Disjoint Box Covering in a Rectilinear Polygon*

Sujoy Bhore[1], Guangping Li[2], Martin Nöllenburg[2], and Jules Wulms[2]

1 **Université libre de Bruxelles, Belgium**
sujoy.bhore@gmail.com
2 **Algorithms and Complexity Group, TU Wien, Vienna, Austria**
{guangping, noellenburg, jwulms}@ac.tuwien.ac.at

Abstract

We study a variant of the geometric covering problem, namely, *Disjoint Box Covering in a Rectilinear Polygon* (DBCR). Given a rectilinear polygon R with m edges and a finite set P of n points inside R, in the DBCR problem, the objective is to find a set $\mathcal{S}$ of pairwise disjoint axis-aligned rectangles, such that $\mathcal{S}$ covers P, each rectangle is fully contained inside the polygon R, and $\mathcal{S}$ has minimum cardinality. We show that this problem is NP-hard for points in general position. Moreover, we design an $O(n \log n + m \log m)$-time exact algorithm for the DBCR problem if the input rectilinear polygon is a histogram.

1 Introduction

In the geometric set cover problem, the input is a range space $\Sigma = (X, \mathcal{R})$, where X is a universe of points in $\mathbb{R}^d$ and $\mathcal{R}$ is a family of subsets of X called ranges. The subsets in $\mathcal{R}$ are defined by the intersection of X and geometric shapes such as axis-parallel rectangles, disks, etc. The objective is to select a minimum-size subset $\mathcal{C} \subseteq \mathcal{R}$ of ranges such that every point in the universe X is covered by some range in $\mathcal{C}$. The geometric set cover problem is known to be NP-complete even for simple geometric range spaces, e.g., when $\mathcal{R}$ is a set of unit disks or unit squares in $\mathbb{R}^2$ [16]. However, the problem is far more tractable in the geometric domain than the classical set cover problem. For instance, while it is known that the set cover problem is inapproximable within factor $O(\log n)$, there exist several polynomial time approximation schemes for the geometric variant of the problem that used underlying geometric structures to achieve better algorithmic results; see [1, 3, 12, 13].

We study a special variant of the geometric set cover problem, namely, the *Disjoint Box Covering in a Rectilinear Polygon* (DBCR) problem. Given a rectilinear polygon R, possibly with holes, and a finite set P of n points inside R, in the DBCR problem, the objective is to find a set $\mathcal{S}$ of pairwise disjoint axis-aligned rectangles, such that $\mathcal{S}$ covers P and each rectangle is fully contained inside R. The DBCR problem is particularly motivated by geographically-informed text-based visualizations, such as spatial word or tag clouds. Imagine that we are given a set of locations, the point set P, inside a geographic region, the polygon R, with several possible text labels per location, which may refer to different categories of P (colored points). It is important for spatial word clouds to not only capture the frequency or weight but additionally the spatial relevance of the words (see [9]). If we model the words as axis-parallel rectangles, then the objective is to cover P with disjoint rectangles such that each rectangle contains only points of the same color and stays inside R. In this work, we address this problem under the simplifying assumption that the points are inside a rectilinear polygon and are all of same color. Without any color restrictions, one

* Research supported by the Austrian Science Fund (FWF) under grant P31119.

37th European Workshop on Computational Geometry, St. Petersburg, Russia, April 7–9, 2021.
This is an extended abstract of a presentation given at EuroCG'21. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

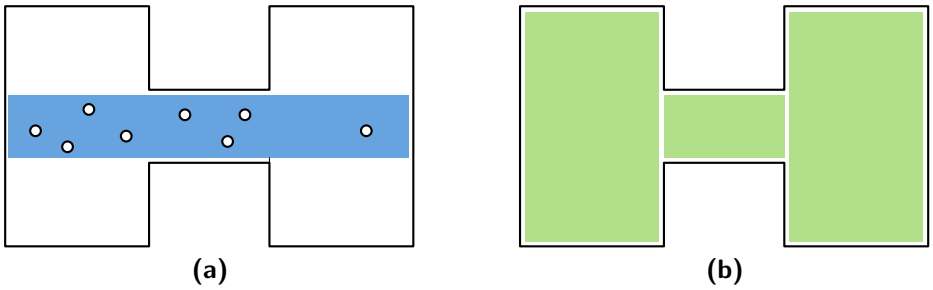

Figure 1 Comparing a solution to DBCR in **(a)** by a partitioning of the polygon in **(b)**.

