are designed for general and outerplane Delaunay triangulations with $c_1 = 4$, $c_2 = 10$ and $c_1 = 5$, $c_2 = 4$ respectively.

To get such a small constants, geometry of Euclidean $r$-neighbourhoods of segments is heavily exploited in these procedures, which largely amount to fast construction of the least cardinality hitting sets for sets of 1-dimensional intervals on the real line.

It is assumed below that segments of $E$ are all of non-zero length w.l.o.g. Indeed, one can easily design a finder for the case $I = N_r(x)$ with $x \in \mathbb{R}^2$, which gets a $7$-point set in $O(1)$ time to hit all objects from $N_I$.

3.2.1. General case of planar $G$

The following observations can be made about shape of Euclidean $r$-neighbourhoods of segments.

**Observation 1.** Let $e, e' \in E$ be such that $M = N_r(e) \cap N_r(e') \neq \emptyset$. Then $M = N_r(z(e')) \cap N_r(e') = N_r(e) \cap N_r(z(e'))$, where $z(e') = \{x \in e' : d(x, e) \leq 2r\}$.

Let $l(e)$ be a straight line through $e$ for some non-zero length segment $e \in E$ and $h_1(e)$ and $h_2(e)$ be positive and negative halfplanes respectively whose boundary coincides with $l(e)$; here orientation is chosen arbitrarily for $l(e)$. The set bd $N_r(e)$ can be represented in the form of a union of two halfcircles and two segments $f_1(e)$ and $f_2(e)$, where $f_i(e) \subset \text{int } h_i(e)$, $i = 1, 2$. Let $l_i(e)$ be the straight line through $f_i(e)$.

**Observation 2.** Let $\{v_1, v_2\} = l(e) \cap \text{bd } N_r(e)$, $e \in E$. For every $e, e' \in E$ with $N_r(e) \cap N_r(e') \neq \emptyset$ or such $i_0 \in \{1, 2\}$ exists that $d(x, l_{i_0}(e)) \leq r$ for each $x \in z(e')$.

The procedure below seeks hitting sets for sets of dependent objects in general case where the only restriction is that segments are allowed from $E$ to intersect at most at their endpoints. It is based on finding hitting sets for sets of 1-dimensional $r$-neighbourhoods of (interval) projections of segments from $\{z(e')\}_{e' \in E'}$, $E' \subseteq E$, onto straight lines $l_i(e)$. Let $N_r(f) = \{x \in l_i(e) : d(x, f) \leq r\}$ for an arbitrary interval $f \subset l_i(e)$, $i = 1, 2$. The following folklore
lemma reports on the complexity of getting minimum cardinality hitting set for a set of 1-dimensional intervals. Its proof is left for the appendix.

**Lemma 5.** The minimum cardinality hitting set can be found for a set of $n$ 1-dimensional intervals on the real line in $O(n \log n)$ time and $O(n)$ space.

### Hitting Set Finder for Dependent Objects

**Input:** a parameter $0 < \delta < 1$ and a set $\mathcal{N}_I(\delta)$, where $I \in \mathcal{N}_r(E)$ and $\mathcal{N}_I(\delta) \subseteq \{N \in \mathcal{N}_r(E) : w(N \cap I) > \delta w(I)\} \cup \{I\}$ for some weight map $w : Y_0 \to \mathbb{R}_+$;

**Output:** a hitting set $H \subseteq \mathbb{R}^2$ for $\mathcal{N}_I(\delta)$.

1. set $\{v_1, v_2\} = I(e(I)) \cap \text{bd}I$ and
   \[\mathcal{P} := \mathcal{N}_I(\delta) \setminus \{N \in \mathcal{N}_r(E) : N \cap \{v_1, v_2\} \neq \emptyset\};\]
2. form sets $Z_i = \{z(c(I))(e) : e \in E(\mathcal{P}), z(c(I))(e) \subset h(e(I))\}$, $i = 1, 2$;
3. form a set $P_i$ of orthogonal projections of segments from $Z_i$ onto the straight line $l_i(e(I))$ and construct sets $P_i(r) = \{N_{1r}(p) : p \in P_i\}$, $i = 1, 2$;
4. find the minimum cardinality hitting set $H_i \subset l_i(e(I))$ for $P_i(r)$, $i = 1, 2$,
   as in the proof of the lemma [5];
5. for each $x_0 \in H_i$ and $i = 1, 2$ construct a set $S(x_0)$ of 4 points such that
   \[N_{\sqrt{2}r}(x_0) \subset \bigcup_{x \in S(x_0)} N_r(x)\]
   and return a set $H = \{v_1, v_2\} \cup \bigcup_{x \in S(x_0)} S(x_0)$.

The following lemma summarizes on the procedure performance.

**Lemma 6.** Let $m = |\mathcal{N}_I(\delta)|$. The **Hitting Set Finder for Dependent Objects** procedure returns a hitting set $H$ for $\mathcal{N}_I(\delta)$ of size at most $\frac{3}{2}m + 2$ in $O(m \log m)$ time and $O(m)$ space.

**Proof.** All steps except for step 4 of the procedure require $O(m)$ time whereas step 4 takes $O(m \log m)$ time according to the lemma [5]. It remains to get the bound $|H| \leq \frac{3}{2}m + 2$ and prove that $H$ is a hitting set for $\mathcal{N}_I(\delta)$. Indeed, due to step 1 and observation 2 one has either $z(c(I))(e) \subset h_1(e(I))$ or $z(c(I))(e) \subset h_2(e(I))$ for every $e \in E(\mathcal{P})$. Moreover, each interval $J \in P_i(r)$ is an orthogonal projection of some object $P_i^{-1}(J) \in \mathcal{N}_r(Z_i)$. According to proof of the lemma [5] for each $i = 1, 2$ at step 4 a maximal subset $Q_i \subseteq P_i(r)$ is build of pairwise non-overlapping intervals with $|Q_i| = |H_i|$. Thus, the respective set $\{P_i^{-1}(J) : J \in Q_i\}$ consists of non-intersecting objects. By observation 1 we have that $w(P_i^{-1}(J) \cap I) > \delta w(I)$ for all $J \in Q_i$. Therefore $|Q_i| \leq \frac{1}{\delta}$ and $|H| = 4|Q_1| + 4|Q_2| + 2 \leq \frac{3}{2}m + 2$.

By observation 2 each point of segment from $Z_i$ is within the distance $r$ from $l_i(e(I))$. Therefore each segment of $Z_i$ is within $\sqrt{2}r$ distance from some point of $H_i$. By construction at step 5 we get that $H$ is a hitting set for $\mathcal{N}_I(\delta)$.

**Remark 3.** The **Hitting Set Finder for Dependent Objects** procedure in fact returns a set which hits all objects from $\mathcal{N}_I^1 = \{N \in \mathcal{N}_r(E) : e(N) \cap \\text{bd}I \neq \emptyset\}$. Thus, one can choose those objects at step 1 of the **Epsilon Net Finder** algorithm to include into the growing $\delta$-independent set $I$, which are from $\mathcal{N}_r(E) \setminus \bigcup_{I \in I^1} \mathcal{N}_I^1$. 

