Minimum Ply Covering of Points with Unit Squares*

Stephane Durocher[1], J. Mark Keil[2], and Debajyoti Mondal[2][0000−0002−7370−8697]

[1] University of Manitoba, Winnipeg, Canada
stephane.durocher@umanitoba.ca
[2] University of Saskatchewan, Saskatoon, Canada
{keil,dmondal}@cs.usask.ca

Abstract. Given a set P of points and a set U of axis-parallel unit squares in the Euclidean plane, a minimum ply cover of P with U is a subset of U that covers P and minimizes the number of squares that share a common intersection, called the minimum ply cover number of P with U. Biedl et al. [Comput. Geom., 94:101712, 2020] showed that determining the minimum ply cover number for a set of points by a set of axis-parallel unit squares is NP-hard, and gave a polynomial-time 2-approximation algorithm for instances in which the minimum ply cover number is constant. The question of whether there exists a polynomial-time approximation algorithm remained open when the minimum ply cover number is $\omega(1)$. We settle this open question and present a polynomial-time $(8+\varepsilon)$-approximation algorithm for the general problem, for every fixed $\varepsilon > 0$.

1 Introduction

The *ply* of a set S, denoted $\text{ply}(S)$, is the maximum cardinality of any subset of S that has a non-empty common intersection. The set S *covers* the set P if $P \subseteq \bigcup_{S_i \in S} S_i$. Given sets P and U, a subset $S \subseteq U$ is a *minimum ply cover* of P if S covers P and S minimizes $\text{ply}(S)$ over all subsets of U. Formally:

$$\text{plycover}(P, U) = \underset{\substack{S \subseteq U \\ S \text{ covers } P}}{\arg\min}\ \text{ply}(S). \tag{1}$$

The ply of such a set S is called the *minimum ply cover number* of P with U, denoted $\text{ply}^*(P, U)$. Motivated by applications in covering problems, including interference minimization in wireless networks, Biedl et al. [3] introduced the *minimum ply cover problem*: given sets P and U, find a subset $S \subseteq U$ that minimizes (1). They showed that the problem is NP-hard to solve exactly, and remains NP-hard to approximate by a ratio less than two when P is a set of points in $\mathbb{R}^2$ and U is a set of axis-aligned unit squares or a set of unit disks in

* This work is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

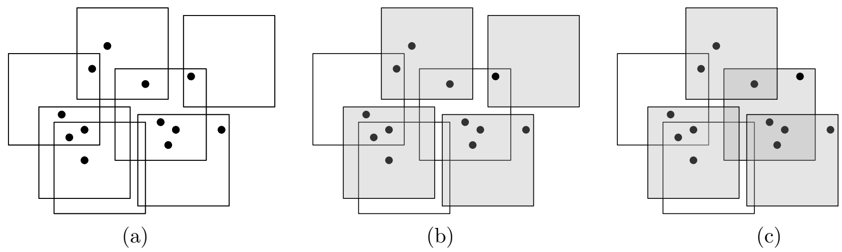

Fig. 1. (a) An input consisting of points and unit squares. (b) A covering of the points with ply 1, which is also the minimum ply cover number for the given input. (c) A covering of the points with ply 2.

$\mathbb{R}^2$. They also provided 2-approximation algorithms parameterized in terms of $\text{ply}^*(P,U)$ for unit disks and unit squares in $\mathbb{R}^2$. Their algorithm for axis-parallel unit squares runs in $O((k+|P|)(2\cdot|U|)^{3k+1})$ time, where $k=\text{ply}^*(P,U)$, which is polynomial when $\text{ply}^*(P,U)\in O(1)$. Biniaz and Lin [4] generalized this result for any fixed-size convex shape and obtained a 2-approximation algorithm when $\text{ply}^*(P,U)\in O(1)$. The problem of finding a polynomial-time approximation algorithm to the minimum ply cover problem remained open when the minimum ply cover number, $\text{ply}^*(P,U)$, is not bounded by any constant. This open problem is relevant to the motivating application of interference minimization. For example, algorithms for constructing a connected network on a given set of wireless nodes sometimes produce a network with high interference [8]. Selecting a set of network hubs that minimizes interference relates to the ply covering problem in a setting where ply may not be a constant.

Given a set P and a set U of subsets of P, the *minimum membership set cover problem*, introduced by Kuhn et al. [12], seeks to find a subset $S\subseteq U$ that covers P while minimizing the maximum number of elements of S that contain a common point of P. A rich body of research examines the minimum membership set cover problem (e.g., [6,13]). The minimum ply cover problem is a generalization of the minimum membership set cover problem: U is not restricted to subsets of P, and ply is measured at any point covered by U instead of being restricted to points in P. Consequently, the cardinality of a minimum membership set cover is at most the cardinality of a minimum ply cover. Erlebach and van Leeuwen [9] showed that the minimum membership set cover problem remains NP-hard when P is a set of points in $\mathbb{R}^2$ and U are unit squares or unit disks. For unit squares, they gave a 5-approximation algorithm for instances where the optimum objective value is bounded by a constant. Improved approximation algorithms are found in [2] and [10]. We refer the readers to [1,5] for more details on geometric set cover problems.

