

Line Segment Covering of Cells in Arrangements*

Matias Korman Sheung-Hung Poon Marcel Roeloffzen

October 25, 2018

Abstract

Given a collection L of line segments, we consider its arrangement and study the problem of covering all cells with line segments of L . That is, we want to find a minimum-size set L' of line segments such that every cell in the arrangement has a line from L' defining its boundary. We show that the problem is NP-hard, even when all segments are axis-aligned. In fact, the problem is still NP-hard when we only need to cover rectangular cells of the arrangement. For the latter problem we also show that it is fixed parameter tractable with respect to the size of the optimal solution. Finally we provide a linear time algorithm for the case where cells of the arrangement are created by recursively subdividing a rectangle using horizontal and vertical cutting segments.

1 Introduction

Set cover [3] is one of the most fundamental problems of computer science. This problem is usually formulated in terms of hypergraphs: the input of the problem is a hypergraph $\mathcal{H} = (X, \mathcal{F})$ where $\mathcal{F} \subseteq 2^X$ is a collection of subsets of X , and we aim for a subset $\mathcal{F}' \subseteq \mathcal{F}$ of smallest cardinality that *covers* X (i.e., $\cup_{F \in \mathcal{F}'} F = X$). This problem is known to be NP-hard and even hard to approximate [3, 6].

Given its importance, it is not surprising that this problem has been studied extensively. In most cases, the set \mathcal{F} is given implicitly (this is specially true when considering geometric variants of the problem). For example, in the well-known k -center problem [5] we want to cover a set S of n points with unit disks. In the hypergraph definition, this is equivalent to $X = S$ and \mathcal{F} is the collection of subsets of S that can be covered with a single unit disk.

Sometimes the relationship between X and \mathcal{F} is much more involved. For example, in the *discrete center problem*, we only consider the disks whose center is a point of S . Akin to the discrete variant of the k -center problem, in this paper we study a geometric setting where the elements X and sets \mathcal{F} are defined by the same geometric primitives. Specifically, we study the problem of covering

*A preliminary version of this paper appeared in the proceedings of the COCOA 2015 conference [8]

the cells of an arrangement of line segments L with segments of L . Given a set L of line segments in the plane a *cell* in the arrangement of L is defined as a maximally connected region that is not intersected by any segments of L . Essentially the cells are the ‘empty’—not intersected by segments of L —regions in the arrangement defined by L . Now let C denote the set of all cells in the arrangement of L . We say that a cell $c \in C$ is *covered* by a line segment $\ell \in L$ if and only if ℓ is part of the boundary of c . Similarly c is covered by a set L' of line segments if and only if there is a segment $\ell \in L'$ that covers c . The goal is then to find a minimum-size set $L' \subset L$ that covers all cells of C . We call this the *line-segment covering* problem.

The problem can also be viewed as a guarding problem. In the traditional art gallery problem, the goal is to place guards so that the guards together see the whole gallery (often a simple polygon). Many variants of this have been studied. Bose *et al.* [2] study guarding and coloring problems between lines. They provide results for several types of guards and objects to guard, such as guarding the cells of the arrangement with the lines, or guarding the lines by selecting cells. Their results however do not extend to line segments as they use properties of the lines that do not hold for line segments. To the best of our knowledge covering cells in an arrangement of line segments with the segments has not been studied before.

We study three different variants of this problem. First, in Section 2 we show that the line-segment covering problem is NP-hard, even when all segments of L are axis-aligned. In Section 3 we consider a slightly different variant, where we are required to cover only rectangular cells, those defined by four line segments. For this variant we show that the NP-hardness reduction still works. However, we show that this variant is fixed parameter tractable with respect to the size of the optimal solution. In Section 4, inspired by subdivisions induced by KD-trees, we study a variant where the line segments define a type of rectangular subdivision. That is, an axis aligned rectangle that is recursively subdivided with horizontal or vertical line segments, similar to the subdivision defined by a KD-tree [1]. For this case we show that an optimal cover can be computed in linear time, assuming that the partitioning is given as a tree-structure defined by the splitting lines.