way to solve the problem is to compute a minimum rectangular partitioning of the input polygon, which is solvable in polynomial time [18]. However, in a U-shaped polygon the solution of the partitioning problem could already differ from the solution of the DBCR problem (see Fig. 1). In fact, the DBCR problem is NP-hard if R is an arbitrary rectilinear polygon (Section 2). However, if R is a histogram, i.e., a rectilinear polygon consisting of a base line and a monotone path, we can give a polynomial-time algorithm (Section 3).

Related work. Many variants of the geometric covering problem have been investigated over the years, e.g., (p, k)-box covering [2]; class cover problems [8], red-blue cover [5, 11, 10]. Moreover, geometric covering problems have been widely studied in various algorithmic paradigms such as approximation algorithms (see [1, 13, 6, 14]) and parameterized algorithms (see [4, 7]). However, most of these works do not consider the case where the output objects must be disjoint. This makes the DBCR problem particularly unique, and none of these algorithms can be directly applied in our context.

2 NP-completeness of DBCR

In this section, we show firstly that the DBCR problem is NP-complete. We do a reduction similar to the one by Chan and Hu for red-blue set cover [11], using the following Lemmata.

▶ **Lemma 2.1.** *([19]) Every planar graph $G = (V, E)$ of maximum degree at most 4 has an orthogonal grid drawing on an $O(|V|) \times O(|V|)$ grid.*

▶ **Lemma 2.2.** (*Folklore*) *Given a graph G and an edge e in G, define a new graph G' obtained from G by subdividing e through the addition of two new vertices. Then the size of a minimum vertex cover of G' is exactly the size of a minimum vertex cover of G plus 1.*

▶ **Theorem 2.3.** *Disjoint Box Covering in a Rectilinear Polygon is NP-hard.*

Proof. We reduce from the vertex cover problem on 3-regular planar graphs, which is known to be NP-hard by Garey and Johnson [17]. Given a 3-regular planar graph G with n vertices, we create an orthogonal grid drawing δ_G by Lemma 2.1; see Figure 2a as an illustration. In the following, we construct a new graph G' by adding several dummy vertices in δ_G.

Firstly, we add a dummy vertex on each bend of δ_G, to create a straight-line drawing. We call a vertex v in δ_G a *corner* vertex, if there exists a pair of perpendicular line segments in δ_G that meet at v. Then, for each edge e connecting two corner vertices, we add a dummy vertex in the middle of e. This results in the new graph G', where each edge is incident to at most one corner vertex. To apply Lemma 2.2, for each original edge e in G, we check whether there is an even number of dummy vertices on e, including dummy corner vertices.

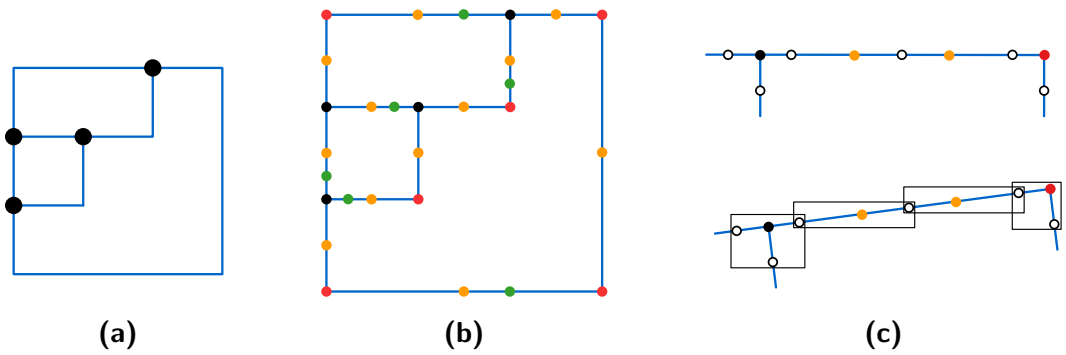

Figure 2 The reduction on graph $G = K_4$. **(a)** An orthogonal grid drawing δ_G of G. **(b)** The dummy vertices are added in δ_G to the bends (red) and on the edges, both to prevent multiple corner vertices incident to an edge (yellow), and for parity (green). **(c)** Cover rectangles, for black, red, yellow and green vertices, cover all (white) points in P and are drawn on the shifted grid.