Our contribution: In this paper we consider the minimum ply cover problem for a set P of points in $\mathbb{R}^2$ with a set U of axis-aligned unit squares in $\mathbb{R}^2$. We show that for every fixed $\varepsilon>0$, the minimum ply cover number can be approximated in polynomial time for unit squares within a factor of $(8+\varepsilon)$. The algorithm is for the general case, i.e., no assumption on the ply cover of the input instance is needed. Hence, this settles an open question posed in [3] and [4].

Our algorithm overlays a regular grid on the plane and then approximates the ply cover number from the near exact solutions for these grid cells. The most interesting part of the algorithm is to model the idea of bounding the ply cover number with a set of budget points, and to exploit this set's geometric properties to enable dynamic programming to be applied. We show that one can set budget at the corners of a grid cell and check for a solution where the number of squares hit by a corner does not exceed its assigned budget. A major challenge to solve this decision problem is that the squares that hit the four corners may mutually intersect to create a ply that is bigger than any budget set at the corners. We show that an optimal solution can take a few well-behaved forms that can be leveraged to tackle this problem.

2 Minimum Ply Covering with Unit Squares

Let P be a finite set of points in $\mathbb{R}^2$ and let U be a set of axis-parallel unit squares in general position in $\mathbb{R}^2$, i.e., no two squares in U have edges that lie on a common vertical or horizontal line. In this section we give a polynomial-time algorithm to approximate the minimum ply cover number for P with U.

We consider a unit grid $\mathcal{G}$ over the point set P. The rows and columns of the grid are aligned with the x- and y-axes, respectively, and each cell of the grid is a unit square. We choose a grid that is in general position relative to the squares in U. In addition, no grid line intersects the points of P. A grid cell is called *non-empty* if it contains some points of P. We prove that one can first find a near exact ply cover for each non-empty grid cell R and then combine the solutions to obtain an approximate solution for P. We only focus on the ply inside R, because if the ply of a minimum ply cover is realized outside R, then there also exists a point inside R giving the same ply number.

We first show how to find a near exact ply cover when the points are bounded inside a unit square and then show how an approximate ply cover number can be computed for P. We will use the following property of a minimum ply cover. We include the proof in the full version [7] due to space constraint.

Lemma 1. *Let P be a set of points in a unit square R and let U be a set of axis-parallel unit squares such that each square contains either the top left or top right corner of R. Let $W_\ell \subseteq U$ and $W_r \subseteq U$ be the squares that contain the top left and top right corners of R, respectively. Let $S \subseteq U$ be a minimum ply cover of the points in R such that every square in S is necessary. In other words, if a square of S is removed, then the resulting set cannot cover all the points of R. Then S admits the property that one can remove at most one square from S to ensure that squares of $S \cap W_\ell$ do not intersect squares in $S \cap W_r$ (e.g., Fig. 2).*

Ply Cover for Points in a Grid Cell.

Let R be a 1×1 closed grid cell. Let $Q \subseteq P$ be the set of points in R, and let $W \subseteq U$ be the set of squares that intersect R. Note that by the construction of

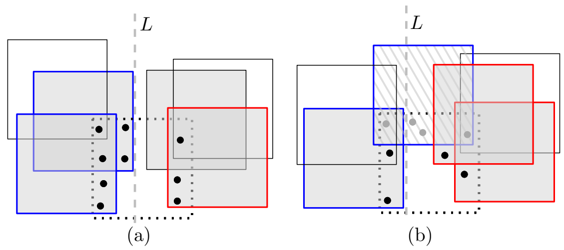

Fig. 2. (a)–(b) Illustration for the configuration of Lemma 1, where $(S \cap W_\ell)$ and $(S \cap W_r)$ are shown in blue and red, respectively. R is shown in dotted line.

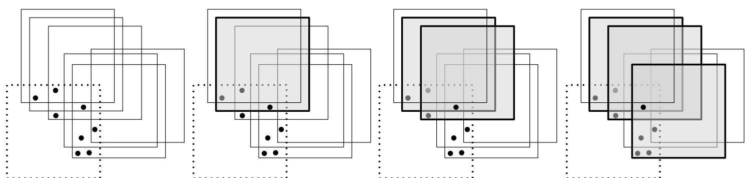

Fig. 3. Illustration for Case 1. The squares taken in the solution are shaded in gray. R is shown in dotted line.

the grid $\mathcal{G}$, every square in W contains exactly one corner of R. We distinguish some cases depending on the position of the squares in W. In each case we show how to compute either a minimum ply cover or a ply cover with ply at most four more than the minimum ply cover number in polynomial time.

