PTAS for Stabbing Unit Squares and Variants

Tanmay Inamdar* Sounak Modak† Kushal Singanporia†

Abstract

In this article, we consider the problem of finding a minimum number of horizontal segments, such that each of the given rectangles is *stabbed* by at least one chosen segment (i.e., the segment intersects both vertical sides of the rectangle). This problem was first studied by [Chan et al., ISAAC 2018], who gave an $O(\log\log n)$-approximation for the problem, by observing that it can be modeled as a special case of a geometric hitting set problem in $\mathbb{R}^3$, which was known to admit an $O(\log\log n)$-approximation. To the best of our knowledge, no subsequent improvement over this result is known. In this work, we design a polynomial-time approximation scheme (PTAS) for the problem, in the special case when the given rectangles are in fact disjoint unit squares, and the horizontal segments have a bounded length. We also design a PTAS for the Maximum Coverage variant of the setting, where the goal is to stab the maximum number of squares using at most k horizontal segments. In spite of multiple restrictions imposed on the input, our results constitute as the first step towards improving the $O(\log\log n)$-approximation for stabbing arbitrary rectangles by arbitrary horizontal segments. Finally, we sketch an extension of our PTASes to the setting where the bounded-length segments can be vertical as well as horizontal. We note that this complements a known APX-hardness result when the segments can have unbounded length.

1 Introduction

In the Set Cover problem, the input consists of a *set system* $(\mathcal{U}, \mathcal{F})$, where $\mathcal{U}$ is a finite universe of size n, and $\mathcal{F}$ is a family of subsets of $\mathcal{U}$, and the aim is to find a minimum-size sub-family $\mathcal{F}' \subseteq \mathcal{F}$ whose union covers the universe. Due to its wide applicability, Set Cover is perhaps the most important combinatorial optimization problem. In the seminal work of Karp [12], it was shown to be NP-hard. Faced with this intractability, the problem has received significant attention in the field of polynomial-time approximation algorithms, where the goal is to design efficient algorithms that provably output solutions within a certain guaranteed factor from an optimal solution. In this regime, it is well-known that a greedy algorithm achieves an approximation factor of $\ln n+1$ [16], and moreover this factor is essentially tight, assuming $\mathsf{P} \neq \mathsf{NP}$ [5].

However, this is not the end of the research for approximability of Set Cover. In fact, there are a large number of set systems whose structure allows us to overcome the lower bound of $\Omega(\log n)$, and achieve near-constant approximation factors, e.g., [1, 2, 4, 7, 10, 15, 17, 18]. One of the most prominent classes of examples that have shown remarkable success along this line comes from computational geometry. Informally speaking, Geometric Set Cover refers to an instance of Set Cover, where the set system has an underlying geometric realization. A classic example is when each element of the universe corresponds to a point, say in $\mathbb{R}^2$, and each set in the family corresponds to the subset of points contained in a certain kind of geometric object, e.g., unit disk. In the "dual" setting, the roles of points and geometric objects are reversed – the goal is to find the smallest set of points such that each geometric object contains at least one of the selected points. This setting is known as *piercing* or *hitting* of geometric objects by points. Although the exact terminology is different based on the roles of geometric objects playing the roles of elements and sets, they all fall in the broad umbrella of Geometric Set Cover.

Motivated from applications in resource allocation and scheduling, the special case of Geometric Set Cover, where the goal is to find the minimum number of lines/segments in the plane that *cross* all of the given rectangles has been extensively studied in both approximation algorithms and parameterized complexity. In this setting, the rectangles as well as lines/segments are typically restricted to be axis-parallel, and a line/segment that intersects two parallel sides of a rectangle is said to *stab* the rectangle. There are different variants of rectangle stabbing problems, based on (1) whether the rectangles are disjoint or overlapping, (2) lines/segments are vertical, horizontal, or both, and (3) whether a set of lines/segments is explicitly given in the input, or the solution can contain any line/segment in the plane. Much of the early research in this area considered lines rather than segments, and the set of lines was not explicitly given in the input. Gaur et al. [9] designed a 2-approximation for stabbing arbitrary

*Indian Institute of Technology Jodhpur
taninamdar@gmail.com
Supported by IITJ Research Initiation Grant (grant number I/RIG/TNI/ 20240072
†Institute of Mathematical Sciences
sounakmodak@gmail.com, kushal03132@gmail.com

rectangles with (non-given) horizontal and vertical lines, which remains the best approximation for the problem to this day.

More recently, Chan et al. [3] initiated the study of stabbing with segments instead of lines. They considered different variants, including a variant where the goal is to minimize the total length of the horizontal segments to stab all the rectangles, for which they designed a constant-factor approximation, which was subsequently improved to a quasi-polynomial-time approximation scheme (QPTAS) in an unpublished work of Eisenbrand et al. [6], and later to a polynomial-time approximation scheme (PTAS) by Khan et al. [13][1]. Among the different variants of rectangle stabbing [11, 13] by segments introduced in [3], the most relevant to this work is the one called Cardinality Stabbing, where the set of horizontal segments is given in the input, and the goal is to find a minimum number of segments that stab all of the given rectangles. Chan et al. [3] showed that this variant is APX-hard, and gave an $O(\log \log n)$-approximation in polynomial time.

Our Results. The aforementioned approximation algorithm of Chan et al. [3] for Constrained Stabbing works even when the rectangles can be of arbitrary sizes and may intersect each other, and the input segments can have arbitrary length. To the best of our knowledge, prior to our work, it was not known whether one can improve upon the $O(\log \log n)$ factor by restricting the input. In this work, we consider the following special case of Constrained Stabbing.

Stabbing Disjoint Unit Squares

Input: An instance $\mathcal{I} = (\mathcal{R}, \mathcal{S})$, where $\mathcal{R}$ is a set of n axis-parallel, disjoint unit squares and $\mathcal{S}$ is a set of m horizontal segments, each of length at most d.