If not, we then add a dummy vertex anywhere on the edge, to ensure the vertex cover of G' changes predictably (see Figure 2b).

To construct point set P in the DBCR instance from G', we replace each edge in G' by a point. If an edge e is incident to a corner vertex v, we put a point at $\frac{1}{3}$ of grid length from v on e. Otherwise, we create a point in the middle of the edge. Now we shift the points such that there are no two points on the same horizontal or vertical line. To achieve this, we map the drawing $\delta_{G'}$ to an orthogonal grid rotated by α. We look at each corner vertex c in $\delta_{G'}$ and consider each pair of orthogonal edges incident to c with lengths x_c and y_c. We choose $\alpha < \min\left\{\arctan(\min_c \frac{x_c}{y_c})/2, \arctan(\min_c \frac{y_c}{x_c})/2\right\}$. For each vertex v in G', we draw a rectangle B_v, such that B_v is the minimum bounding box covering v and the points on edges of v (see Figure 2c). By our choice of α these rectangles overlap only at points in P.

We say that B_v is the *cover rectangle* of the vertex v. For each corner vertex, we add a margin of length $\frac{1}{2}\epsilon$ to its cover rectangle, where ϵ is an infinitesimally small distance. This ensures that the covered points are not on its boundaries and the rectangles grow by ϵ both horizontally and vertically. Intuitively, there are two types of cover rectangles, small rectangles for corner vertices and long rectangular bars otherwise. We can then make the following observations.

▶ **Observation 2.4.** *Two cover rectangles intersect if and only if its corresponding vertices are adjacent; For each edge of G', only the rectangles of its two incident vertices intersect with it and the intersection contains the point on the edge.*

The union of cover rectangles builds the rectilinear polygon R in our DBCR instance.

▶ **Observation 2.5.** *Given an arbitrary rectangle S in R that covers a subset P' of point set P, the minimum bounding box of P' is inside a cover rectangle.*

After shifting, the points (and the corresponding cover rectangles) on a horizontal/vertical grid line are in a monotonic order. For two points that are not covered by the same rectangle B_v of any vertex v, there is no rectangle inside P that covers these points at the same time.

The correctness of the reduction is now straightforward. A vertex cover in G' corresponds to a set $\mathcal{S}$ of cover rectangles in R that covers all the points P. By Observation 2.4, if two cover rectangles intersect each other, their corresponding vertices are adjacent. Due to our construction, no two corner vertices are adjacent. Therefore, each intersection involves at

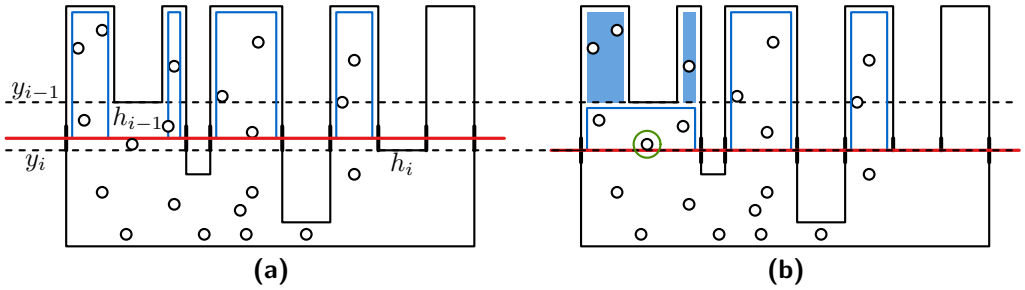

Figure 3 The red sweep line stores the vertical segments of the histogram (in bold) and the rectangles in the solution (in blue) at a certain y-coordinate. **(a)** The status is shown between events at y_{i-1} and y_i. **(b)** The point circled in green is found by querying Q_P at the event of h_i.