Case 1 (A corner of R intersects all squares in W) In this case we compute a minimum ply cover. Without loss of generality assume that the top right corner of R intersects all the squares in W. We now can construct a minimum ply cover by the following greedy algorithm $\mathcal{G}$.

Step 1: Let z be the leftmost (break ties arbitrarily) uncovered point of Q. Find the square $B \in W$ with the lowest bottom boundary among the squares that contain z.

Step 2: Add B to the solution, remove the points covered by B.

Step 3: Repeat Steps 1 and 2 unless all the points are covered.

Fig. 3 illustrates such an example for Case 1. It is straightforward to compute such a solution in $O((|W| + |Q|)\log^2(|W| + |Q|))$ time using standard dynamic data structures, i.e, the point z can be maintained using a range tree and the square B can be maintained by leveraging dynamic segment trees [11].

Lemma 2. *Algorithm $\mathcal{G}$ computes a minimum ply cover.*

Proof. To verify the correctness of the greedy algorithm, first observe that in this case the number of squares in a minimum cardinality cover coincides with a minimum ply cover. We now show that the above greedy algorithm constructs a minimum cardinality cover. We employ an induction on the number of squares in a minimum cardinality cover. Let $W_1, W_2, \ldots, W_k$ be a set of squares in a minimum cardinality cover. First consider the base case where $k = 1$. Since W_1 covers all the points, it also covers z. Since z is the leftmost point and since our choice of square B has the lowest bottom boundary, B must cover all the points. Assume now that if a minimum cardinality cover contains less than k squares, then the greedy algorithm constructs a minimum cardinality cover. Consider now the case when we have k squares in a minimum cardinality cover. For any minimum cardinality cover, if z is covered by a square W_1, then we can replace it with the greedy choice B. The reason is that any point covered by W_1 would also be covered by B. By induction hypothesis, we have a minimum cardinality cover for the points that are not covered by B. Hence the greedy solution must give a minimum cardinality cover. □

Case 2 (Two consecutive corners of R intersect all the squares in W) In this case we compute a minimum ply cover. Without loss of generality assume that the top left and top right corners of R intersect all the squares in W. Let W_ℓ and W_r be the squares of W that intersect the top left corner and top right corner, respectively. We construct a minimum ply cover by considering whether a square of W_ℓ intersects a square of W_r.

If the squares of W_ℓ do not intersect the squares of W_r, then we can reduce it into two subproblems of type Case 1. We solve them independently and it is straightforward to observe that the resulting solution yields a minimum ply cover. Similar to Case 1, here we need $O((|W| + |Q|) \log^2(|W| + |Q|))$ time. Consider now the case when some squares in W_ℓ intersect some squares of W_r. By Lemma 1, there exists a minimum ply cover S such that at least one of the following two properties hold:

- C_1 There exists a vertical line L that passes through the left or right side of some square and separates $S \cap W_\ell$ and $S \cap W_r$, as illustrated in Fig. 2(a).
- C_2 There exists a square M in S such that after the removal of M from S, one can find a vertical line L that passes through the left or right side of some square and separates $(S \setminus \{M\}) \cap W_\ell$ and $(S \setminus \{M\}) \cap W_r$. This is illustrated in Fig. 2(b), where the square M is shown with the falling pattern.

To find a minimum ply cover, we thus try out all possible L (for C_1), and all possible M and L (for C_2). More specifically, to consider C_1, for each vertical line L passing through the left or right side of some square in W, we independently find a minimum ply cover for the points and squares on the left halfplane of L and right halfplane of L. We then construct a ply cover of Q by taking the union of these two minimum ply covers.

To consider C_2, for each square M, we first delete M and the points it covers. Then for each vertical line L determined by the squares in $(W \setminus \{M\})$, we independently find a minimum ply cover for the points and squares on the left halfplane of L and right halfplane of L. We then construct a ply cover of Q by taking the union of these two minimum ply covers and M. Finally, among all the ply covers constructed considering C_1 and C_2, we choose the ply cover with the minimum ply as the minimum ply cover of Q.

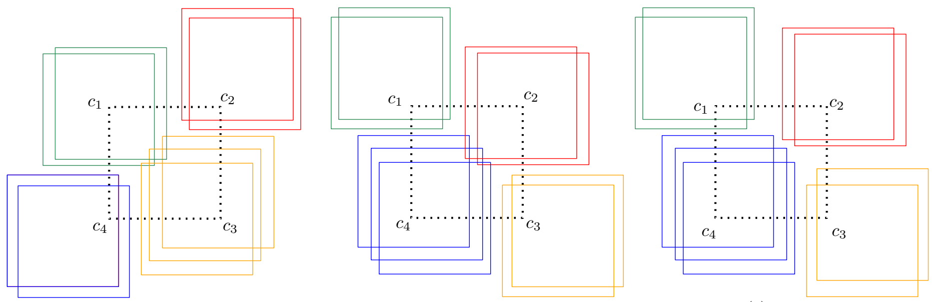

Fig. 4. Illustration for the scenarios that may occur after applying Lemma 1: (a)–(b) Diagonal, and (c) Disjoint. R is shown in dotted line.