17
3.2.2. Case of remote edges of $G$

It is not hard to observe that, given $I \in \mathcal{N}_r(E)$, a hitting set can be efficiently constructed for $\mathcal{N}_I(\delta)$ of much smaller size in the case, where either $d(e, e') > r$ or $d(e, e') = 0$ for any distinct $e, e' \in E$. Indeed, this is possible because the set $\mathcal{N}_I(\delta) \setminus \mathcal{N}_I^1$ can be converted into the set $\{\text{bd} I \cap N \in \mathcal{N}_I(\delta) \setminus \mathcal{N}_I^1\}$ of 1-dimensional arcs, keeping those nonempty mutual intersections of objects from $\mathcal{N}_I(\delta)$ which are nonempty within $I$. Due to this “equivalency”, hitting sets can be quickly computed in such case of size at most $\frac{c}{\delta} + c_2$ for sets $\mathcal{N}_I(\delta)$ with the smaller constant $c_1$ than that constant guaranteed by the lemma for the Hitting Set Finder for Dependent Objects procedure.

For any object $I \in \mathcal{N}_r(E)$ let $C(I)$ be the set of 4 endpoints of segments $f_i(e(I)), i = 1, 2$, and $U(I) = C(I) \cup (l(e(I)) \cap \text{bd} I)$, where $|U(I)| = 6$. As a start, a simple observation can be made about shape of Euclidean $r$-neighbourhoods of nonzero length segments.

**Lemma 7.** Let $I, N_1, N_2 \in \mathcal{N}_r(E)$ be distinct and $d(e(I), e(N)) \in (r, 2r)$ for $i = 1, 2$. If $I \cap N_1 \cap N_2 \neq \emptyset$, then either $N_1 \cap N_2 \cap \text{bd} I \neq \emptyset$ or $N_1 \cap U(I) \neq \emptyset$ for some $i_0 \in \{1, 2\}$.

**Proof.** Let $\chi_i = \text{bd} I \cap N_i$ and $\pi_i = \text{bd} N_i \cap I$ for $i = 1, 2$. Assume that $N_i \cap U(I) = \emptyset$ for all $i \in \{1, 2\}$. It should be proved that $\chi_1 \cap \chi_2 \neq \emptyset$ if $I \cap N_1 \cap N_2 \neq \emptyset$. Let $p(x) \in e(I)$ be Euclidean projection of $x$ onto $e(I)$ for $x \in \mathbb{R}^2$. It is sufficient to establish the following monotonicity property: for any $x \in \text{bd} I$ and $i = 1, 2$ nonempty intersection $[p(x), x] \cap N_i$ is a (possibly zero-length) segment with endpoint in $x$. Indeed, for $x \in I \cap N_1 \cap N_2$ this implies that the ray with the origin $p(x)$ and direction $x - p(x)$ intersects $\text{bd} I$ at some point of $\chi_1 \cap \chi_2$.

Suppose, in contrary, there is a point $x_0 \in \text{bd} I$ and $i_0 \in \{1, 2\}$ such that the interval $(p(x_0), x_0)$ has two (possibly identical) points $x'_i$ and $x''_i$ of intersection with $\pi_{i_0}$. There is a point $x' \in [p(x_0), x_0]$ and a endpoint $x'' \in e(N_{i_0})$ with $(x' - x'', x_0 - p(x_0)) = 0$ such that $d(x', x'') \leq r$. It implies inclusion $x'' \in \bigcup_{x \in U(I)} N_i(x)$ taking into account that $r < d(e(I), e(N_{i_0})) \leq 2r$. But this inclusion is impossible by our assumption that $N_{i_0} \cap U(I) = \emptyset$.

Below an assumption is imposed on segments from $E(\mathcal{N}_I(\delta))$.

**Assumption 2.** Given $0 < \delta < 1$ and $I \in \mathcal{N}_r(E)$ $d(e, e(I)) > r$ for all $e \in E(\mathcal{N}_I(\delta)) \setminus \{e \in E : N_r(e) \cap U(I) \neq \emptyset\}$.

Under this assumption, a special procedure is presented which builds a short length hitting set for $\mathcal{N}_I(\delta)$. To do this, it constructs a minimum cardinality hitting set for the set of “1-dimensional” intervals $\{N \cap \text{bd} I : N \in \mathcal{N}_I(\delta), N \cap U(I) = \emptyset\}$.

---

$\langle \cdot, \cdot \rangle$ denotes Euclidean scalar product in $\mathbb{R}^2$.  

---

18
**Hitting Set Finder for Dependent Objects**

**Input:** a parameter $0 < \delta < 1$, an object $I \in \mathcal{N}_r(E)$ and a set $\mathcal{N}_I(\delta)$;

**Output:** a hitting set $H \subset \mathbb{R}^2$ for $\mathcal{N}_I(\delta)$.

1. set $\mathcal{P} := \mathcal{N}_I(\delta) \setminus \{N \in \mathcal{N}_r(E) : N \cap U(I) \neq \emptyset\}$ and $J := \{N \cap \text{bd} I : N \in \mathcal{P}\}$;
2. applying polar coordinates, find the minimum cardinality hitting set $H'$ for $J$ as in proof of the lemma 5;
3. return $H = H' \cup U(I)$.

If the assumption 2 holds, tighter bounds can be obtained (based on the lemma 7) for size of the hitting set for $\mathcal{N}_I(\delta)$ which the **Hitting Set Finder for Dependent Objects** procedure gives than those reported in the lemma 6 for hitting sets returned by the **Hitting Set Finder for Dependent Objects** procedure.

**Lemma 8.** Let $m = |\mathcal{N}_I(\delta)|$ and $\mathcal{P} = \mathcal{N}_I(\delta) \setminus \{N \in \mathcal{N}_r(E) : N \cap U(I) \neq \emptyset\}$. If $E(\mathcal{P})$ consists of segments at the distance more than $r$ from $e(I)$, then the **Hitting Set Finder for Dependent Objects** procedure returns a hitting set $H$ for $\mathcal{N}_I(\delta)$ of size at most $\frac{1}{\delta} + 6$ in $O(m \log m)$ time and $O(m)$ space.

**Proof.** The set $H$ gives a hitting set for $\mathcal{N}_I(\delta)$ as $H'$ is a hitting set for $J$ by construction reported in proof of the lemma 5. Thus, it remains to estimate $|H'|$. As byproduct of this construction one gets a maximal subset $J'$ of non-overlapping arcs from $J$ with $|H'| = |J'|$. Let $\mathcal{P}' \subseteq \mathcal{P}$ be the subset such that $J' = \{N \cap \text{bd} I : N \in \mathcal{P}'\}$. Due to the lemma 7 $\mathcal{P}'$ consists of non-overlapping objects within $I$. Therefore $|H'| \leq \frac{1}{\delta}$.

Of course, when segments from $E$ are all at Euclidean distance either zero or more than $r$ from each other, the **Hitting Set Finder for Dependent Objects** procedure returns a small sized hitting set for $\mathcal{N}_I(\delta)$ as the previous lemma claims. As an another application of this lemma, it is demonstrated below that its assumptions are also hold when $E$ is a subset of edge set of a Gabriel graph.

**Lemma 9.** In notation of the previous lemma, if $G = (V,E)$ is any subgraph of a Gabriel graph, then $E(\mathcal{P})$ consists of segments which are at the distance more than $r$ from $e(I)$.