least one non-corner vertex. Consider a rectangles $S \in \mathcal{S}$ whose corresponding vertex is not a corner vertex. If the rectangle S intersects other rectangles in $\mathcal{S}$, we could eliminate the intersection by shrinking long rectangle S, while still covering all the points. After eliminating all the intersections in $\mathcal{S}$, it is a solution to the DBCR problem for point set P in R. In the other direction, we are given a set $\mathcal{S}$ of disjoint rectangles inside R that cover all the points in P. By Observation 2.5, each rectangle in $\mathcal{S}$ is completely inside a cover rectangle in the polygon R. We select all the cover rectangles that contains rectangles in $\mathcal{S}$. These rectangles cover all the points and their corresponding vertices in G' cover all the edges in G', thus they would build a vertex cover in G'. ◀

Containment in NP is straightforward for the DBCR problem, since we have to check only whether each point in P is covered by a rectangle in $\mathcal{S}$, and each rectangle in $\mathcal{S}$ is contained in R. These tasks can be done efficiently using point location data structures.

3 Covering points inside histogram

We solve the DBCR problem inside a *histogram*, which is defined as in [15] (see Figure 3).

Histogram: Define a (vertical) histogram with m edges as a rectilinear polygon with one horizontal edge (the base) equal in length to the sum of all other horizontal edges.

The DBCR problem in a histogram can be solved in polynomial time using a greedy algorithm $\mathcal{A}$. We assume the base of the input histogram H is located at $y = 0$ and extends only upwards, ending in $\frac{m}{2} - 1$ horizontal edges.

Algorithm $\mathcal{A}$ first sorts the $\frac{m}{2}$ horizontal pieces based on their y-coordinates in descending order, resulting in queue Q_H. Furthermore, the points in P are sorted by their y-coordinates, resulting in a queue Q_P. Using a horizontal sweep line l, an optimal solution $\mathcal{S}$ is constructed starting from the top of the histogram H. A *status* s stored with the sweep line maintains the x-coordinates of vertical rectangle edges in $\mathcal{S}$, as well as the x-coordinates of the vertical segments of H, at a certain y-coordinate (see Figure 3a).

As the sweep line l moves downwards, an event occurs at every horizontal segment h of H. At an event the status must be updated, and we query Q_H to find the next horizontal segment h_i of H. By querying Q_P, we can quickly find all points between h_i and the previous horizontal segment h_{i-1}, at y-coordinates y_i and y_{i-1}, respectively. If there are no points, then the solution $\mathcal{S}$ and status s require no change. However, if there are points, we check whether (at least) one of the points is not covered by the rectangles stored in s. If this is

the case for some point p, then we add a new rectangle to the solution $\mathcal{S}$, starting at y_{i-1} and spanning the whole width between the vertical segments of H neighboring p. These vertical segments can be found by querying s using the x-coordinate of p, and walking in both directions through s until a vertical segment is encountered. To prevent rectangles in $\mathcal{S}$ from overlapping, we settle and remove any rectangles from s that would otherwise overlap the newly added rectangle. The settled rectangles will vertically extend down to y_{i-1} in the final solution $\mathcal{S}$ (see Figure 3b).

Finally, depending on what type of segment h_i is, the status s must be updated accordingly. The segment h_i can have one of the following two types:

- Segment h_i is the top of a vertical spike of H. In this case, s is updated to include the x-coordinates of the vertical segments, pointing downwards from h_i.
- Segment h_i connects two vertical spikes of H. Status s must then be updated to no longer include the vertical segments that point downwards to h_i.

A complete solution found by algorithm $\mathcal{A}$ can be found in Figure 4.

▶ **Lemma 3.1.** *Algorithm $\mathcal{A}$ finds an optimal solution for the DBCR problem, given a set P of points inside a histogram H.*

Proof. We make a case distinction on whether an optimal solution OPT that has at least two rectangles r_1, r_2 horizontally next to each other without vertical histogram segments between them. First assume there are no such rectangles, and hence can extend each rectangle left- and rightwards until the polygon is hit, without intersecting any other rectangle. Next, we start from the base of the polygon and extend rectangles upwards until a horizontal edge of H is reached. Any rectangles intersected in the process shrink by moving their bottom edge upwards, leaving no points uncovered, since they will be covered by the rectangles that are extending upwards. The resulting solution is equivalent to a greedy solution of algorithm $\mathcal{A}$.