Question: Find minimum-size $\mathcal{S}' \subseteq \mathcal{S}$ such that $\mathcal{S}'$ stabs all n squares of $\mathcal{R}$.

Note that, this is a considerable restriction of the original setting, since a segment of length at most d can stab at most d disjoint unit squares. Therefore, the classical greedy algorithm for Set Cover already gives a $(\ln d + 1)$-approximation in our setting. Our contribution here is to improve the approximation ratio to $1+\epsilon$. More formally, our main result is the following.

Theorem 1 *For any $\epsilon > 0$, there exists an algorithm that takes an instance of* Stabbing Disjoint Unit Squares*, runs in time $m^{O(d \log d/\epsilon^2)} \cdot n^{O(1)}$ and returns a $(1+\epsilon)$-approximation.*

[1] A QPTAS is an $(1 \pm \epsilon)$-approximation algorithm that runs in $2^{(\log n)^c f(1/\epsilon)}$ for any fixed $\epsilon > 0$, where $c \geq 1$ is a constant. A PTAS is a special case where $c = 1$, which notably runs in polynomial time.

At a high level, this result is based on the classical *shifting strategy* introduced by Hochbaum and Maass [10]. At a high level, we first divide the given instance into a set of vertical sub-instances of width $O(d/\epsilon)$, via a straightforward application of shifting strategy. Then, we further horizontally sub-divide each vertical sub-instance into even smaller sub-instances whose optimal solution is bounded by $O(d \log d/\epsilon^2)$, enabling us to efficiently find it via exhaustive enumeration. The second step in our approach is inspired from a PTAS by Eisenbrand et al. [6] for minimizing the total length of the segments. We note that, although we follow a similar roadmap, we have to handle several nuances due to the significant differences between the two problems. In Section 3, we prove Theorem 1 by formally stating the algorithm and its analysis. Subsequently, we also sketch an approach that can extend to the setting where the given segments of length at most d can be vertical as well as horizontal.

Next, we consider the "Max Coverage" variant of Stabbing Disjoint Unit Squares, defined as follows.

Maximum Stabbing of Disjoint Unit Squares

Input: An instance $\mathcal{I} = (\mathcal{R}, \mathcal{S}, k)$, where $\mathcal{R}$ is a set of n axis-parallel, disjoint unit squares, $\mathcal{S}$ is a set of m horizontal segments, each of length at most d, and k is a non-negative integer.

Question: Find a subset $\mathcal{S}' \subseteq \mathcal{S}$ of size at most k, such that $\mathcal{S}'$ stabs the maximum number of squares from $\mathcal{R}$.

For this version, we design a PTAS using a combination of the shifting strategy and dynamic programming, which is inspired from the work of Gandhi et al. [8]. More formally, we prove the following theorem in Section 4.

Theorem 2 *For any $\epsilon > 0$, there exists an algorithm that takes an instance of* Maximum Stabbing of Disjoint Unit Squares*, runs in time $n^{O(d/\epsilon^2)}$ and returns a $(1+\epsilon)$-approximation.*

We will also sketch an adaptation of this algorithm to handle vertical segments.

Finally, it will be apparent from our proof that both of our algorithms can also be adapted to the setting where the input consists of disjoint rectangles, where the length and breadth of rectangles are $\Theta(1)$. Due to space constraints, some of the proofs are deferred to the appendix.

2 Preliminaries

We assume that squares are closed sets, i.e., their boundaries are included. We assume, without loss of generality, that no segment or square boundary lies at an integer y-coordinate; this can be ensured via an appropriately shifting the origin. For expressions such as $\frac{d}{\epsilon}$ that may not be integers, we assume throughout that they are rounded up using the ceiling function, i.e., $\lceil \frac{d}{\epsilon} \rceil$, wherever necessary. We note that the total number of segments m can be reduced to $O(n^3)$ by restricting attention to segments aligned with specific x and y coordinates corresponding to square boundaries. Each segment can be shifted vertically to lie in one of $O(n)$ "equivalence classes" given by the top/bottom y-coorindates of any of the n squares, or the intervals between them. Next, we restrict the left and right endpoints of each segment such that they align with the left/right boundary of a square. Note that these operations can be performed without changing the set of squares stabbed by each segment. This bounds the number of segments by $O(n^3)$.

3 A PTAS for Stabbing Disjoint Unit Squares

Throughout this section, we fix an optimal solution $\mathcal{O} \subseteq \mathcal{S}$ of size $\mathrm{OPT}(\mathcal{I})$.

3.1 Algorithm Overview

Our $(1+\epsilon)$-approximation algorithm proceeds as follows, with each step justified by structural lemmas.

- Select a suitable vertical offset z and remove the rectangles intersected by the corresponding vertical lines $\mathcal{L}_z$. *(Lemma 3)*
- Stab the removed rectangles using a segment set $\mathcal{S}_E \subseteq \mathcal{S}$ of cost $O(\epsilon \cdot \mathrm{OPT}(\mathcal{I}))$. *(Lemma 3)*
- In each strip, perform a horizontal sweep to select a set S_h of segments that induce sub-instance boundaries. *(Lemma 5)*
- Solve each sub-instance via brute-force over small segment subsets.
- The total solution cost remains bounded by $(1 + c'\epsilon) \cdot \mathrm{OPT}(\mathcal{I})$. *(Lemma 4)*

Now, we proceed to a formal description of each of the steps. We introduce a family of vertical lines to help partition the instance into simpler parts. For any integer offset $z \in [0, \frac{d}{\epsilon})$, let $\mathcal{L}_z$ denote the set of vertical lines with x-coordinates given by $z + i \cdot \frac{d}{\epsilon}$ for $i \in \mathbb{Z}$. The offset z determines the horizontal shift of this regularly spaced grid of lines across the instance. In the analysis that follows, we use a probabilistic argument to show that some value of z yields favorable partitioning properties.