**Proof.** By definition of Gabriel graph the closed disk $D$ does not contain endpoints of segments of $E \setminus \{e(I)\}$ whose diameter is $e(I)$. Of course, $\text{bd} I \subset \bigcup_{x \in U(I)} N_r(x)$ if length of $e(I)$ is less than or equal to $2r$. Therefore $d(e(I), e) > r$ for any $e \in E(\mathcal{P})$. Suppose that length of $e(I)$ is more than $2r$. In this case $I \setminus D \subset \bigcup_{x \in U(I)} N_r(x)$ which leads to the same conclusion about $E(\mathcal{P})$. 

19
Corollary 1. If \( G \) is any subgraph of a Gabriel graph, then \(|H| \leq \frac{1}{2} + 6\) for a hitting set \( H \) returned by the Hitting Set Finder for Dependent Objects* procedure.

In the next part a fast algorithm is designed which returns constant sized hitting sets for \( N_I \), given any \( I \in N_v(E) \). It explicitly exploits the assumption that \( E \) is subset of edge set of a Gabriel graph, giving additional gain of \( c_1 = 0 \) and \( c_2 = 18 \). Though, the Hitting Set Finder for Dependent Objects* procedure turns out to be helpful in design of a hitting set finder algorithm in the case where \( E \) is subset of edge set of an outerplane Delaunay triangulation.

3.2.3. Case of special proximity graph \( G \)

Special segment configurations are considered below which are interesting for network applications. They are produced from planar finite point sets \( V \) using the so called empty disk property: two points \( u, v \in V \) are joined by a straight line segment when a disk exists which contains \( u \) and \( v \) on its boundary and does not contain any other points from \( V \). This property makes the other segments avoid having their endpoints in that disk. It is shown that it leads to short length hitting sets to exist for sets of dependent objects.

Gabriel graphs. Let \( V \) be a point set in general position on the plane no 4 of which are cocircular.

Definition 7. A plane graph \( G = (V,E) \) is called a Gabriel graph when \([u,v] \in E \) iff the disk does not contain any other points of \( V \) distinct from \( u \) and \( v \), which has \([u,v]\) as its diameter.

Let us start with the case where \( E \) is subset of edges of a Gabriel graph. In this case only a modest constant sized set of points is needed to hit the set \( N_I(\delta) \), given any \( I \in N_v(E) \). This point set (denoted by \( U_0(I) \)) is constructed below as follows. Given \( s \in \{1,2\} \), let \( u_0 = \frac{u_1 + u_2}{2} \) and \( U_s(I) \) be a set with \(|U_s(I)| = 7\) such that \( N_{2r}(u_s) \subset \bigcup_{u \in U_s(I)} N_r(u) \), where \( u_1 \) and \( u_2 \) are endpoints of \( e(I) \). For \( i = 1,2 \) points of \( U_s(I) \) can be arranged to get the inclusion

\[
N_{2r}(u_s) \cap h_i(e(I)) \subset \bigcup_{u \in U_s(I)} N_r(u)
\]

hold, where \( U_s(I) = U_s(I) \cap h_i(e(I)) \) and \(|U_s(I)| \leq 4\).

For \( i, s = 1,2 \) let \( g_i \subset h_i(e(I)) \) also be the straight line with \( d(g_i,e(I)) = 2r \), which is parallel to \( e(I) \); set \( u_{si} = g_i \cap N_{2r}(u_s) \). Finally, denote halved Euclidean length of \( e(I) \) by \( \Delta \). If \( \Delta \in [0, \sqrt{2r}] \), consider points \( z_i = bd N_{2r}(u_1) \cap bd N_{2r}(u_2) \cap h_i(e(I)) \) and \( a_{si} = \frac{u_{si} + z_i}{2} \). When \( \Delta \in (\sqrt{2r},2r] \), consider a coordinate system centered at \( u_s \) whose \( x \)-axis is along a segment \( e(I) \) towards \( u_3 - s \) whereas \( y \)-axis is towards \( g_i \); set \( a_{si} = (\Delta / 2, \Delta) \). For \( \Delta > 2r \) let \( a_{si} = \frac{u_{si} + u_{2r}}{2} \), where \( b_{si} \in g_i \cap bd N_\Delta(u_0) \) is closest to \( u_{si} \).

Lemma 10. Let \( U_0(I) := U_1(I) \cup U_2(I) \cup \{a_{si}\}_{i,s=1,2} \), where \(|U_0(I)| = 18\). If \( G = (V,E) \) is any subgraph of a Gabriel graph, then \( U_0(I) \cap N \neq \emptyset \) for any \( N \in N_I \) and \( I \in N_v(E) \).
PROOF. To prove the claim it is sufficient to establish the inclusion

\[ N_{2r}(e(I)) \subseteq N_{\Delta}(u_0) \cup \bigcup_{u \in U_0(I)} N_r(u). \]

Indeed, recall that each \( e \in E \setminus \{e(I)\} \) must avoid having its endpoints in \( N_{\Delta}(u_0) \) by definition of Gabriel graph. As segments of \( E \) intersect at most at their endpoints, the inclusion \( e(N) \cap \bigcup_{u \in U_0(I)} N_r(u) \neq \emptyset \) holds true due to the fact that \( e(N) \cap N_{2r}(e(I)) \neq \emptyset \) for every \( N \in \mathcal{N}_I \).

If \( \Delta \in [0, \sqrt{2}r] \), it can be shown that \( |z_i - u_{si}| \leq 2r \). When \( \Delta \in (\sqrt{2}r, 2r] \) radius \( r \) disk covers points \((\Delta, \Delta), (\Delta, 2r), (\Tfrac{2r^2}{\Delta}, 2r\sqrt{1 - \frac{r^2}{\Delta^2}})\) and \( u_s = (0, 2r) \), which is centered at \( u_{si} = (\Delta/2, \Delta) \) for the coordinate system with the origin at \( u_s \), \( s = 1, 2 \). Indeed, one gets \( \frac{2r^2}{\Delta} + (2r - \Delta)^2 \leq r^2 \), checking for values of argument \( \Delta_1 = \sqrt{2}r \) and \( \Delta_2 = 2r \). Moreover, applying calculus it is enough to check for the same values of arguments to get

\[
\left( \frac{\Delta}{2} - \frac{2r^2}{\Delta} \right)^2 + \left( \Delta - 2r\sqrt{1 - \frac{r^2}{\Delta^2}} \right)^2 = \frac{5\Delta^2}{4} + 2r^2 - 4r\Delta^2 - r^2 \leq r^2.
\]

If \( \Delta > 2r \) it remains to prove that

\[
\left( \frac{\Delta - \sqrt{\Delta^2 - 4r^2}}{2} - \frac{2r^2}{\Delta} \right)^2 + \left( 2r - 2r\sqrt{1 - \frac{r^2}{\Delta^2}} \right)^2 \leq r^2.
\]

The first sum term can be estimated from above by \( \frac{4r^4}{\frac{\Delta^2}{\Delta^2 - 4r^2} + \frac{2r^2}{\sqrt{\Delta^2 - 4r^2} + 2}} \leq r^2 \). The second sum term can be bounded by \( \frac{r^2\left(1 - \sqrt{1 - r^2/\Delta^2}\right)}{\Delta^2(1 + \sqrt{1 - r^2/\Delta^2})} < \frac{r^2}{4} \).