2 NP-hardness for Rectilinear Line Segments

In this section we show that the line-segment covering problem is NP-hard, even if the input consists of only horizontal and vertical line segments. We reduce the problem from PLANAR 3SAT [9]. An input instance for the 3SAT problem is a set $\{x_1, x_2, \dots, x_n\}$ of n variables and a Boolean expression in conjunctive normal form. That is, the expression is a conjunction of clauses $\Phi = c_1 \wedge \dots \wedge c_m$ such that each clause c_i is a disjunction of three literals (a variable or negation of a variable). The problem is then to decide if there is a truth assignment for the variables so that Φ is true. In PLANAR 3SAT we impose further restrictions by looking at the representation of Φ as a bipartite graph with variables and clauses

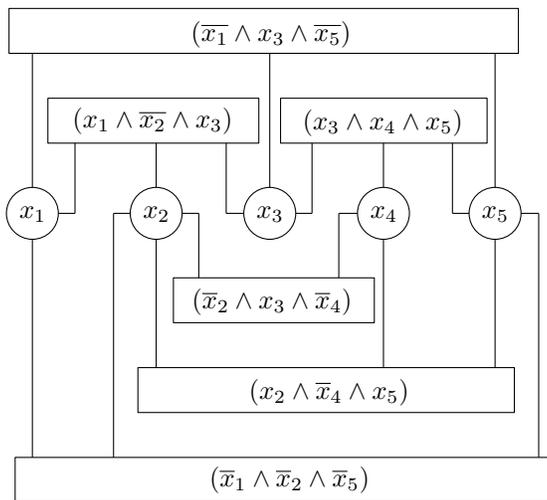


Figure 1: PLANAR 3SAT problem instance along with a planar embedding.

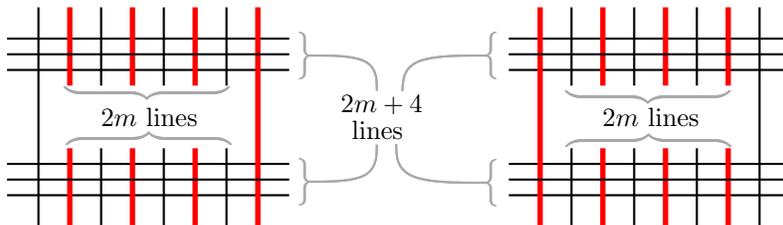


Figure 2: Gadget for a variable with a true (left) and false (right) assignment. Red (thick) edges show the two possible covers with $m + 1$ segments.

as vertices. A variable-node v is connected to a clause-node c if and only if v occurs in c . In PLANAR 3SAT we assume that this graph is planar. Specifically we assume that a planar embedding is given that places all variable-nodes on a horizontal line and all clause-nodes above or below this line (see Fig. 1). We also assume that no variable appears more than once in any clause (that is, the above described bipartite graph is a proper graph and not a multigraph).

It is well-known that the PLANAR 3SAT problem is NP-hard [9]. Also note that it is easy to see that the line-segment covering problem is in NP. Indeed, given a possible covering, we can construct the arrangement of line segments and verify in polynomial time that indeed all cells are covered. In the remainder of this section we provide a polynomial time reduction and prove its correctness.

2.1 Reduction

For each variable in Φ we create a gadget consisting of $4m + 8$ horizontal segments and $4m + 2$ vertical segments. The leftmost and rightmost vertical line stab all

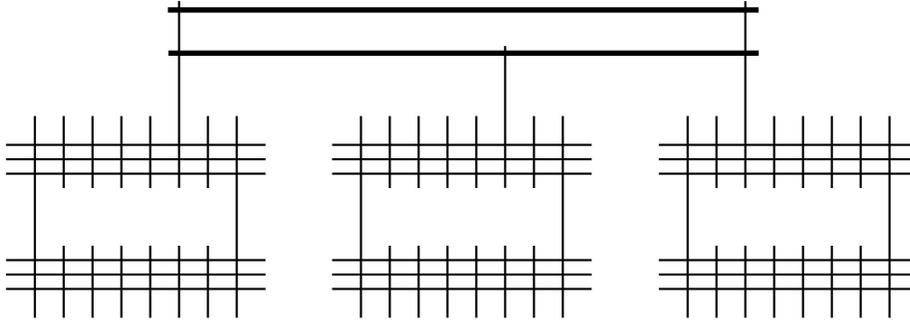


Figure 3: A clause gadget used in showing NP-hardness in Section 2, parts are variable gadgets that connect to the edges of a clause gadget (marked with thicker line segments).