The goal of introducing $\mathcal{L}_z$ is to partition the set of rectangles into vertical regions such that each optimal stabbing segment interacts with only a limited portion of the input. This decomposition localizes the problem and allows us to bound the number of segments needed within each region, enabling more efficient solutions.

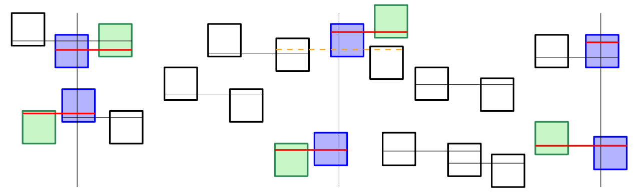

Figure 1: Squares intersecting the lines of $\mathcal{L}_z$ are shown as shaded blue squares. For each such square, we select an arbitrary segment that stabs it (shown in red). These segments, along with other rectangles they stab—even those contained strictly between the lines of $\mathcal{L}_z$ (shown as shaded green)—are not part of any of the sub-instances. However, the remaining segments (e.g., dashed orange segment) that cross the lines of $\mathcal{L}_z$ will be part of both sub-instances.

For a given value of z, let E_z denote the subset of squares in $\mathcal{R}$ that are intersected (stabbed) by the vertical lines of $\mathcal{L}_z$. We now claim the following.

Lemma 3 [Good-z Value] *There exists a value $0 \leq z < \frac{d}{\epsilon}$, satisfying that the number of squares of $\mathcal{O}$ that intersect any line of $\mathcal{L}_z$ is at most $\epsilon \cdot \mathrm{OPT}(\mathcal{I})$. Furthermore, for such a z, we can find in polynomial time a subset $\mathcal{S}_E$ of line segments from $\mathcal{S}$ that stabs all square in E_z, such that the total number of segments in $\mathcal{S}_E$ is at most $\epsilon \cdot \mathrm{OPT}(\mathcal{I})$.*

Since we have shown that a suitable value of z exists, we can evaluate all z values in the range $[0, d/\epsilon)$. From this point onward, we assume that such a z has been fixed, and we work with the corresponding set of vertical lines $\mathcal{L}_z$. After removing the squares in E_z and the segments used to stab them, some segments may remain that intersect vertical lines in $\mathcal{L}_z$ and stab squares from two adjacent vertical strips. Such segments may appear in more than one sub-instance and can be double-counted in the analysis. We also remove any additional squares that are stabbed by the segments used for E_z, even if those squares are not themselves in E_z. We now show that this potential overcount is bounded, and that the total cost over all sub-instances remains within a small fraction of $\mathrm{OPT}(\mathcal{I})$.

To formalize this, we define the vertical sub-instances that remain after the removal step described above. Let $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_q$ denote these sub-instances for some $q > 0$, where each $\mathcal{I}_j = (\mathcal{R}_j, \mathcal{S}_j)$ is defined as follows: $\mathcal{R}_j$ is the set of squares fully contained within the j-th vertical strip, and $\mathcal{S}_j$ is the set of segments that lie fully or partially within that strip.

Lemma 4 [Crossing Segments] *The total optimal cost across all vertical sub-instances satisfies:* $\sum_{j=1}^{q} \mathrm{OPT}(\mathcal{I}_j) \leq (1 + c'\epsilon) \cdot \mathrm{OPT}(\mathcal{I})$, *for some constant* $c' > 0$.

We consider a specific sub-instance $\mathcal{I}_j = (\mathcal{R}_j, \mathcal{S}_j)$, and in the next lemma, we describe how to sub-divide it further that will ultimately help us obtain a $(1+\epsilon)$-approximation for the sub-instance.

Lemma 5 (Strip Partitioning and S_h Bound) *There exists a set of horizontal segments $S_h \subseteq \mathcal{S}$ of size at most $\epsilon \cdot \mathsf{OPT}(\mathcal{I}_j)$, computable in polynomial time, that stabs a subset of $\mathcal{R}$ and partitions the remaining squares within each strip into disjoint rectangular sub-instances.*

Proof. Initially, all segments and squares are *unmarked*. We process the segments according to their y-coordinates in increasing order. At each y-coordinate corresponding to a segment s, we run the $(\ln d+1)$-factor approximation algorithm for the set cover instance given by all unmarked segments whose y-coordinate is at most that of s, and all unmarked squares stabbed by such segments. Note here that since each segment can stab at most d unit squares, greedy algorithm gives the desired guarantee. The process stops at a segment s^*, when the approximation algorithm on the corresponding sub-instance returns an output of size at least $d(\ln d+1)/\epsilon^2$. At this point, (1) we add all unmarked segments whose y-coordinate is equal to that of s^* to S_h, (3) create a sub-instance $\mathcal{I}_{j,1}$, consisting of all unmarked squares and unmarked segments inside the vertical strip below the y-coordinate of s^*, and finally (3) mark all the squares and segments that were part of the last call to the approximation algorithm corresponding to s^*.

Note that for the segment s' just before s^*, the size of the approximate solution was strictly smaller than $d(\ln d+1)/\epsilon^2$. Now, observe that the set of squares that are newly introduced in the sub-instance corresponding to s^* are those unmarked squares that are only stabbed by segments at the same y-coordinate as s^*. It is easy to see that the number of such squares is bounded by d/ϵ. Therefore, the size of optimal solution for the sub-instance is at least d/ϵ^2, and at most $d(\ln d+1)/\epsilon^2 + d/\epsilon = O(d \log d/\epsilon^2)$.