**Delaunay triangulations.** One needs more involved hitting set finder in general case where \( G \) is any subgraph of a Delaunay triangulation.

**Definition 8.** A plane graph \( G = (V, E) \) is called a Delaunay triangulation when \( [u, v] \in E \) iff there is a disk \( T \) such that \( u, v \in \text{bd} T \) and \( V \cap \text{int} T = \emptyset \).

Proof of the lemma below describes a procedure which seeks hitting sets for \( \mathcal{N}_I(\delta) \). In fact it uses a combination of ideas appearing both in construction of the set \( U_0(I) \) and the **Hitting Set Finder for Dependent Objects** procedure.

**Lemma 11.** Let \( m = |\mathcal{N}_I(\delta)| \). If \( G = (V, E) \) is any subgraph of a Delaunay triangulation, then a hitting set \( H \) can be built for \( \mathcal{N}_I(\delta) \) within \( O(m \log m) \) time and \( O(m) \) space of size at most \( \frac{4}{3} + 10 \).
Proof. At first a procedure is described which gives a hitting set for \( N_I(\delta) \). Let \( \mathcal{P} := N_I(\delta) \setminus \{ N \in N_I(E) : N \cap U_0(I) \neq \emptyset \} \). If \( \mathcal{P} = \emptyset \), return \( H := U_0(I) \) as a hitting set for \( N_I(\delta) \) of size 18. Otherwise, if \( d(e,e(I)) > r \) for all \( e \in E(\mathcal{P}) \), then a procedure is applied to get a hitting set \( H \) of size at most \( \frac{1}{2} + 18 \) for \( N_I(\delta) \), which analogous to the Hitting Set Finder for Dependent Objects* procedure.

The following argument is conducted when \( E(\mathcal{P}) \) contains those segments which are at the distance from \( e(I) \) less than or equal to \( r \). There is such a disk \( D \) that bd \( D \cap V \) consists of two endpoints of \( e(I) \) and int \( D \cap V = \emptyset \) by definition of Delaunay triangulation. Let \( U_0(I,i) = U_0(I) \cap h_i(e(I)), i = 1,2 \). An index \( i_0 \in \{1,2\} \) can be easily found such that \( N_{2\delta}(e(I)) \cap h_{i_0}(e(I)) \subset D \cup \bigcup_{a \in U_0(I,i_0)} N_a(u) \). Let \( \mathcal{P} = N_I(\delta) \setminus \{ N \in N_I(E) : N \cap U_0(I,i_0) \neq \emptyset \} \). Perform step 2 of the Hitting Set Finder for Dependent Objects procedure for \( \mathcal{P} \) to get sets \( Z_i, i = 1,2 \). One proceeds with this procedure for the subset \( N_I(\delta) \supset N_I(\delta) = \{ P \in \mathcal{P} : z_{e(I)}(e(P)) \in Z_{3-I_i} \} \) to get a set \( H_1 \) of size at most \( \frac{4}{3} \). Thus, a hitting set \( H = H_1 \cup U_0(I,i_0) \) is constructed for \( N_I(\delta) \) of size at most \( \frac{1}{2} + 10 \), having in mind that \( z \cap \bigcup_{x \in U_0(I,i_0)} N_r(x) \neq \emptyset \) for any \( z \in Z_{i_0} \) (see proof of the lemma [10]).

Definition 9. A plane graph \( G = (V,E) \) is called an outerplane if

\[
e \cap \text{bd conv} \left( \bigcup_{e \in E} e \right) \neq \emptyset
\]

for any \( e \in E \).

When \( G \) is an outerplane Delaunay triangulation, a different algorithm is designed in proof of the lemma below. It uses a combination of ideas appearing in both Hitting Set Finder for Dependent Objects and Hitting Set Finder for Dependent Objects* procedures in the way which is analogous to the way taken for general Delaunay triangulations.

Lemma 12. Let \( m = \vert N_I(\delta) \vert \). If \( G = (V,E) \) is any subgraph of an outerplane Delaunay triangulation, a hitting set \( H \) can be build for \( N_I(\delta) \) within \( O(m \log m) \) time and \( O(m) \) space of size at most \( \frac{5}{3} + 4 \).

Proof. Again, a procedure is given first, which provides a hitting set for \( N_I(\delta) \). Let \( \mathcal{P} := N_I(\delta) \setminus \{ N \in N_I(E) : N \cap U(I) \neq \emptyset \} \). If \( d(e,e(I)) > r \) for all \( e \in E(\mathcal{P}) \), the Hitting Set Finder for Dependent Objects* procedure is applied to get a hitting set \( H \) for \( N_I(\delta) \) of size at most \( \frac{1}{2} + 6 \).

Assume that \( E(\mathcal{P}) \) contains those segments which are at the distance from \( e(I) \) less than or equal to \( r \). There is such a disk \( D \) that bd \( D \cap V \) consists of two endpoints of \( e(I) \) and int \( D \cap V = \emptyset \). The object \( I \) is composed of two radius \( r \) half-disks and a rectangle \( R \) which is halved by \( e(I) \) on two rectangles \( R_1 \) and \( R_2 \), where \( R_i \subset \text{cl} h_i(e(I)), i = 1,2 \). We have \( R_{i_0} \setminus D \subset \bigcup_{x \in U(I)} N_r(x) \).
for some $i_0 \in \{1, 2\}$. Let $U(I, i_0) = U(I) \cap clh_{i_0}(e(I))$ and $\mathcal{P} = \mathcal{N}_I(\delta) \backslash \{N \in \mathcal{N}_I(E) : N \cap U(I, i_0) \neq \emptyset\}$. Perform step 2 of the HITTING SET FINDER FOR DEPENDENT OBJECTS procedure for $\mathcal{P}$ to get sets $Z_i, i = 1, 2$. One proceeds with this procedure for the subset $\mathcal{N}_I(\delta) \supseteq \mathcal{N}_I'(\delta) = \{P \in \mathcal{P} : z_{e(I)}(e(P)) \in \mathcal{Z}_{3-i_0}\}$ to get a set $H_{3-i_0}$ of size at most $\frac{4}{5}$. For the subset $\mathcal{N}_I(\delta) \supseteq \mathcal{N}_I''(\delta) = \{P \in \mathcal{P} : z_{e(I)}(e(P)) \in Z_{i_0}\}$ the HITTING SET FINDER FOR DEPENDENT OBJECTS* procedure is applied to get a set $H_{i_0}$ of size at most $\frac{1}{2}$, having the lemma 8 in mind. Thus, a hitting set $H = H_1 \cup H_2 \cup U(I, i_0)$ is obtained for $\mathcal{N}_I(\delta)$ of size at most $\frac{3}{2} + 4$.

3.3. Constant factor approximation algorithms and their performances

In this subsection our main algorithmic results are formulated which are fast constant approximations for the IPGD problem on a set of straight line segments allowed to intersect at most at their endpoints. More accurate approximations are also provided for special geometric configurations of segments which could be of interest in network applications.