horizontal lines of the gadget, whereas $2m$ of the other lines stab the top $2m + 4$ horizontal lines and $2m$ stab the lower $2m + 4$ horizontal lines as illustrated in Fig. 2. As we describe later, some of the vertical line segments may be further extended (above or below) to connect to the clause gadgets, but the horizontal segments will not cross any segments of other gadgets.

Intuitively speaking, we will show that any covering of the variable gadgets must choose one every other vertical segment, including either the right or leftmost segment. The choice of using either the rightmost or leftmost vertical segment is equivalent to assigning the variable to be true or false. The clause gadget will create additional cells that will be covered *for free* (without selecting additional line segments) provided that at least one variable satisfies the clause.

Let $s_1^{(i)}, \dots, s_{2m}^{(i)}$ be the vertical segments created in the gadget for variable x_i (numbered from left to right). We would like to sort the clauses c_1, \dots, c_j in which x_i occurs in the order in which they appear on the embedding of the PLANAR 3SAT instance. However, this is not well-defined (since it is not always clear when a segment goes before another), so we proceed as follows: let $c_1, \dots, c_{j'}$ be the clauses that contain variable x_i and are embedded above the line containing all variable nodes, sorted in clockwise order of their connections to x_i . Similarly, let $c_{j'+1}, \dots, c_j$ be the clauses that are embedded below the line (this time in counter-clockwise order). We define the ordering of the clauses around x_i as the concatenation of both orderings. Since we have $2m$ vertical line segments for each clause and m clauses we ensure that any vertical segment of a variable gadget is extended only towards a single clause, and any clause is associated to exactly three segments.

We now detail the gadget associated to clause $c = \ell_i \wedge \ell_j \wedge \ell_k$ (for $i < j < k$), where ℓ_i is a literal of variable x_i (similarly, ℓ_j and ℓ_k are literals of variables x_j and x_k , respectively). First, we extend the three segments associated to clause c (above or below depending on where c is placed in the embedding). We extend the segments associated to variables x_i and x_k slightly further than the segment of x_j . We complete our transformation by adding two horizontal segments that

create a rectangle with the three extended segments, see Fig. 3.

This concludes the construction of a line-segment-covering instance L from a PLANAR 3SAT input Φ . Next we show that there is a satisfying assignment for Φ if and only if there is a subset L' of L of size at most $n(2m + 1)$ that covers all cells in the arrangement.

2.2 Correctness

In the following theorem, we show that the line segment covering problem is NP-complete even when the input line segments are rectilinear.

Lemma 1. *A PLANAR 3SAT expression Φ is satisfiable if and only if there is a cover of size at most $n(2m + 1)$ for its corresponding line-segment-covering instance L .*

Proof. First we prove that given a satisfying assignment for Φ we can cover L with $n(2m + 1)$ segments. If a variable is true, then we select the rightmost vertical segment, and for each set of $2m$ segments that only intersect the top or bottom we select the odd ones counting from the leftmost segment starting at one, see also Fig. 2. If the variable is false we select the leftmost longer segment and the even ones from the sets of shorter segments.

Next we show that all cells are covered. We consider three types of cells: cells in the interior of the grids created of the variable gadgets are called *variable cells*, the single rectangular cell associated to a clause gadget is called *clause cell*; any other cell (included the unbounded one) that is created with our construction is simply called an *other cell*.

Since we have selected one every other segment, clearly all variable cells are covered. The fact that the variable assignment satisfies all clauses implies that at least one of the three vertical segments defining a clause cell has been selected, so the clause cell is covered. Hence, all clause cells are also covered. Finally, for the remaining cells it suffices to see that each such cell always has two consecutive vertical segments of a variable gadget in its boundary. Indeed, Such cells are only created when connecting clauses and variables and in particular, their left and right boundaries are created by those extensions. Thus, when walking along the boundary of any such cell, we will find the next vertical segment of the variable gadget (or the predecessor in case the segment was the last one). One of the segments must have been selected, so also these cells are covered.