Now, we continue this procedure to create the sub-instances $\mathcal{I}_{j,2}, \ldots, \mathcal{I}_{j,t}$. Moreover, we have $\sum_{i=1}^{t} \mathrm{OPT}(\mathcal{I}_{j,i}) \leq \mathrm{OPT}(\mathcal{I}_j)$, since the sub-instances $\mathcal{I}_{j,i}$ are independent.

The overall algorithm runs in polynomial time since it iterates over polynomially many segments and invokes a polynomial-time approximation algorithm at each step. Since at each step, at most d/ϵ segments are added to S_h only after the size of optimal solution of the sub-instance below is at least d/ϵ^2, we can bound the number of added segments as follows:$|S_h| \leq \sum_{i=1}^{t-1} \epsilon \cdot \mathrm{OPT}(\mathcal{I}_{j,i}) \leq \epsilon \cdot \mathrm{OPT}(\mathcal{I}_j)$. □

Now we have all the ingredients required to prove Theorem 1. Due to space constraints, we give a formal proof in the appendix. Note that the running time is dominated by the time required to brute-froce in each of the smaller sub-instances, where the size of the optimal solution is bounded by $O(d \log d/\epsilon)$.

Handling horizontal as well as vertical segments. We now sketch the modifications required to extend our PTAS to work in the setting where the segments may be vertical as well as horizontal. First, we increase the range of z values to $[0, \frac{cd \ln d}{\epsilon})$ for some suitably large $c \geq 1$. Note that this also makes the horizontal distance between two consecutive lines of $\mathcal{L}_z$ to be $\Theta(\frac{d \log d}{\epsilon})$. We consider a vertical strip of width 2 centered around each line of $\mathcal{L}_z$, and say that a horizontal (resp. vertical) segment is *bad* if it crosses a line of $\mathcal{L}_z$ (resp. if it lies within the strip of width 2 around a line of $\mathcal{L}_z$). Note that each segment is bad w.r.t. at most $O(1)$ distinct values of z. Then, by using a probabilistic argument, one can show that, there exists a value of z such that at most $\frac{\epsilon}{\ln d+1}\mathrm{OPT}(\mathcal{I})$ segments (of both kinds) from a fixed optimal solution $\mathcal{O}$ are bad. We guess such a value of z, which we now fix. Now, consider all squares that intersect a line of $\mathcal{L}_z$, called E_z. We consider all such squares and all segments that can stab E_z, and use an $(\ln d+1)$-approximation to find a solution of size at most $\epsilon \cdot \mathrm{OPT}(\mathcal{I})$. Now, we decompose the instance into vertical sub-instances.

Now we further divide each vertical sub-instance $\mathcal{I}_j$ into smaller sub-instance using a modification to the procedure described in Lemma 5. To this end, we follow a similar procedure of considering segments by increasing y-value (for horizontal segments, this will be their y-value, for vertical segments, this will be the y-value of their lower endpoint), but set the threshold of size of approximate solution to be $d^2 \log d/\epsilon$. It follows that between two consecutive segments, at most d^2/ϵ new squares can be introduced. For each such square, we add a segment that stabs it to our solution. Note that due to the threshold, the number of segments thus added is at most ϵ times the optimal solution for the sub-instance below. This decomposes the vertical sub-instance further into smaller sub-instances, where the size of optimal solution is bounded by $O(d^2 \log d/\epsilon)$, which can be solved by brute force. This strategy leads to a PTAS with running time $n^{O(d^2 \log d/\epsilon)}$.

4 A PTAS for Maximum Stabbing of Disjoint Unit Squares

In this section, we present a PTAS for Maximum Stabbing of Disjoint Unit Squares. Our high-level strategy mirrors the earlier two-phase approach: we partition the instance horizontally and vertically into smaller sub-instances, allowing a small approximation loss in the objective value. For the sake of analysis, we fix an optimal solution $\mathcal{O}_k(\mathcal{I}) \subseteq \mathcal{S}$ of size k that stabs a subset $\mathcal{R}^* \subseteq \mathcal{R}$ of size t.

For an instance $\mathcal{I}' = (\mathcal{R}', \mathcal{S}')$ and an integer ℓ, let $\mathrm{OPT}_\ell(\mathcal{I}')$ denote the maximum number of squares in $\mathcal{R}$ that can be stabbed using any ℓ segments from $\mathcal{S}$. In particular, for the original instance $\mathcal{I}$ we set $\mathrm{OPT}_k := \mathrm{OPT}_k(\mathcal{I}) = |\mathcal{R}^*|$ for notational convenience.

Step 1. Partitioning the instance into vertical strips. We begin by defining a modified family of vertical lines, similar to the construction in the previous section. For $0 \le z < \frac{2d}{\epsilon}$, let $\mathcal{L}_z$ denote the set of vertical lines of the form $x = z + i \cdot \frac{2d}{\epsilon}$ for all integers $i \in \mathbb{Z}$.

A unit square s is said to be *h-bad* with respect to z if the maximum horizontal distance from any point in s to the nearest line in $\mathcal{L}_z$ is at most d; otherwise, we classify S as *h-good* with respect to z. Note that the term *h-bad* refers to horizontal distance, although it is measured from vertical lines—a distinction that should not be confused. We now establish the following lemma, which plays a role analogous to Lemma 3.

Lemma 6 *There exists a value $0 \le z < \frac{2d}{\epsilon}$ such that the number of squares in $\mathcal{R}^*$ that are h-bad with respect to z is at most $\epsilon \cdot \mathsf{OPT}_k$.*

By enumerating all possible values of z, we fix one that satisfies the guarantee of Lemma 6. This value of z remains fixed for the remainder of the discussion. Based on the vertical lines in $\mathcal{L}_z$, we define vertical sub-instances $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_p$, where each $\mathcal{I}_j = (\mathcal{R}_j, \mathcal{S}_j)$ consists of:

- $\mathcal{R}_j$: the set of squares that are *h-good* with respect to $\mathcal{L}_z$ and lie entirely within the region between the j-th and $(j+1)$-th vertical lines in $\mathcal{L}_z$, and
- $\mathcal{S}_j$: the set of segments that can stab at least one square in $\mathcal{R}_j$.