Since there are $O(|W|)$ possible choices for L and $O(|W|)$ possible choices for M, the number of ply covers that we construct is $O(|W|^2)$. Each of these ply covers consists of two independent solutions that can be computed in $O((|W|+|Q|)\log^2(|W|+|Q|))$ time using the strategy of Case 1. Hence the overall running time is $O((|W|^3 + |W|^2|Q|)\log^2(|W|+|Q|))$.

Case 3 (Either two opposite corners or at least three corners of R intersect the squares in W) Let S be a minimum ply cover of Q such that all the squares in S are necessary (i.e., removing a square from S will fail to cover Q). Let c_1, c_2, c_3, c_4 be the top-left, top-right, bottom-right and bottom-left corners of R, respectively. Let W_i, where $1 \le i \le 4$, be the squares of W that contain c_i. Similarly, let S_i be the subset of squares in S that contain c_i.

By Lemma 1, one can remove at most four squares from S such that the squares of S_i do not intersect the squares of $S_{(i \bmod 4)+1}$. We assume these squares to be in the solution and hence also remove the points they cover. Consequently, we now have only the following possible scenarios after the deletion.

Diagonal: The squares of S_i do not intersect the squares of $S_{(i \bmod 4)+1}$. The squares of S_1 may intersect the squares of S_3, but the squares of S_2 do not intersect the squares of S_4 (or, vice versa). See Fig. 4(a) and (b).

Disjoint: If two squares intersect, then they belong to the same set, e.g., Fig. 4(c).

We will compute a minimum ply cover in both scenarios. However, considering the squares we deleted, the ply of the final ply cover we compute may be at most four more than the minimum ply cover number.

Case 3.1 (Scenario Diagonal) We now consider the scenario Diagonal. Our idea is to perform a search on the objective function to determine the minimum ply cover number. Let k be a guess for the minimum ply cover number. If $k \le 4$, we will show how to leverage Case 1 to verify whether the guess is correct. If $k > 4$, then one can observe that the ply is determined by a corner of R, as follows. Let H be the common rectangular region of k mutually intersecting squares in the solution. If H does not contain any corner of R, then it lies interior to R. Since H is a rectangular region, we could keep only the squares that determine the boundaries of H to obtain the same point covering with at most 4 squares. Therefore, for $k > 4$, the region determining the ply cover number must include a corner of R. We will use a dynamic program to determine such a ply cover (if exists).

In general, by $T(r, k_1, k_2, k_3, k_4)$ we denote the problem of finding a minimum ply cover for the points in a rectangle r such that the ply is at most $\max\{k_1, k_2, k_3, k_4\}$, and each corner c_i respects its budget k_i, i.e., c_i does not intersect more than k_i squares. We will show that r can always be expressed as a region bounded by at most four squares in W and T returns a feasible ply cover if it exists. To express the original problem, we add four dummy squares in W determined by the four sides of R such that they lie outside of R. Thus $r = R$ is the region bounded by the four dummy squares.

We are now ready to describe the details. Without loss of generality assume that a square $A \in S_4$ intersects a square $B \in S_2$, as shown in Fig. 5(a). We assume A and B to be in a minimum ply cover of R and try out all such pairs. We first consider the case when $k \le 4$ and the minimum ply cover already contains A and B. We enumerate all $O(|W|^4)$ possible options for $k \le 4$, S_2, and S_4 with $\text{ply}(S_2 \cup S_4) \le k$ and for each option, we use Case 1 to determine whether $\text{ply}(W_1)$ and $\text{ply}(W_3)$ are both upper bounded by k. We thus compute the solution to $T(r, k_1, k_2, k_3, k_4)$ and store them in a table $D(r, k_1, k_2, k_3, k_4)$, which takes $O((|W|^5 + |W|^4|Q|)\log^2(|W| + |Q|))$ time.

We now show how to decompose $T(r, k_1, k_2, k_3, k_4)$ into two subproblems assuming that the minimum ply cover already contains A and B. We will use the table D as a subroutine.

The first subproblem consists of the points that lie above A and to the left of B, e.g., Fig. 5(a) and (b). We refer to this set of points by Q_1. The corresponding region r' is bounded by four squares: A, B, and the two (dummy) squares from r. We now describe the squares that need to be considered to cover these points.