The reverse statement is similar. Assume that we have a cover L' for L of size $n(2m + 1)$. First observe that each variable gadget needs at least $2m + 1$ selected line segments to cover its interior cells. To achieve this we must select either the left or rightmost segment, after which there is a unique cover for the remaining cells that uses only $2m$ segments. Covering the cells within the variable gadgets with fewer than $2m + 1$ segments is not possible, so any cover consist of exactly $2m + 1$ segments per variable gadget. Furthermore, none of these segments can be reused between different variable gadgets. Thus, we conclude that each clause gadget must be covered by the lines selected from the variable gadgets.

We create a variable assignment for each variable as before, depending if the leftmost or rightmost segments has been selected.

Since L' covers all cells, it must also cover the clause cells, which implies that at least one of the three vertical segments has been selected. Equivalently, this implies that in each clause the choice of assignment of the variables makes at least one of its literals true and the formula Φ is satisfiable. \square

Since the reduction is easily computed in polynomial time we conclude the following result.

Theorem 2. *Given a set L of axis-aligned line segments, it is NP-hard to find a minimum-size set $L' \subseteq L$ so that for each cell of the arrangement at least one of its defining segments is in L' .*

3 Covering only rectangular cells

From the above reduction we can see that the main difficulty of the problem lies in covering the rectangular cells. Thus, in this section we turn our attention to a variant of the problem in which segments are axis-aligned and we are not required to cover all cells, but only those that are rectangles. That is, cells whose boundary is formed by exactly four line segments. First we briefly argue that this variant is also NP-hard by adapting the NP-hardness proof in the previous section. Then we show that the problem is fixed parameter tractable (FPT) with respect to k , the number of segments in the optimal solution.

3.1 NP-hardness

The hardness almost follows from the construction of Section 2. Indeed, clause cells are the only critical part of the reduction that need to be modified. Instead, we create the clause gadget with 6 segments as shown in Figure 4. This modified gadget contains three rectangular cells. Note that incoming segments from variables can cover at most two of these cells, but at least one segment must be added so as to cover the intermediate rectangular cell. This additional edge can cover two cells, either the left and middle cells, or the middle and right cells. Thus, it follows that we can find a covering of all rectangular cells of this modified instance with $n(2m + 1) + m$ segments if and only if the associated PLANAR 3SAT instance is satisfiable, and thus this variation is also NP-hard.

Theorem 3. *Given a set L of axis-aligned line segments, it is NP-hard to find a minimum-size set $L' \subseteq L$ so that for each rectangular cell of the arrangement at least one of its defining segments is in L' .*

3.2 FPT on the size of the optimal solution

Next we show that the problem is fixed parameter tractable (FPT) with respect to k , the size of the optimal solution. Our aim is to compute a kernel of small size (or conclude that there is no solution of size at most k).

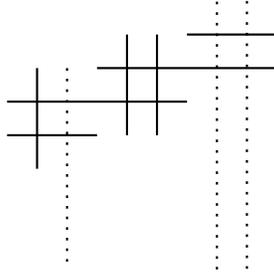


Figure 4: Modified clause gadget. The three dashed vertical segments connect to the corresponding variable gadgets. This new gadget creates three rectangular cells that can be guarded with one additional segment if and only if the variable assignment satisfies the clause.

Since we want to cover only rectangular cells we can represent each cell by an *associated subset* (or subset for short) $C = \{\ell_1, \ell_2, \ell_3, \ell_4\}$ with the four bounding line segments as its elements. This reduces the line segment covering problem to a hitting set problem for a collection \mathcal{C} of subsets of size four. Our approach is to reduce the number of subsets to consider; first to a set \mathcal{C}_1 where for any two line segments there are at most $2k$ subsets that contain both these line segments; then to a set \mathcal{C}_2 where for any single line segment there are at most $2k^2$ subsets containing it. First we prove the following lemma.