Step 2. Solving a sub-instance within a vertical strip. Fix a vertical sub-instance $\mathcal{I}_j = (\mathcal{R}_j, \mathcal{S}_j)$. We further decompose this sub-instance using a second application of the shifting technique—this time in the vertical direction, and in a simpler form.

For $0 \le z' < \frac{1}{\epsilon}$, let $\mathcal{L}'_{z'}$ denote the set of horizontal lines defined by $y = z' + i \cdot \frac{1}{\epsilon}$, for all integers $i \in \mathbb{Z}$. A square $S \in \mathcal{R}_j$ is said to be *v-bad* with respect to z' if it intersects any line in $\mathcal{L}'_{z'}$, and *v-good* otherwise. In general, we say that a square is *good* if it is both h-good as well as v-good, and *bad* otherwise. We now state the following lemma, which is analogous to Lemma 6. The proof follows directly from the same probabilistic argument and hence is omitted.

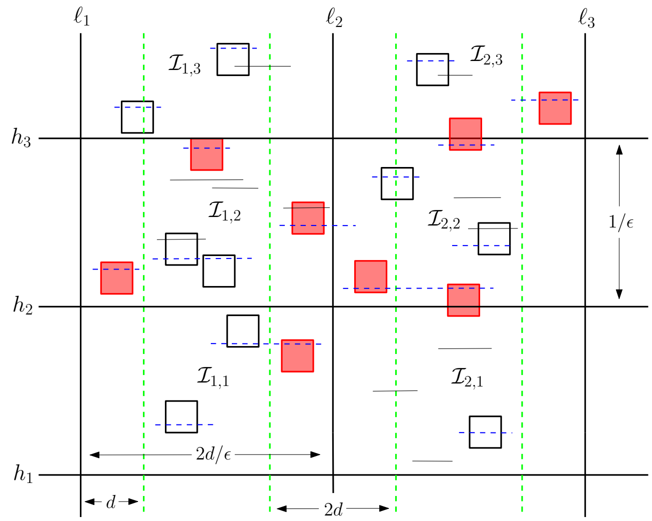

Figure 2: Squares colored red are bad squares, dotted segments shows optimal solution. Notice that our algorithm will not consider bad squares (Optimum included).

Lemma 7 *There exists some $0 \le z' < \frac{1}{\epsilon}$ such that at most $\epsilon \cdot \mathsf{OPT}_k$ squares in $\mathcal{R}^*$ are v-bad w.r.t. z'.*

We fix such a value of z' and decompose each vertical sub-instance $\mathcal{I}_i$ into smaller sub-instances $\mathcal{I}_{i,1}, \ldots, \mathcal{I}_{i,q}$. Each $\mathcal{I}_{i,j} = (\mathcal{R}_{i,j}, \mathcal{S}_{i,j})$ is defined as follows.

- $\mathcal{R}_{i,j}$ consists of only good squares from $\mathcal{R}_i$ that are fully contained within the region bounded by vertical lines i, $i+1$ and horizontal lines j, $j+1$,
- $\mathcal{S}_{i,j}$ is the set of segments from $\mathcal{S}_i$ that stab at least one square in $\mathcal{R}_{i,j}$.

Note that each sub-instance $\mathcal{I}_{i,j}$ is fully contained within a rectangle of dimensions $\frac{1}{\epsilon} \times \frac{2d}{\epsilon}$.

Lemma 8 *For each $i \ge 1$ and $j \ge 1$, let $\mathcal{I}_{i,j} = (\mathcal{R}_{i,j}, \mathcal{S}_{i,j})$ denote the j-th rectangular sub-instance of the i-th vertical sub-instance. Define $k_{i,j} := |\mathcal{S}_{i,j} \cap \mathcal{O}_k(\mathcal{I})|$ and $t_{i,j} := |\mathcal{R}_{i,j} \cap \mathcal{R}^*|$. Then:*

$$\sum_{i,j} \mathrm{OPT}_{k_{i,j}}(\mathcal{I}_{i,j}) \ge (1 - 2\epsilon) \cdot \mathrm{OPT}_k(\mathcal{I}).$$

Step 3: Solving Bounded Sub-Instances via Preprocessing

Each sub-instance $\mathcal{I}_{i,j} = (\mathcal{R}_{i,j}, \mathcal{S}_{i,j})$ lies within a rectangle of area at most $\frac{2d}{\epsilon^2}$. Since the squares are unit-sized and disjoint, the number of squares in any sub-instance is bounded by $\lambda := \frac{2d}{\epsilon^2}$. We similarly restrict attention to the subset of segments $\mathcal{S}_{i,j} \subseteq \mathcal{S}$ that intersect the same region. We exhaustively enumerate all subsets of segments from $\mathcal{S}_{i,j}$ of size at most λ and, for each, compute how many squares in $\mathcal{R}_{i,j}$ it can stab. Based on this, we construct a feasibility table: $T[i, j, s, \ell] = \texttt{true}$ iff there exists a subset of $\mathcal{S}_{i,j}$ of size at most ℓ that stabs at least s squares in $\mathcal{R}_{i,j}$.

- **Initialization:** $T[i, j, 0, \ell] := \texttt{true}$ for $0 \leq \ell \leq \lambda$
- **Monotonicity Rule:** If $T[i, j, s, \ell] = \texttt{true}$, then also set $T[i, j, s', \ell] := \texttt{true}$ for all $s' \leq s$.

This preprocessing requires time $m^{O(\lambda)}$ per sub-instance and produces a compact DP table indexed by (s, ℓ) pairs.