Now assume we have an optimal solution OPT that has at least two such rectangles r_1, r_2. Note that one of these rectangles may extend further upwards than the other. Assume w.l.o.g. that r_1 extends further upwards if this is the case.

Consider the greedy solution OPT', which instead contains rectangles r'_1, r'_2, such that r'_1 stops at the top of r_2 and r'_2 covers the whole space occupied by r_1, r_2 horizontally. Rectangle r'_2 can also extend as far downwards as the lowest boundary of r_1, r_2. This cannot lead to an intersection with the polygon boundary, because the histogram ends in a base that stretches the whole width horizontally. Furthermore, any intersected rectangles come in from the side or bottom, and can be safely shrunk, such that the points no longer covered by these shrunk rectangles are now covered by r'_2.

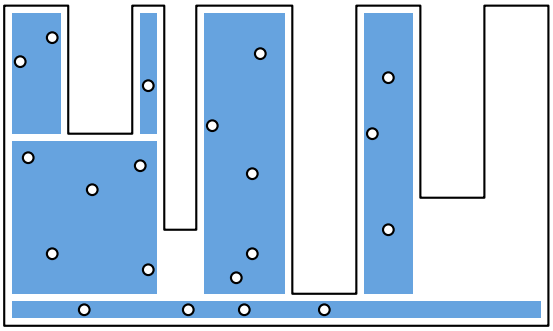

Figure 4 A complete solution for the example in Figure 3, as found by algorithm $\mathcal{A}$.

Finally, we can conclude that $|OPT| = |OPT'|$, since the number of rectangles stays equal in the transformation from OPT to OPT'. After applying this transformation for as long as there are two rectangles in the above horizontally neighboring position, all rectangles correspond to the greedily chosen rectangles of algorithm $\mathcal{A}$. ◀

Finally we show that algorithm $\mathcal{A}$ has polynomial running time.

▶ **Lemma 3.2.** *Algorithm $\mathcal{A}$ runs in $O(n \log n + m \log m)$ time.*

Proof. The $\frac{m}{2}$ horizontal segments of H and the n points in P are separately sorted and put in queues in $O(m \log m)$ and $O(n \log n)$ time, respectively. These queues are then queried, but no additional elements are added, again leading to $O(m \log m)$ and $O(n \log n)$ time spent. Finally, segments of histogram H and solution $\mathcal{S}$ are stored in and removed from status s. Since histogram H has $\frac{m}{2}$ vertical segments, and for each of the $\frac{m}{2}$ horizontal segments of H, only a single rectangle can exist in the solution, status s contains at most $\frac{m}{2}$ segments. As status s is sorted on x-coordinates, we can use a balanced binary search tree to ensure $O(\log m)$ update time. There are $\frac{m}{2}$ events, one for each horizontal segment, and hence maintaining the status takes $O(m \log m)$ time in total. ◀

▶ **Theorem 3.3.** *Given a set P of n points inside a histogram H with m edges, there exists an algorithm that solves the DBCR problem optimally in $O(n \log n + m \log m)$ time.*