- Note that for Diagonal, no square in S_1 intersects A or B, hence we can only focus on the squares of W_1 that do not intersect A or B.
- The squares of W_2 that do not intersect Q_1 are removed. The squares of W_2 that contains the bottom left corner of B are removed because including them will make B an unnecessary square in the cover to be constructed.
- Similarly, the squares of W_4 that do not intersect Q_1 or contains the top right corner of A are removed.
- No square in W_3 needs to be considered since to cover a point of Q_1 it must intersect A or B, which is not allowed in Diagonal.

The second subproblem consists of the points that lie below B and to the right of A, e.g., Fig. 5(a) and (c). The corresponding region r'' is bounded by four squares: A, B, and the two squares from r. We denote these points by Q_2. The squares to be considered can be described symmetrically.

Let W' and W'' be the set of squares considered to cover Q_1 and Q_2, respectively. By the construction of the two subproblems, we have $Q_1 \cap Q_2 = \varnothing$ and $W' \cap W'' = \varnothing$.

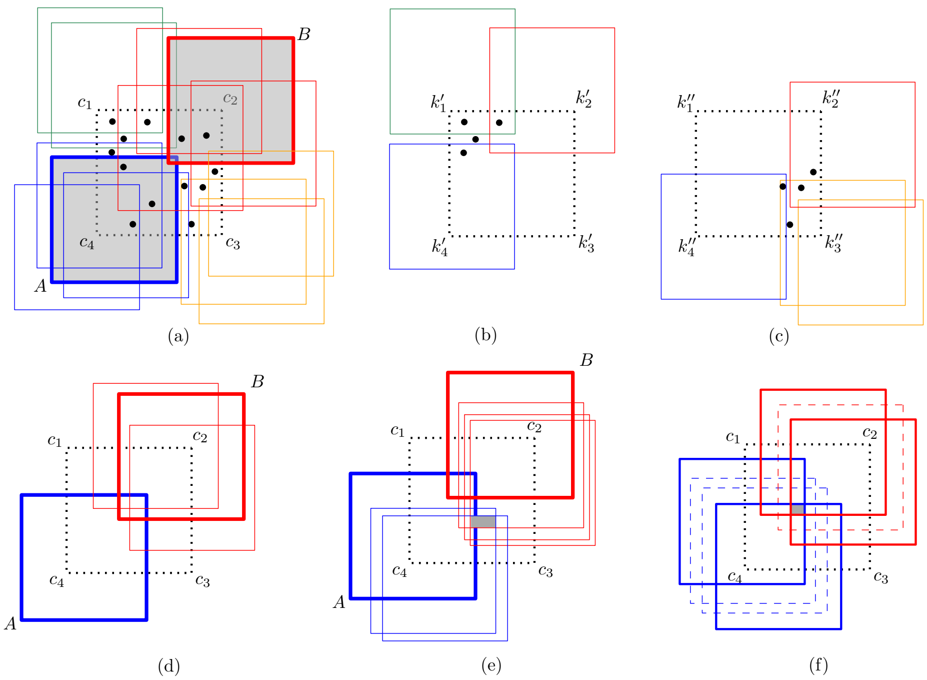

Fig. 5. Illustration for the dynamic program. (a)–(c) Decomposition into subproblems. (d)–(f) Illustration for the $(k+1)$ mutually intersecting squares. The dashed squares can be safely discarded. R is shown in dotted line.

For each corner c_i, we use k'_i and k''_i to denote the budgets allocated for c_i in the first and the second subproblems, respectively. Since we need to ensure that the ply of the problem T is at most $k = \max\{k_1, k_2, k_3, k_4\}$ and each corner c_i respects its budget k_i, we need to carefully distribute the budget among the subproblems when constructing the recurrence formula. Furthermore, let S'_2 and S'_4 be the sets of squares corresponding to c_2 and c_4 that are returned as the solution to the first subproblem. Similarly, define S''_2 and S''_4 for the second subproblem. We now have the following recurrence formula.

$$\begin{matrix} T(r,k_1, \\ k_2,k_3,k_4) \end{matrix} = \min_{\substack{\{A\in W_4, B\in W_2 : A\cap B\neq\varnothing\} \\ k'_1=k''_1=k_1, k'_3=k''_3=k_3, \\ k'_2+k''_2=k_2-1, \\ k'_4+k''_4=k_4-1}} \begin{cases} \begin{matrix} T(r',k'_1,k'_2,k'_3,k'_4)\cup \\ T(r'',k''_1,k''_2,k''_3,k''_4)\cup\{A,B\} \end{matrix} & \text{, if } \delta \le k \\ \begin{matrix} T(r',k'_1,k'_2,k'_3,k'_4)\cup \\ T(r'',k''_1,k''_2,k''_3,k''_4)\cup\beta \end{matrix} & \text{, if } \delta > k \text{ and } k \ge 4 \\ D(r,k_1,k_2,k_3,k_4) & \text{, if } \delta > k \text{ and } k \le 3 \end{cases}$$