Step 4: Combining Sub-Instance Solutions under Global Budget

Let M denote the total number of sub-instances $\mathcal{I}_{i,j}$. To compute a global solution using at most k segments overall, we apply a dynamic programming routine across sub-instances.

Define a DP table: Let $DP[m, b]$ denote the maximum number of squares that can be stabbed using at most b segments across the first m sub-instances, here the sub-instances $\mathcal{I}_{i,j}$ are ordered arbitrarily. In the following, we slightly abuse the notation and in the table T, use m as a short-hand for i, j, where mth sub-instance is $\mathcal{I}_{i,j}$.

- **Initialization:** $DP[0, b] := 0$ for all $b \in [0, k]$
- **Transition:** For each sub-instance $m \in [1, M]$ and each budget $b \in [0, k]$,

$$DP[m, b] = \max_{0 \leq \ell \leq b} \Big(DP[m-1, b-\ell] + \max\{s \mid T[m, s, \ell] = \texttt{true}\} \Big)$$

- **Final Result:** $\max_{0 \leq b \leq k} DP[M, b]$ gives the best total coverage using $\leq k$ segments.

The above routine runs in $O(M \cdot k \cdot \lambda)$ time, assuming constant-time access to the precomputed T-table per sub-instance. Note that M, k, and λ are polynomial (in fact, linear) in m and n, which implies that the running time of the algorithm is dominated by the $n^{O(d/\epsilon^2)}$ time required to fill the $T[\cdot]$ table in each bounded-size sub-instance. This completes the proof of Theorem 2.

Handling vertical and horizontal segments. The modifications required to handle vertical segments are straightforward. In the first step, we remove all vertical segments that lie within a distance d from a line of $\mathcal{L}_z$. The rest of this step remains unaffected. In the second step, the vertical distance between lines of $\mathcal{L'}_{z'}$ is again chosen to be $\frac{2d}{\epsilon}$, and we perform a similar analysis for the vertical segments, using a horizontal strip of height d around each line of $\mathcal{L'}_{z'}$. Using similar arguments, it can be shown that at least $(1 - \Omega(\epsilon))$-fraction of squares stabbed by an optimal solution remain unaffected for some choice of z, z'. Then, a similar enumeration and dynamic programming can be used to find a near-optimal solution. Note that the running time of the PTAS now becomes $n^{O(d^2/\epsilon^2)}$ since each bounded sub-instance has size $\frac{2d}{\epsilon} \times \frac{2d}{\epsilon}$.

5 APX-Hardness for Unbounded Segments

We note that a result of Kowalska and Pilipczuk [14], that shows APX-hardness for the problem of finding the minimum number of horizontal/vertical segments to cover all the given points in $\mathbb{R}^2$. It is not too difficult to see that, by scaling the instance appropriately, one can ensure that the minimum distance between two points, as well as that between two non-intersecting segments is at least a constant, say 5. Then, we replace each point p in the scaled instance by a unit square S_p centered at p; and if any segment s has p as its endpoint, then we slightly extend s so that it completely stabs S_p. It is straightforward to see the bijection between the set of feasible solutions for the original instance of Segment Set Cover, and the new instance of Stabbing Unit Squares, establishing the APX-hardness of the latter problem. Note that due to our initial scaling, the length of the segments is not necessarily bounded.

6 Conclusion

In this work, we consider a restricted variant of rectangle stabbing problem, where the rectangles are disjoint unit squares, and the horizontal segments have bounded length. We give PTASes for the Set Cover as well as Maximum Coverage variants of the problem. It is not too difficult to see that the "disjoint square" requirement can be relaxed to consider rectangles whose length and breadth are $\Theta(1)$. Even in this setting, our techniques will yield PTASes for the two variants with the same running time, up to the constants hidden in the big-Oh notation in the exponent.

We also sketch the modifications needed to adapt our PTASes for handling axis-parallel segments of both kinds, under the assumption that the segments have bounded length. In contrast, we observe that a known result in the literature implies APX-hardness for the problem *without* an upper bound on the lengths of the segments.

References

[1] H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite vc-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.

[2] T. M. Chan, E. Grant, J. Könemann, and M. Sharpe. Weighted capacitated, priority, and ge-

ometric set cover via improved quasi-uniform sampling. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1576–1585, 2012.

[3] T. M. Chan, T. C. van Dijk, K. Fleszar, J. Spoerhase, and A. Wolff. Stabbing Rectangles by Line Segments - How Decomposition Reduces the Shallow-Cell Complexity. In W.-L. Hsu, D.-T. Lee, and C.-S. Liao, editors, *29th International Symposium on Algorithms and Computation (ISAAC 2018)*, volume 123 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 61:1–61:13, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[4] K. L. Clarkson and K. Varadarajan. Improved approximation algorithms for geometric set cover. *Discrete & Computational Geometry*, 37(1):43–58, 2007.

[5] I. Dinur and D. Steurer. Analytical approach to parallel repetition. In D. B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633. ACM, 2014.

[6] F. Eisenbrand, M. Gallato, O. Svensson, and M. Venzin. A QPTAS for stabbing rectangles. *CoRR*, abs/2107.06571, 2021.

[7] T. Erlebach and E. J. Van Leeuwen. Ptas for weighted set cover on unit squares. In *Proceedings of the 13th International Conference on Approximation, and 14 the International Conference on Randomization, and Combinatorial Optimization: Algorithms and Techniques*, APPROX/RANDOM'10, pages 166–177, Berlin, Heidelberg, 2010. Springer-Verlag.

[8] R. Gandhi, S. Khuller, and A. Srinivasan. Approximation algorithms for partial covering problems. *J. Algorithms*, 53(1):55–84, 2004.

[9] D. R. Gaur, T. Ibaraki, and R. Krishnamurti. Constant ratio approximation algorithms for the rectangle stabbing problem and the rectilinear partitioning problem. *Journal of Algorithms*, 43(1):138–152, 2002.