3.3.1. General case

Our first result is on constant factor approximation for general case of the IPGD problem.

**Theorem 1.** There is a $\left(50 + 52\sqrt{\frac{\nu}{3}} + \nu\right)$-approximation for the IPGD problem, which works in

$$O \left( \left( n^2 + \frac{n \log n}{\nu^2} + \frac{\log n}{\nu^3} \right) n^2 \log n \right)$$

time and $O \left( \frac{n^2 \log n}{\nu} \right)$ space for any small $\nu > 0$.

**Proof.** One can compute a set $Y_0$ of $O(n^2)$ vertices of the arrangement formed by curves from $\{bdN\}_{N \in \mathcal{N}_I(E)}$ in $O(n^2)$ time.

To construct required approximation algorithm for the IPGD problem one employs the PARAMETRIC AGARWAL-PAN algorithm for $(Y_0, \mathcal{N}_I(E))$ which incorporates the EPSILON NET FINDER subalgorithm for $\alpha = \tau = 1$ and $\beta = 3$, where $\theta_0$ is chosen according to the equation (4) of the lemma 2 whereas constants $c_1$ and $c_2$ are reported in the lemma 6. The HITTING SET FINDER FOR DEPENDENT OBJECTS procedure (see subsection 3.2) is used at step 2 of this subalgorithm as an auxiliary procedure to seek hitting sets for subsets $\mathcal{N}_{e,t} \subseteq \{N \in \mathcal{N}_I(E) : w(N) > \varepsilon w(Y_0), w(N \cap I) > \theta_0 \varepsilon w(Y_0)\}$, build at the subalgorithm step 1. Naive range counting and reporting data structures are used throughout the PARAMETRIC AGARWAL-PAN algorithm which maintain times $\tau_1 = \omega = \psi = O(|Y_0|)$ and $\tau_2 = \tau^{(0)} = \omega^{(0)} = \gamma = O(1)$.

Sets $\mathcal{N}_{e,t}$ are disjoint. Due to the lemma 9 the assumption 1 holds for complexity $\xi$ of the HITTING SET FINDER FOR DEPENDENT OBJECTS procedure. As $\frac{1}{\varepsilon} = O(k) = O(OPT)$ in the PARAMETRIC AGARWAL-PAN algorithm.
with parameter $k$, the Epsilon Net Finder subalgorithm works in $O(n^4)$ time taking into account lemmas 2 and 6. Substituting found time bounds into the equation (2) of the lemma 1 one gets $O(n^4 \log n)$ time and $O(n^2 \log n)$ space complexity of the compound algorithm.

Taking lemmas 2, 3 and 6 into account, one gets that the Epsilon Net Finder subalgorithm returns a weak $\varepsilon$-net of size at most $\left(50 + 52 \sqrt{\frac{12}{13}}\right) \frac{1}{\varepsilon}$. Due to the lemma 1 one can adjust parameters $\delta, \eta, \lambda_1$ and $\mu$ of the Parametric Agarwal-Pan algorithm to get $\left(50 + 52 \sqrt{\frac{12}{13}} + \nu \right)$-approximation without affecting orders of its time and space complexities for any small $\nu > 0$. More specifically, setting $\mu := 1, \eta := \frac{\nu}{300}, \lambda_1 = \delta := \frac{\nu}{600}$, it can be shown that $\left(50 + 52 \sqrt{\frac{12}{13}} \left(\frac{\nu}{600} + (1 + \frac{\nu}{300}) e^{\nu/600}\right)\right) \leq \left(50 + 52 \sqrt{\frac{12}{13}}\right) + \nu$ for small $\nu > 0$. Moreover, it gives claimed dependencies of the algorithm time and space costs on $\nu$.

The proposed algorithm can be further extended for the case, where segments of $E$ have disjoint relative interiors.

3.3.2. Special segment configurations

More accurate approximation algorithms can be built for special configurations of straight line segments. In [14] it is shown that the IPGD problem is NP- and in fact W[1]-hard for the case where $E$ is edge set of a Delaunay triangulation (or of a Gabriel graph) which often arise in network applications. Theorems below report fast constant factor approximations for all these special segment configurations.

**Theorem 2.** For any small $\nu > 0$ the following statements hold true:

1. if $G$ is an outerplane graph, there is a $\left(34 + 24 \sqrt{2} + \nu\right)$-approximation;
2. if each pair of distinct segments from $E$ is at Euclidean distance either zero or more than $r$ from each other, there is a $\left(12 + 6 \sqrt{3} + \nu\right)$-approximation;
3. there is a $\left(34 + 44 \sqrt{\frac{6}{11}} + \nu\right)$-approximation if $G$ is any subgraph of a Delaunay triangulation; in addition, for outerplane $G$ an $\left(24 + 28 \sqrt{\frac{6}{11}} + \nu\right)$-approximation exists.

All these approximations work in $O\left(n^2 + \frac{n \log n}{\nu^2} + \frac{\log n}{\nu^3}\right)$ time and $O\left(n^2 \log n\right)$ space.

The theorem is proved in the analogous way using the corollary 1, lemmas 1, 2, 3, 4, 8, 11 and 12.
Theorem 3. If $G$ is any subgraph of a Gabriel graph, there is a 18-approximation algorithm for the IPGD problem. Its time and space complexities are of the order $O(n^2 \log n)$ and $O(n^2)$ respectively.

Proof. Due to the lemma 10 one has a constant time hitting set finder for sets $N_{c,I}$ with $c_1 = 0$ and $c_2 = 18$. It gives $\theta_0 = 0$ in the Epsilon Net Finder subalgorithm according to the lemma 2. To design an approximation algorithm for the case where $E$ is subset of edge set of a Gabriel graph, one can apply the Epsilon Net Finder subalgorithm directly (i.e. not as a subalgorithm within the Parametric Agarwal-Pan algorithm) for $\delta = \epsilon = 0$ and $w \equiv 1$, resulting in a maximal independent set $I$ within $N_r(E)$. A set $U_0(I)$ can be used as a hitting set for $N_I, I \in I$, whose construction is described just before the lemma 10. Of course, this algorithm is 18-approximate as $|I| \leq \text{OPT}$.

To establish complexity bounds for constructed algorithm, its implementation details are given below. At first it could be checked whether $N \cap N' = \emptyset$ for every distinct $N,N' \in N_r(E)$ in $O(n^2)$ time and keep this information in the form of an adjacency list $L$ of the respective intersection graph with ordered vertex set. The latter takes $O(n^2 \log n)$ time and $O(n^2)$ space. Initially, let $P := N_r(E)$ and $I := \emptyset$. Then the following steps should be taken until $P = \emptyset$.

Some object $I \in P$ is chosen setting $I := I \cup \{I\}, P_I := \{N \in P : N \cap I \neq \emptyset\}$ and $P := P \setminus P_I$. The list $L$ is updated by excluding those its records which contain information about objects from $P_I$.

Obviously, total complexity is of the order

$$O \left( n \log n \sum_{I \in \mathcal{I}} |P_I| \right) = O(n^2 \log n)$$

of these steps.