Here δ is the ply of $(S'_2 \cup S'_4 \cup S''_2 \cup S''_4 \cup A \cup B)$ and β is the set of squares that remain after discarding unnecessary squares from $(S'_2 \cup S'_4 \cup S''_2 \cup S''_4 \cup A \cup B)$, i.e., removal of these squares would still ensure that all points are covered by the remaining squares. Since S_1 and S_4 are disjoint, one can also set $k'_3 = 0$ in $T(r', k'_1, k'_2, k'_3, k'_4)$ and $k''_1 = 0$ in $T(r'', k''_1, k''_2, k''_3, k''_4)$.

If $\delta \leq k$, then the union of $\{A, B\}$ and the squares obtained from the two subproblems must have a ply of at most k for the following two reasons. First, the squares of $S_1 = S'_1 \cup S''_1$ (similarly, S_3) cannot intersect the squares of $S_2 \cup S_4 = S'_2 \cup S''_2 \cup S'_4 \cup S''_4$. Second, by the budget distribution, the ply of S_1 can be at most $k_1 \leq k$ and the ply of S_3 can be at most $k_3 \leq k$.

If $\delta > k$ and $k \leq 4$, then each of S_1, S_2, S_3, S_4 can have at most three rectangles. We can look it up using the table $D(r, k_1, k_2, k_3, k_4)$.

If $\delta > k > 4$, then we can have $k + 1$ mutually intersecting squares in $(S'_2 \cup S'_4 \cup S''_2 \cup S''_4 \cup A \cup B)$ and in the following we show how to construct a solution with ply cover at most k respecting the budgets, or to determine whether no such solution exists.

If $T(r', k'_1, k'_2, k'_3, k'_4)$ and $T(r'', k''_1, k''_2, k''_3, k''_4)$ each returns a feasible solution, then we know that $(k + 1)$ mutually intersecting squares can neither appear in $S'_2 \cup S'_4$ nor in $S''_2 \cup S''_4$. Therefore, these $k+1$ mutually intersecting squares must include either both A and B, or at least one of A and B. We now consider the following options.

Option 1: S_4 and S_2 each contains at least two squares that belong to the set of $k + 1$ mutually intersecting squares. Since the region created by the $k + 1$ mutually intersecting squares is a rectangle, as illustrated in Fig. 5(f), we can keep only the squares that determine the boundaries of this rectangle to obtain the same point covering.

After discarding the unnecessary squares, we only have β squares where $|\beta| = 4 < k$. Thus the ply of the union of $S_1 \cup S_3$ and the remaining β squares is at most k. Hence we can obtain an affirmative solution by taking $T(r', k'_1, k'_2, k'_3, k'_4) \cup T(r'', k''_1, k''_2, k''_3, k''_4) \cup \beta$.

Option 2: S_4 only contains A and A intersects all k squares of $S'_2 \cup S''_2 \cup B$. Since the $k + 1$ mutually intersecting region is a rectangle, as illustrated in Fig. 5(d), we can keep only the squares that determine the boundaries of this rectangle to obtain the same point covering. After discarding the unnecessary squares, we only have β squares where $|\beta| = 3 < k$. Hence we can handle this case in the same way as in Option 1.

Option 3: S_2 only contains B and B intersects all k squares of $S'_4 \cup S''_4 \cup A$. This case is symmetric to Option 2.

In the base case of the recursion, we either covered all the points, or we obtain a set of problems of type Case 1 or of Scenario Disjoint (Case 3.1.2). The potential base cases corresponding to Case 1 are formed by guessing $O(|W|^2)$ pairs of intersecting squares from opposite corners, as illustrated in Fig. 6(a). The potential $O(|W|^4)$ base cases corresponding to Scenario Disjoint are formed by two pairs of intersecting squares from opposite corners, as illustrated in Fig. 6(b).

The precomputation of the base cases takes $O(|W|^4 f(|W|, |Q|))$ time, where $f(|W|, |Q|)$ is the time to solve a problem of type Case 1 and of Scenario Disjoint. We will discuss the details of $f(|W|, |Q|)$ in the proof of Theorem 1.