[10] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and vlsi. *J. ACM*, 32(1):130–136, Jan. 1985.

[11] S. Jana and S. Pandit. Covering and packing of rectilinear subdivision. *Theor. Comput. Sci.*, 840:166–176, 2020.

[12] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

[13] A. Khan, A. Subramanian, and A. Wiese. A PTAS for the horizontal rectangle stabbing problem. *Math. Program.*, 206(1):607–630, 2024.

[14] K. Kowalska and M. Pilipczuk. Parameterized and Approximation Algorithms for Coverings Points with Segments in the Plane. In O. Beyersdorff, M. M. Kanté, O. Kupferman, and D. Lokshtanov, editors, *41st International Symposium on Theoretical Aspects of Computer Science (STACS 2024)*, volume 289 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 47:1–47:16, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[15] N. H. Mustafa and S. Ray. Improved results on geometric hitting set problems. *Discrete & Computational Geometry*, 44(4):883–895, 2010.

[16] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Math. Program.*, 14(1):265–294, 1978.

[17] K. R. Varadarajan. Epsilon nets and union complexity. In *Proceedings of the 25th ACM Symposium on Computational Geometry, Aarhus, Denmark, June 8-10, 2009*, pages 11–16, 2009.

[18] K. R. Varadarajan. Weighted geometric set cover via quasi-uniform sampling. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 641–648, 2010.

A Missing Proofs

In this section, we provide the proofs omitted from the main paper due to space constraints. For convenience, we restate the respective lemmas.

A.1 Proofs from Section 3

Lemma 3 [Good-z Value] *There exists a value $0 \le z < \frac{d}{\epsilon}$, satisfying that the number of squares of $\mathcal{O}$ that intersect any line of $\mathcal{L}_z$ is at most $\epsilon \cdot \mathrm{OPT}(\mathcal{I})$. Furthermore, for such a z, we can find in polynomial time a subset $\mathcal{S}_E$ of line segments from $\mathcal{S}$ that stabs all square in E_z, such that the total number of segments in $\mathcal{S}_E$ is at most $\epsilon \cdot \mathrm{OPT}(\mathcal{I})$.*

Proof. Suppose we select z uniformly at random from $[0, d/\epsilon)$. Now we define a random variable for each segment $s_i \in \mathcal{O}$:

$$X_{s_i} = \begin{cases} 1 & \text{if } s_i \text{ crosses a line of } \mathcal{L}_z, \\ 0 & \text{otherwise.} \end{cases}$$

Note that the probability that a vertical line from $\mathcal{L}_z$ intersects a fixed segment s is the ratio of the segment's length to the spacing between vertical lines. Since each segment has length at most d, this probability is at most $\frac{d}{d/\epsilon} = \epsilon$. Thus, for each $s_i \in \mathcal{O}$, we have that, $\mathbb{E}[X_{s_i}] = \Pr[X_{s_i} = 1] \le \epsilon$.

Let $X = \sum_{s_i \in \mathcal{O}} X_{s_i}$. By linearity of expectation, we obtain:

$$\mathbb{E}[X] = \sum_{s_i \in \mathcal{O}} \mathbb{E}[X_{s_i}] \le \epsilon \cdot \mathrm{OPT}(\mathcal{I}).$$

This shows the existence of a required $z \in [0, \frac{d}{\epsilon})$. Now, observe that square in E_z can only be stabbed by a segment that also crosses a line $\mathcal{L}_z$, and further no segment can stab two distinct squares in E_z. Since this property is also satisfied by $\mathcal{O}$, it follows that $|E_z|$ is bounded by $\epsilon \cdot \mathrm{OPT}(\mathcal{I})$. Such distinct segments can be found by arbitrarily picking one segment for each square in E_z. □

Lemma 4 [Crossing Segments] *The total optimal cost across all vertical sub-instances satisfies: $\sum_{j=1}^{q} \mathrm{OPT}(\mathcal{I}_j) \le (1 + c'\epsilon) \cdot \mathrm{OPT}(\mathcal{I})$, for some constant $c' > 0$.*

Proof. For each sub-instance $\mathcal{I}_j$, we define a corresponding feasible solution $\mathcal{O}_j \subseteq \mathcal{O}$ as follows. $\mathcal{O}_j$ consists of those segments in $\mathcal{O}$ that either lie entirely within the vertical strip defining $\mathcal{I}_j$, or intersect one of the vertical boundaries of the strip and stab at least one square in $\mathcal{R}_j$. By construction, these segments stab all the squares in $\mathcal{R}_j$, and hence $\mathcal{O}_j$ is a feasible solution for $\mathcal{I}_j$. Therefore,

$$\mathrm{OPT}(\mathcal{I}_j) \le |\mathcal{O}_j|.$$

We now decompose the original optimal solution $\mathcal{O}$ into two disjoint subsets:

- $\mathcal{O}^{\text{in}}$: segments that lie entirely within a single vertical strip,
- $\mathcal{O}^{\text{cross}}$: segments that intersect a vertical line separating adjacent strips.

Clearly, $\mathcal{O} = \mathcal{O}^{\text{in}} \uplus \mathcal{O}^{\text{cross}}$, and each segment in $\mathcal{O}^{\text{cross}}$ may appear in at most two sub-instance solutions.