References

1. Pankaj K. Agarwal and Jiangwei Pan. Near-linear algorithms for geometric hitting sets and set covers. *Discret. Comput. Geom.*, 63(2):460–482, 2020. doi:10.1007/s00454-019-00099-6.
2. Hee-Kap Ahn, Sang Won Bae, Erik D. Demaine, Martin L. Demaine, Sang-Sub Kim, Matias Korman, Iris Reinbacher, and Wanbin Son. Covering points by disjoint boxes with outliers. *Comput. Geom.*, 44(3):178–190, 2011. doi:10.1016/j.comgeo.2010.10.002.
3. Boris Aronov, Esther Ezra, and Micha Sharir. Small-size ϵ-nets for axis-parallel rectangles and boxes. *SIAM J. Comput.*, 39(7):3248–3282, 2010. doi:10.1137/090762968.
4. Pradeesha Ashok, Sudeshna Kolay, Neeldhara Misra, and Saket Saurabh. Unique covering problems with geometric sets. In Dachuan Xu, Donglei Du, and Dingzhu Du, editors, *Computing and Combinatorics*, pages 548–558, Cham, 2015. Springer International Publishing. doi:10.1007/978-3-319-21398-9_43.
5. Pradeesha Ashok, Sudeshna Kolay, and Saket Saurabh. Multivariate complexity analysis of geometric red blue set cover. *Algorithmica*, 79(3):667–697, 2017. doi:10.1007/s00453-016-0216-x.
6. Pradeesha Ashok, Aniket Basu Roy, and Sathish Govindarajan. Local search strikes again: PTAS for variants of geometric covering and packing. *J. Comb. Optim.*, 39(2):618–635, 2020. doi:10.1007/s10878-019-00432-y.
7. Aritra Banik, Fahad Panolan, Venkatesh Raman, Vibha Sahlot, and Saket Saurabh. Parameterized complexity of geometric covering problems having conflicts. *Algorithmica*, 82(1):1–19, 2020. doi:10.1007/s00453-019-00600-w.
8. Sergey Bereg, Sergio Cabello, José Miguel Díaz-Báñez, Pablo Pérez-Lantero, Carlos Seara, and Inmaculada Ventura. The class cover problem with boxes. *Computational Geometry*, 45(7):294–304, 2012. doi:10.1016/j.comgeo.2012.01.014.
9. Kevin Buchin, Daan Creemers, Andrea Lazzarotto, Bettina Speckmann, and Jules Wulms. Geo word clouds. In *Proc. 2016 IEEE Pacific Visualization Symposium (PacificVis)*, pages 144–151. IEEE Computer Society, 2016. doi:10.1109/PACIFICVIS.2016.7465262.

10 Robert D. Carr, Srinivas Doddi, Goran Konjevod, and Madhav Marathe. On the red-blue set cover problem. In *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, SODA '00, page 345–353, USA, 2000. Society for Industrial and Applied Mathematics. doi:10.5555/338219.338271.

11 Timothy M. Chan and Nan Hu. Geometric red-blue set cover for unit squares and related problems. *Comput. Geom.*, 48(5):380–385, 2015. doi:10.1016/j.comgeo.2014.12.005.

12 Kenneth L. Clarkson. Algorithms for polytope covering and approximation. In *Proc. 3rd Workshop on Algorithms and Data Structures (WADS)*, volume 709 of *Lecture Notes in Computer Science*, pages 246–252. Springer, 1993. doi:10.1007/3-540-57155-8_252.

13 Kenneth L. Clarkson and Kasturi R. Varadarajan. Improved approximation algorithms for geometric set cover. *Discret. Comput. Geom.*, 37(1):43–58, 2007. doi:10.1007/s00454-006-1273-8.

14 Erik D. Demaine, Uriel Feige, MohammadTaghi Hajiaghayi, and Mohammad R. Salavatipour. Combination can be hard: Approximability of the unique coverage problem. *SIAM J. Comput.*, 38(4):1464–1483, 2008. doi:10.1137/060656048.

15 Herbert Edelsbrunner, Joseph O'Rourke, and Emmerich Welzl. Stationing guards in rectilinear art galleries. *Computer Vision, Graphics, and Image Processing*, 27(2):167 – 176, 1984. doi:10.1016/S0734-189X(84)80041-9.

16 Robert J. Fowler, Mike Paterson, and Steven L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Inf. Process. Lett.*, 12(3):133–137, 1981. doi:10.1016/0020-0190(81)90111-3.

17 Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. WH Freeman and Company, New York, USA, 1990. doi:10.5555/574848.

18 W. T. Liou, Jimmy J. M. Tan, and Richard C. T. Lee. Minimum partitioning simple rectilinear polygons in $O(n \log \log n)$ time. In *Proc. 5th Symposium on Computational Geometry (SoCG)*, pages 344–353. ACM, 1989. doi:10.1145/73833.73871.

19 Leslie G Valiant. Universality considerations in VLSI circuits. *IEEE Transactions on Computers*, 100(2):135–140, 1981. doi:10.1109/TC.1981.6312176.