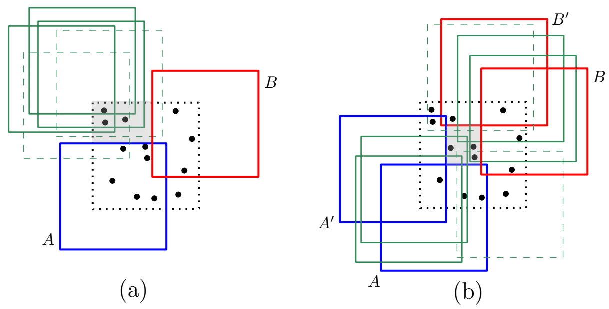

Fig. 6. Illustration for the base cases, where the region corresponding to the base cases are shown in gray. (a) The base case corresponds to Case 1, where we ignore the squares that intersect the chosen squares A and B. (b) An example of the base case that corresponds to scenario Disjoint, where we need to construct a solution such that no two squares from opposite corners intersect. We ignore all the squares of W_1 or W_3 that intersect the chosen squares A and B, or A' and B', as well as those that makes any of them unnecessary. R is shown in dotted line.

Since r is determined by at most four squares (e.g., Fig. 6), and since there are four budgets, the solution to the subproblems can be stored in a dynamic programming table of size $O(|W|^4k^4)$. Computing each entry requires examining $O(|W|^2)$ pairs of squares. Thus the overall running time becomes $O(|W|^6k^4 + |W|^4 f(|W|, |Q|))$.

Case 3.2 (Scenario Disjoint) In this case, we can find a sequence of empty rectangles $\sigma = (e_1, e_2, \ldots)$ from top to bottom such that they do not intersect any square of S, as illustrated in Fig. 7(a)–(b). The idea is again to exploit a dynamic programming with a budget given for each corner of R. A subproblem is expressed by a region determined by at most two squares — one intersecting the left side and the other intersecting the right side of R. In Fig. 7(c), this region is shown in gray. The overall running time for this case is $O(|W|^4k^4 + |W|^4 \log|Q| + |Q|\log|Q|)$. See the full version [7] for more details.

The following theorem combines all cases and its proof is in full version [7].

Theorem 1. *Given a set Q of points inside a unit square R and a set W of axis-parallel unit squares, a ply cover of size $4+k^*$ can be computed in $O((|W|^8(k^*)^4 + |W|^8 \log|Q| + |W|^4|Q|\log|Q|)\log k^*)$ time, where $k^* = \mathrm{ply}^*(Q, W) \leq \min\{|Q|, |W|\}$.*

Covering a General Point Set.

Given a set P of points and a set U of axis-parallel unit squares, both in $\mathbb{R}^2$, we now give a polynomial-time algorithm that returns a ply cover of P with U whose ply is at most $(8+\varepsilon)$ times the minimum ply cover number of P with U. Recall that our algorithm partitions P along a unit grid and applies Theorem 1 iteratively at each grid cell to select a subset of U that is a near minimum ply cover for the grid cell. Elements of U selected to cover points of P in a given grid cell overlap neighbouring grid cells, which can cause the ply to increase in those

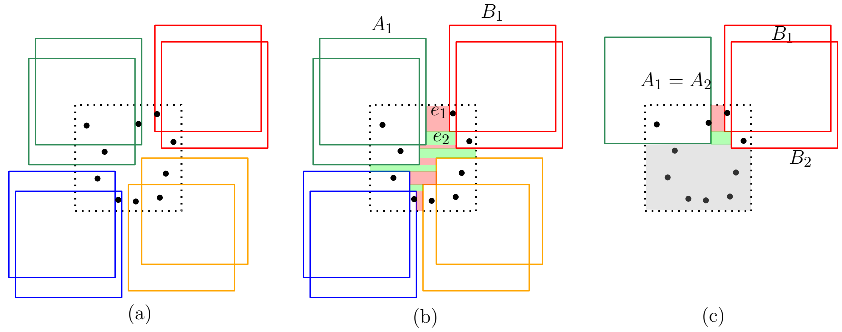

Fig. 7. Illustration for the dynamic program. R is shown in dotted line.

neighbouring cells; Lemma 3 allows us to prove Theorem 2 and Corollary 1, showing that the resulting ply is at most $(8 + \varepsilon)$ times the optimal value.

Partition P using a unit grid, i.e., each cell in the partition contains $P \cap [i, i+i) \times [j, j+1)$, for some $i, j \in \mathbb{Z}$. Each grid cell has eight grid cells adjacent to it. Let $C_1, \ldots, C_4$ denote the four grid cells in counter-clockwise order that are its diagonal neighbours. We now have the following lemma with the proof in the full version [7].

Lemma 3. *If any point p in a grid cell C is contained in four squares, $\{S_1, \ldots, S_4\} \subseteq U$, such that for each $i \in \{1, \ldots, 4\}$, S_i intersects the cell C_i that is C's diagonal grid neighbour, then $C \subseteq S_1 \cup S_2 \cup S_3 \cup S_4$.*

We now partition P along a unit grid and apply Theorem 1 iteratively to find a near minimum ply cover for each grid cell. For each cell that contains a point p of P, we leverage Lemma 3 to show that at most 8 grid cells can contribute to the ply of p. We thus obtain the following theorem with the proof in the full version [7].