Thus, the total number of segments used across all $\mathcal{O}_j$ satisfies:

$$\sum_{j=1}^{q} |\mathcal{O}_j| \le |\mathcal{O}^{\text{in}}| + 2|\mathcal{O}^{\text{cross}}| \le |\mathcal{O}| + 2|\mathcal{O}^{\text{cross}}|.$$

Following a similar argument as in the z-value lemma (Lemma 3), we can bound the number of crossing segments by $|\mathcal{O}^{\text{cross}}| \le c\epsilon \cdot \mathrm{OPT}(\mathcal{I})$, for some constant $c > 0$. While the exact lemma concerns expected values over a random choice of z, the underlying analysis applies here to control the total number of cross-strip segments in the fixed partition. Therefore,

$$\sum_{j=1}^{q} \mathrm{OPT}(\mathcal{I}_j) \le \sum_{j=1}^{q} |\mathcal{O}_j| \le (1 + 2c\epsilon) \cdot \mathrm{OPT}(\mathcal{I}),$$

which proves the lemma by setting $c' = 2c$. □

Theorem 1 *For any $\epsilon > 0$, there exists an algorithm that takes an instance of* Stabbing Disjoint Unit Squares, *runs in time $m^{O(d \log d/\epsilon^2)} \cdot n^{O(1)}$ and returns a $(1 + \epsilon)$-approximation.*

Proof. As established in Lemma 5, each sub-instance created via partitioning contains at most $O(d \log d/\epsilon^2)$ squares. For each such sub-instance $\mathcal{I}_{j,i}$, we exhaustively enumerate all subsets of segments of size at most $O(d \log d/\epsilon^2)$ and select the smallest feasible one that stabs all the contained squares. The number of such subsets is

$$\binom{m}{O(d \log d/\epsilon^2)} = m^{O(d \log d/\epsilon^2)}.$$

Thus, solving a single sub-instance $\mathcal{I}_{j,i}$ takes time $m^{O(d \log d/\epsilon^2)}$. Each vertical strip induces $t \le n$ such sub-instances.

Other components of the algorithm contribute only polynomial overhead:

- The set S_h, computed by horizontal sweeping, has size at most $\epsilon \cdot \mathrm{OPT}(\mathcal{I}_j)$ and is obtained in polynomial time.
- The set E_z, determined by the lines of $\mathcal{L}_z$, also has size at most $\epsilon \cdot \mathrm{OPT}(\mathcal{I})$ and is computable in polynomial time.

Lets focus on solution by each $\mathcal{I}_j$, it has subinstances which consists of optimally solved $\mathcal{I}_{i,j}$ and S_h bounded by $\epsilon \cdot \mathrm{OPT}(\mathcal{I}_j)$. Hence it has bound $(1+\epsilon) \cdot \mathrm{OPT}(\mathcal{I}_j)$. Lemma 4 implies that combining the solutions for vertical sub-instances and E_z will give a solution of total size at most $(1+\epsilon) \cdot (1+c'\epsilon) \cdot \mathrm{OPT}(\mathcal{I}) + \epsilon \cdot \mathrm{OPT}(\mathcal{I}) = (1+O(\epsilon)) \cdot \mathrm{OPT}(\mathcal{I})$. This completes the proof. □

A.2 Proofs from Section 4

Lemma 6 *There exists a value $0 \le z < \frac{2d}{\epsilon}$ such that the number of squares in $\mathcal{R}^*$ that are h-bad with respect to z is at most $\epsilon \cdot \mathsf{OPT}_k$.*

Proof. For a randomly chosen $z \in [0, \frac{2d}{\epsilon})$, the probability that a given square in $\mathcal{R}^*$ is h-bad is at most

$$\Pr[\text{a square is h-bad}] \le \frac{2d}{2d/\epsilon} = \epsilon.$$

Taking expectation over all $|\mathcal{R}^*| = \mathsf{OPT}_k$ squares, we have

$$\mathbb{E}[\#\text{ h-bad squares in } \mathcal{R}^*] \le \epsilon \cdot \mathsf{OPT}_k.$$

Hence, there must exist some $z \in [0, \frac{2d}{\epsilon})$ for which the number of h-bad squares is at most $\epsilon \cdot \mathsf{OPT}_k$, as desired. □

Lemma 8 *For each $i \ge 1$ and $j \ge 1$, let $\mathcal{I}_{i,j} = (\mathcal{R}_{i,j}, \mathcal{S}_{i,j})$ denote the j-th rectangular sub-instance of the i-th vertical sub-instance. Define $k_{i,j} \coloneqq |\mathcal{S}_{i,j} \cap \mathcal{O}_k(\mathcal{I})|$ and $t_{i,j} \coloneqq |\mathcal{R}_{i,j} \cap \mathcal{R}^*|$. Then:*

$$\sum_{i,j} \mathrm{OPT}_{k_{i,j}}(\mathcal{I}_{i,j}) \ge (1-2\epsilon) \cdot \mathrm{OPT}_k(\mathcal{I}).$$

Proof. Consider sub-instance $\mathcal{I}_{i,j}$. The subset $\mathcal{S}_{i,j} \cap \mathcal{O}_k(\mathcal{I})$ contains $k_{i,j}$ segments that can stab $t_{i,j}$ squares in $\mathcal{R}_{i,j} \cap \mathcal{R}^*$. This implies:

$$\mathrm{OPT}_{k_{i,j}}(\mathcal{I}_{i,j}) \ge t_{i,j}.$$

By Lemmas 10 and 11, at most $\epsilon \cdot \mathrm{OPT}_k(\mathcal{I})$ squares from $\mathcal{R}^*$ are h-bad and at most $\epsilon \cdot \mathrm{OPT}_k(\mathcal{I})$ are v-bad. Thus, at least $(1-2\epsilon) \cdot \mathrm{OPT}_k(\mathcal{I})$ squares from $\mathcal{R}^*$ are good and retained in the modified instance.

Because the sub-instances are disjoint, we have:

$$\sum_{i,j} t_{i,j} \ge (1-2\epsilon) \cdot \mathrm{OPT}_k.$$

Combining with the inequality $\mathrm{OPT}_{k_{i,j}}(\mathcal{I}_{i,j}) \ge t_{i,j}$, the lemma follows. □