4. Conclusion

Constant factor approximations are proposed for the special NP- and W[1]-hard geometric Hitting Set problem on a set of Euclidean $r$-neighbourhoods of straight line segments, where segments are allowed to intersect only at their endpoints. More accurate approximations are also provided for special configurations of segments, forming edge sets of outerplane and some of proximity graphs. They demonstrate competitive combination of approximation factor and time complexity being compared with known local search and epsilon net based approximation algorithms for similar Hitting Set problems on sets of pseudo-disks.

We believe that our approximations can be expedited by incorporating clever geometric data structures. Moreover, some extensions are also possible for more general Partial Hitting Set problems on Euclidean $r$-neighbourhoods of segments due to recent results from [13].
References

[1] Agarwal, P.K., Efrat, A., Ganjugunte, S.K., Hay, D., Sankararaman, S., Zussman, G.: The resilience of WDM networks to probabilistic geographical failures, IEEE/ACM Trans. Netw., 21(5), 1525–1538 (2013).

[2] Agarwal, P., Pan, J.: Near-Linear Algorithms for Geometric Hitting Sets and Set Covers, Proc. of 30th Ann. Symp. on Comput. Geom., 271-279 (2014)

[3] Alon, N.: A Non-linear Lower Bound for Planar Epsilon-nets, Discr. and Comput. Geom., 47(2), 235-244 (2012)

[4] Antunes, D., Mathieu, C., Mustafa, N.: Combinatorics of Local Search: An Optimal 4-Local Hall’s Theorem for Planar Graphs, Proc. of 25th Ann. Eur. Symp. on Alg. (ESA), 87, 8:1-8:13 (2017)

[5] Basappa, M.: Line Segment Disk Cover, Lecture Notes in Computer Science, 10743, 81-92 (2018)

[6] Biniaz, A., Liu, P., Maheshwari, A., Smid, M.: Approximation algorithms for the unit disk cover problem in 2D and 3D, Comput. Geom., 60, 8-18 (2017)

[7] Blelloch, G.: Space-efficient dynamic orthogonal point location, segment intersection, and range reporting, Proc. of 19th Symp. on Discr. Alg., 894903 (2008)

[8] Brevilliers, M., Chevallier, N., Schmitt, D.: Triangulations of Line Segment Sets in the Plane, Lecture Notes in Computer Science, 4855, 388-399 (2007)

[9] Bus, N., Garg, S., Mustafa, N., Ray, S.: Limits of Local Search: Quality and Efficiency, Discr. and Comput. Geom., 57(3), 607-624 (2017)

[10] Bus, N., Mustafa, N., Ray, S.: Practical and efficient algorithms for the geometric hitting set problem, Discr. Appl. Math., (2018)

[11] Cormen, T., Leiserson, C., Rivest, R., Stein, C.: Introduction to Algorithms, Third Edition, 1292. The MIT Press, Cambridge, Massachusetts London, England (2009)

[12] Dash, D., Bishnu, A., Gupta, A., Nandy, S.C.: Approximation algorithms for deployment of sensors for line segment coverage in wireless sensor networks, Wireless Networks, 19(5), 857-870 (2013)

[13] Inamdar, T., Varadarajan, K.: On Partial Covering For Geometric Set Systems, https://arxiv.org/abs/1711.04882 (2017)

[14] Kobylkin, K.: Stabbing Line Segments with Disks: Complexity and Approximation Algorithms, Lect. Notes in Comp. Sci., 10716, 356-367 (2017)
[15] Komlos, J., Pach, J., Woeginger, G.: Almost tight bounds for \( \varepsilon \)-Nets, Discr. and Comput. Geom., 7(2), 163-173 (1992)

[16] Kulkarni, J., Govindarajan, S.: New \( \varepsilon \)-Net Constructions, Proceedings of Canadian Conference on Computational Geometry, 159-162 (2010)

[17] Madireddy, R.R., Mudgal, A.: Stabbing line segments with disks and related problems, Proc. of 28th Canadian Conference on Computational Geometry, 201-207 (2016)

[18] Mustafa, N., Dutta, K., Ghosh, A.: A simple proof of optimal epsilon nets, Combinatorica, 1-9 (2017)

[19] Mustafa, N., Ray, S.: Improved Results on Geometric Hitting Set Problems, Discr. and Comput. Geom., 44(4), 883-895 (2010)

[20] Pyrga, E., Ray, S.: New Existence Proofs for \( \varepsilon \)-nets, Proc. of 24th Ann. Symp. on Comput. Geom., 199-207 (2008)

Appendix A. Parametric Agarwal-Pan algorithm

**Iterative reweighting.**

**Input:** a parameter \( k \) and a range space \((Y, R)\);

**Output:** if there exists a \( k \)-element hitting set for \((Y, R)\), it returns true; otherwise, it may give either true or false. Along with the true value it also returns \( w_0 := w \) and \( \varepsilon_0 := \frac{1}{\lambda k e^{\lambda\varepsilon^2/\mu k}} \).

1. set \( t := 1 \); // round counter
2. set \( w \equiv 1 \) and assume \( R = \{R_1, \ldots, R_m\} \);
3. set \( s := 0 \) and \( p := 1 \); // counters for weight updates and sets processed
4. verify the inequality
   \[
   w(R_p) \leq \frac{w(Y)}{\lambda k}
   \]
   and if it is true, set \( s := s + 1 \), multiply weights of points from \( R_p \cap Y \) by \( 1 + \lambda_1 \) and continue repeating step 4 while \( s < \lceil \mu k \rceil \) and (A.1) still holds;
5. for \( s < \lceil \mu k \rceil \) examine whether the equality \( p = m \) holds: if it does, return \( w_0 := w, \varepsilon_0 := \frac{1}{k e^{\lambda\varepsilon^2/\mu k}} \) and true; otherwise, set \( p := p + 1 \) and go to step 4;
6. if \( s = \lceil \mu k \rceil \), check if \( t > \frac{2\lambda \ln(|Y|/k)}{\mu \lambda_1 \kappa} \) holds: when it does, return false; otherwise, set \( t := t + 1 \) and go to step 3.

**Parametric Agarwal-Pan algorithm.**

**Input:** a range space \((Y, R)\) and parameters \( \delta, \mu, \lambda_1 \) and \( \eta \) with \( \kappa > 0 \);

**Output:** a \( C(\delta + \lambda e^{\lambda \mu \kappa/\lambda}) \)-approximate hitting set \( H \subset Y \) for \( R \);

1. set \( k := 1, \lambda := 1 + \eta \) and create range counting and reporting data structures for space \((Y, R)\);
2. compute \( \frac{1}{\lambda} \)-net \( H_\delta \) for \((Y, R)\);
3. set \( Y_\delta := Y \setminus H_\delta, R_\delta := R \setminus \{R \in R : R \cap H_\delta \neq \emptyset\} \);
4. if the Iterative reweighting subalgorithm outputs false for the range space \((Y_\delta, R_\delta)\) and current value of \(k\), set \(k := 2k\) and go to step 2;
5. otherwise, set \(H_1 := H_\delta\), get \(k \leq \text{OPT}(Y_\delta, R_\delta)\) by doing a binary search within the interval \([k/2, k]\) using the Iterative reweighting subalgorithm, where this subalgorithm should return true for the last tried \(k\);
6. set \(k_\delta := k\), \(\varepsilon_\delta := \varepsilon_0\), \(w_\delta := w_0\), where \(w_0\) and \(\varepsilon_0\) are output by the Iterative reweighting subalgorithm for the last tried \(k\);
7. compute \(\varepsilon_\delta\)-net \(H_2\) for \((Y_\delta, R_\delta, w_\delta)\) and return \(H := H_1 \cup H_2\).