Theorem 2. *Given a set P of points and a set U of axis-parallel unit squares, both in $\mathbb{R}^2$, a ply cover of P using U can be computed in $O((|U|^8(k^*)^4 + |U|^8 \log|P| + |U|^4|P|\log|P|)\log k^*)$ time whose ply is at most $8k^*+32$, where $k^* = \text{ply}^*(P, U) \leq \min\{|P|, |U|\}$ denotes the minimum ply cover number of P by U.*

Corollary 1. *Given a set P of points and a set U of axis-parallel unit squares, both in $\mathbb{R}^2$, a ply cover of P using U can be computed in polynomial time whose ply is at most $(8+\varepsilon)$ times the minimum ply cover number $k^* = \text{ply}^*(P, U)$, for every fixed $\varepsilon > 0$.*

Proof. We use Theorem 2 to find a ply cover with ply at most $8k^*+32$, and then consider the following two cases. **Case 1.** Suppose $\varepsilon k^* \geq 32$. Then $8k^* + 32 \leq (8+\varepsilon)k^*$. **Case 2.** Suppose $\epsilon k^* < 32$. We apply the 2-approximation algorithm of Biedl et al. [3] in $O(|P| \cdot |U|)^{3k^*+1})$ time, which is polynomial since $k^* \in O(1)$. $\square$

3 Conclusion

We gave a $(8+\varepsilon)$-approximation polynomial-time algorithm for the minimum ply cover problem with axis-parallel unit squares. Through careful case analysis, it may be possible to further improve the running time of our approximation algorithm presented in Theorem 2. A natural direction for future research would be to reduce the approximation factor or to apply a different algorithmic technique with lower running time. It would also be interesting to examine whether our strategy can be generalized to find polynomial-time approximation algorithms for other covering shapes, such as unit disks or convex shapes of fixed size.

References

1. Agarwal, P.K., Ezra, E., Fox, K.: Geometric optimization revisited. In: Steffen, B., Woeginger, G.J. (eds.) Computing and Software Science - State of the Art and Perspectives, LNCS, vol. 10000, pp. 66–84. Springer (2019)
2. Basappa, M., Das, G.K.: Discrete unit square cover problem. Discret. Math. Algorithms Appl. **10**(6), 1850072:1–1850072:18 (2018)
3. Biedl, T.C., Biniaz, A., Lubiw, A.: Minimum ply covering of points with disks and squares. Comput. Geom. **94**, 101712 (2021)
4. Biniaz, A., Lin, Z.: Minimum ply covering of points with convex shapes. In: Proc. 32nd Canadian Conference on Computational Geometry (CCCG). pp. 2–5 (2020)
5. Chan, T.M., He, Q.: Faster approximation algorithms for geometric set cover. In: Cabello, S., Chen, D.Z. (eds.) Proc. of the 36th International Symposium on Computational Geometry (SoCG). LIPIcs, vol. 164, pp. 27:1–27:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020)
6. Demaine, E.D., Feige, U., Hajiaghayi, M., Salavatipour, M.R.: Combination can be hard: Approximability of the unique coverage problem. SIAM J. Comput. **38**(4), 1464–1483 (2008)
7. Durocher, S., Keil, J.M., Mondal, D.: Minimum ply covering of points with unit squares. CoRR **abs/2208.06122** (2022)
8. Durocher, S., Mehrpour, S.: Interference minimization in k-connected wireless networks. In: Proceedings of the 29th Canadian Conference on Computational Geometry (CCCG). pp. 113–119 (2017)
9. Erlebach, T., van Leeuwen, E.J.: Approximating geometric coverage problems. In: Teng, S. (ed.) Proc. 19th ACM-SIAM Symposium on Discrete Algorithms (SODA). pp. 1267–1276. SIAM (2008)
10. Erlebach, T., van Leeuwen, E.J.: PTAS for weighted set cover on unit squares. In: Serna, M.J., Shaltiel, R., Jansen, K., Rolim, J.D.P. (eds.) Proc. of the 13th International Workshop on Approximation, Randomization, and Combinatorial Optimization (APPROX). LNCS, vol. 6302, pp. 166–177. Springer (2010)
11. van Kreveld, M.J., Overmars, M.H.: Union-copy structures and dynamic segment trees. J. ACM **40**(3), 635–652 (1993)
12. Kuhn, F., Rickenbach, P., Wattenhofer, R., Welzl, E., Zollinger, A.: Interference in cellular networks: The minimum membership set cover problem. In: Proc. of the 11th Conference on Computing and Combinatorics (COCOON). LNSC, vol. 3595, pp. 188–198. Springer-Verlag (2005)
13. Misra, N., Moser, H., Raman, V., Saurabh, S., Sikdar, S.: The parameterized complexity of unique coverage and its variants. Algorithmica **65**(3), 517–544 (2013)