### Appendix B. Proof of the lemma

**Proof.** First, we are to estimate approximation ratio of the Parametric Agarwal-Pan algorithm. For a given \(k\) we prove that at most \(\frac{2k \ln |Y_\delta|}{\lambda_1 k}\) weight changes are done in the Iterative reweighting subalgorithm for sets from \(R_\delta\) at its step 4 summing over all rounds in the case where there is a \(k\)-element hitting set \(H_k \subseteq Y_\delta\) for \(R_\delta\). Indeed, let \(w_k^F(Y_\delta)\) (respectively, \(w_k^F(H_k)\)) be the weight \(w(Y_\delta)\) (respectively, be the weight \(w(H_k)\)) observed at the end of the round in which the Iterative reweighting subalgorithm finishes working.

Let \(z_k\) be also the total number of times that weights are updated (i.e. multiplied by \(1 + \lambda_1\)) of sets from \(R_\delta\). We note that

\[
w_k^F(Y_\delta) \leq |Y_\delta| \left(1 + \frac{\lambda_1}{\lambda k}\right)^{z_k}\quad \text{(B.1)}
\]

From the other hand, we have

\[
\frac{w_k^F(H_k)}{k} = \frac{\sum_{h \in H_k} (1 + \lambda_1)^{z_k(h)}}{k} \geq (1 + \lambda_1)^k \frac{\sum_{h \in H_k} z_k(h)/k}{k} \geq (1 + \lambda_1)^{z_k/k}
\]

where \(z_k(h)\) denotes the number of times that \(w(h)\) is updated. As \(w_k^F(H_k) \leq w_k^F(Y_\delta)\) we get the inequality

\[
k(1 + \lambda_1)^{z_k/k} \leq |Y_\delta| \left(1 + \frac{\lambda_1}{\lambda k}\right)^{z_k}.
\]

Resolving it with respect to \(z_k\), we get \(z_k \leq \frac{2k \ln |Y_\delta|/k}{\lambda_1 k}\). Thus, once we get \(\text{OPT}(Y_\delta, R_\delta) \leq k\), after at most \(\frac{2k \ln |Y_\delta|/k}{\lambda_1 k}\) weight updates the Iterative reweighting subalgorithm outputs true. Bisection method at step 5 of the Parametric Agarwal-Pan algorithm explores true and false responses of the Iterative reweighting subalgorithm to localize \(\text{OPT}(Y_\delta, R_\delta)\). It finally gets \(k_\delta \leq \text{OPT}(Y_\delta, R_\delta)\) with true response of the latter subalgorithm for \(k = k_\delta\).

Let \(H \subseteq Y\) be the set of size at most \(Ck \left(\delta + \lambda e^{\lambda e^{\lambda e^{\lambda e^{\lambda e^{\lambda e}}}}}ight)\), which is returned at step 7 of the Parametric Agarwal-Pan algorithm. We prove that \(H\) is a hitting set for \(R\). Let \(w_k^I(Y_\delta)\) be \(w(Y_\delta)\) observed at the beginning of the (final) round within which the true is returned by the Iterative reweighting
subalgorithm applied at step 5 of the Parametric Agarwal-Pan algorithm. As $s \leq \lceil \mu k \rceil - 1$ in that round, we have

$$w_k^f(Y_\delta) \leq (1 + \frac{\lambda_1}{\mu})^s w_k^f(Y_\delta) \leq e^{\lambda_1 s/(\mu k)} w_k^f(Y_\delta) \leq e^{\lambda_1 \mu/\lambda} w_k^f(Y_\delta).$$

From the other hand, $w_k^f(R) > \frac{w_k^f(Y_\delta)}{\lambda k} > \frac{w_k^f(Y_\delta)}{\lambda_1 k} \geq \frac{w_k^f(Y_\delta)}{\lambda_1 k} \geq w_k^f(R)$ for any $R \in \mathcal{R}_\delta$, where $w_k^f(R)$ denotes weight of $R$ at the end of the final round. Thus, the algorithm output $H$ gives a hitting set for $\mathcal{R}$ and:

$$|H| \leq C k \left( \delta + \lambda e^{\lambda_1 s/(\mu k)} \right) \leq C \left( \delta + \lambda e^{\lambda_1 s/(\mu k)} \right) \text{OPT}(|Y_\delta|, \mathcal{R}_\delta) \leq C \left( \delta + \lambda e^{\lambda_1 s/(\mu k)} \right) \text{OPT}.$$

Now we are to establish bounds for time complexity of the Parametric Agarwal-Pan algorithm. Its step 1 requires $O(\tau^{(0)})$ time according to our assumptions. Binary search at steps 2-4 requires $O(T)$ time, whereas step 7 takes $O(T)$ time using, e.g. ordered set data structures [11]. Then, binary search of step 5 requires $O(T \log \text{OPT}(Y_\delta, \mathcal{R}_\delta))$ time whereas step 7 takes $O(\varphi_{\lambda e^{\mu/\lambda}})$ time.

Thus, it remains for us to estimate $T$. Recall that the Iterative Reweighting subalgorithm call is for space $(Y_\delta, \mathcal{R}_\delta)$. For any $t$th round consists of at most $|\mathcal{R}_\delta|$ range counting operations to compute weights of sets from $\mathcal{R}_\delta$. As $t \leq \frac{2\lambda \ln(|Y|/k)}{\mu k}$, overall time complexity is of the order $O\left( \frac{2\lambda \ln(|Y|/k)}{\mu k} \right)$ for such operations at step 4. We have $|R \cap Y| \leq \frac{|Y|}{\mu k}$ for every $R \in \mathcal{R}_\delta$. As $s \leq \lceil \mu k \rceil$, we get

$$O \left( \frac{2\lambda \left( \text{OPT}(Y_\delta, \mathcal{R}_\delta) + \frac{1}{\mu} \right) \omega \ln |Y|}{\lambda_1 k} + \frac{2\lambda |Y| \left( 1 + \frac{1}{\mu} \right) \tau_2 \ln |Y|}{\delta \lambda_1 k} \right)$$

time complexity for range counting operations to update point weights as well as for range reporting operations at step 4. Therefore we have $T = O \left( \frac{2\lambda |Y| \omega}{\lambda_1 k} \right)$.

As for space cost of the algorithm, we note that

$$w(Y_\delta) \leq |Y_\delta| e^{2 \ln(\frac{2\lambda |Y|}{\mu k}) + 1/\mu}$$

substituting $z_k = \left( \frac{2\lambda \ln(|Y|/k)}{\mu k} + 1 \right) (\mu k + 1)$ into the bound [11] for $w^f_k(Y_\delta)$. Of course, as $w(y) = (1 + \lambda_1)^{z_k}(y)$ for $y \in Y_\delta$, it can be shown that

$$\sum_{y \in Y_\delta} \ln w(y) \leq |Y_\delta| \ln \frac{|Y_\delta| \ln |Y_\delta|}{|Y_\delta|} = O \left( \frac{|Y_\delta| \ln |Y_\delta|}{\mu k} \right).$$

29
Appendix C. Proof of the lemma

PROOF. First, we summarize on the complexity of step 1 of the Epsilon Net Finder subalgorithm. It can be implemented in the straightforward manner as follows. Initially, set $\mathcal{P} := \mathcal{R}_\varepsilon$ and $\mathcal{I} := \varnothing$. An arbitrary set $P \in \mathcal{P}$ is tried for adding to $\mathcal{I}$ by performing a sequence of checks to find out if there is an object $I \in \mathcal{I}$ with $w(P \cap I) > \delta w(Y)$. If such $I$ exists, then choose a single one, add $P$ to a set $\mathcal{R}_{\varepsilon,I}$ which is initially assumed empty and set $P := P \setminus \{P\}$. Otherwise, add $P$ to $\mathcal{I}$ and set $P := P \setminus \{P\}$. We stop when $\mathcal{P} = \varnothing$.

Let $t_{\theta_0} = |\mathcal{I}|$ and $z_i$ be the number of sets from $\mathcal{R}$ which are tried for inclusion to $\mathcal{I}$ when $|\mathcal{I}| = i$. Then, time complexity of our straightforward implementation is of the order:

$$O\left(\psi t_{\theta_0} \sum_{i=1}^{t_{\theta_0}} z_i + \gamma\right) = O\left(\psi t_{\theta_0} \sum_{i=1}^{t_{\theta_0}} z_i + \gamma\right)$$

as the data structure has query time $\psi$. Its space cost is obviously of the same order as required to store the range space $(Y, \mathcal{R}, w)$.

Second, we are to estimate complexity of step 2 of the Epsilon Net Finder subalgorithm. Here we have disjoint sets $\mathcal{R}_{\varepsilon,I}$ formed for each $I \in \mathcal{I}$. Thus, applying the auxiliary procedure requires

$$O\left(\sum_{I \in \mathcal{I}} \xi(|Y \cap I|, |\mathcal{R}_{\varepsilon,I}|)\right) = O\left(\xi(|Y|, |\mathcal{R}_\varepsilon|)\right)$$

time due to the assumption 1.

Finally, we establish claimed upper bound for length of epsilon net produced by the Epsilon Net Finder subalgorithm. Let $\mathcal{I}$ be a maximal $\delta$-independent set, where the parameter $\theta_0$ is to be chosen later. Following the same argument as in the proof of the theorem 4 from [20], we get

$$t_{\theta_0} \leq \sum_{I \in \mathcal{I}} \frac{w(I)}{\varepsilon w(Y)} = \frac{\sum_{y \in Y \cap \bigcup_{I \in \mathcal{I}} I} w(y)m_I(y)}{\varepsilon w(Y)} \leq \frac{\sum_{y \in Y \cap \bigcup_{I \in \mathcal{I}} I} w(y)(m_I(y) + \tau)}{\varepsilon w(Y)} = \frac{\tau w(Y \cap \bigcup_{I \in \mathcal{I}} I) + \sum_{u \in U} w(Y(u))}{\varepsilon w(Y)} \leq \frac{\tau w(Y) + t_{\theta_0} \beta \delta w(Y)}{\varepsilon w(Y)} = \frac{t_{\theta_0} \beta \theta_0}{\alpha} + \frac{\tau}{\alpha \varepsilon}$$

where $Y(u) \subset Y$ contains all points which lie in the intersection of a pair of those objects from $\mathcal{I}$ which form an edge $u \in U$. Thus, it gives upper bounds $t_{\theta_0} \leq \frac{\tau}{(\alpha - t_{\theta_0} \beta \varepsilon)}$ and $\frac{\sum_{I \in \mathcal{I}} w(I)}{\varepsilon w(Y)} \leq \frac{\tau}{(\alpha - t_{\theta_0} \beta \varepsilon)}$. According to our assumptions, the
auxiliary procedure gives a hitting set of size at most $c_1 w(I) + c_2$ for $R_{\varepsilon, I}, I \in \mathcal{I}$. Therefore, $H_{\theta_0}$ is an $\varepsilon$-net of size at most $(c_1 \theta_0 + c_2) \tau (\alpha - \theta_0 \beta) \varepsilon$. Optimizing with respect to $\theta_0 < \alpha \beta$ we obtain $\theta_0^* = \frac{\alpha \beta}{1 + \sqrt{1 + c_2 \alpha c_1 \beta}}$ and get the claimed bound

$$|H_{\theta_0^*}| \leq \left(1 + \frac{1}{\sqrt{1 + \frac{c_2 \alpha}{c_1 \beta}}}\right) \left(\frac{2 c_1 \tau \beta}{\alpha^2} + \frac{c_2 \tau}{\alpha} + \frac{c_2 \tau}{\alpha \sqrt{1 + \frac{c_2 \alpha}{c_1 \beta}}}\right) \frac{1}{\varepsilon}.$$

Appendix D. Proof of the lemma 5

**Proof.** Let $\mathcal{J} = \{J_i\}_{i=1}^n$ be a set of bounded intervals on the real line, i.e. $J_i = [a_i, b_i], i = 1, \ldots, n$. Let $\mathcal{J}'$ be its subset of intervals which is maximal with respect to inclusion and does not contain pairs of intervals $I$ and $J$ with either $I \subseteq J$ or $J \subset I$. Removing such pairs from $\mathcal{J}$ can be done in $O(n \log n)$ time and $O(n)$ space. Indeed, an interval $[a, b]$ can be represented by a point $(a, b)$ on the $xy$-plane above the straight line $y = x$; checking if an interval $[a, b]$ contains some other interval $[c, d]$ is equivalent to checking if the axis-parallel rectangle contains a point $(c, d)$ whose left upper vertex is $(a, b)$ and right lower vertex is $(b, a)$. This check can be done using data structures for processing of orthogonal range emptiness queries on $n$-point sets in $O(\log n)$ time and $O(n)$ space with preliminary preprocessing in $O(n \log n)$ time [7].

Secondly, we get lower and upper ends of intervals from $\mathcal{J}'$ sorted in a single sequence, set $H := \emptyset$ and $\mathcal{P} := \mathcal{J}'$. Then, doing sequentially until $\mathcal{P} = \emptyset$, an interval $I_k = [a_k, b_k] \in \mathcal{P}$ is selected at step $k$ with the maximal upper end $b_k$; its lower end $a_k$ is added to the hitting set $H$ and intervals are excluded from $\mathcal{P}$ which are hit by $a_k$. When $\mathcal{P} = \emptyset$, let $Q = \{I_k\} \subset \mathcal{J}$ be the set of non-overlapping intervals, thus, constructed. We get that $H$ is the minimum cardinality hitting set for $\mathcal{J}$.

Summarizing on the complexity of computing of $H$, we note that sorting of interval ends from $\mathcal{J}'$ can obviously be done in $O(|\mathcal{J}'| \log |\mathcal{J}'|)$ time. Moreover, when reporting those intervals from $\mathcal{P}$, which contain $a_k$, we first start with the interval from $\mathcal{P}$ having the second maximal upper end. Thus, it takes $O(|\mathcal{J}'|)$ overall time for reporting such intervals.