Lemma 4. *Let ℓ, ℓ' and ℓ'' be any three line segments in L . There are at most two subsets in \mathcal{C} containing all three line segments, ℓ, ℓ' and ℓ'' .*

Proof. Since the arrangement is rectilinear, two lines, say ℓ and ℓ' are parallel and the other is orthogonal to these. This means that any cell having all three line segments ℓ, ℓ' and ℓ'' on its boundary must span the strip between ℓ and ℓ' . However at most two such cells can also be adjacent to the third line segments ℓ'' . \square

We start reducing our problem instance by looking at pairs of line segments. Specifically we count for every pair of line segments how many subsets contain both. Then for any pair ℓ, ℓ' shared in more than $2k$ subsets we add the subset $\{\ell, \ell'\}$ and remove all subsets containing both ℓ and ℓ' . Let \mathcal{C}_1 denote this reduced subset.

Lemma 5. *A set $L' \subset L$ of line segments, with $|L'| \leq k$ is a minimum-size cover of \mathcal{C}_1 if and only if it is a minimum-size cover of \mathcal{C} .*

Proof. Clearly the claim holds if $\mathcal{C} = \mathcal{C}_1$. Thus, from now on we assume that $\mathcal{C}^+ = \mathcal{C}_1 \setminus \mathcal{C}$ and $\mathcal{C}^- = \mathcal{C} \setminus \mathcal{C}_1$ are two nonempty sets that contain all elements that were added and removed from \mathcal{C} , respectively.

Now assume that L' with $|L'| \leq k$ is a minimum size cover for \mathcal{C}_1 . Observe that all subsets of \mathcal{C}^- must also be covered, since for every subset $C \in \mathcal{C}^-$ there is a subset $\{\ell, \ell'\}$ such that both ℓ and ℓ' occur in C (and $\{\ell, \ell'\} \in \mathcal{C}^+$). It follows

that L' is also a cover for \mathcal{C} . To show that L' is of minimum size assume for a contradiction that a smaller set L'' is also a cover for \mathcal{C} . Clearly, L'' covers $\mathcal{C} \cap \mathcal{C}_1$. Now take any set $\{\ell, \ell'\}$ in \mathcal{C}^+ , if neither ℓ nor ℓ' is part of L'' , then we claim that $|L''| > k$. Indeed, we introduced $\{\ell, \ell'\}$ into \mathcal{C}_1 only when more than $2k$ subsets in \mathcal{C} contain both ℓ and ℓ' . If neither ℓ nor ℓ' are part of L'' , then Lemma 4 implies that every other line can cover at most two of these subsets. In particular, the cardinality of L'' will be larger than k , contradicting with the fact that L'' is smaller than L' . A similar argumentation shows that any minimum-size cover of \mathcal{C} is also a minimum-size cover of \mathcal{C}_1 , which concludes the proof. \square

Now we further reduce our problem instance to a set \mathcal{C}_2 as follows. We count for each line how many subsets of \mathcal{C}_1 contain it. Then for each line ℓ that has more than $2k^2$ subsets containing it we replace all these subsets by subset $\{\ell\}$.

Lemma 6. *A set L' , with $|L'| \leq k$ is a minimum-size cover for \mathcal{C}_1 if and only if it is a minimum-size cover of \mathcal{C}_2 .*

Proof. As before, it suffices to consider the case in which $\mathcal{C}_1 \neq \mathcal{C}_2$. Let $\mathcal{C}_1^+ = \mathcal{C}_2 \setminus \mathcal{C}_1$ and $\mathcal{C}_1^- = \mathcal{C}_1 \setminus \mathcal{C}_2$ denote the sets that were added and removed from \mathcal{C}_1 to create \mathcal{C}_2 . Since all sets in \mathcal{C}_1^- contain a line of one of the singleton sets of \mathcal{C}_1^+ , a set L' , with $|L'| \leq k$ that is a minimum size cover of \mathcal{C}_2 is also a cover of \mathcal{C}_1 . To show that L' is also a minimum-size cover for \mathcal{C}_1 , assume for a contradiction that there is a smaller cover L'' for \mathcal{C}_1 . The lines of L'' also cover $\mathcal{C}_2 \setminus \mathcal{C}_1^+$. Therefore, if L'' is not a cover of \mathcal{C}_2 , then there must be a subset $\{\ell\} \in \mathcal{C}_1^+$ that is not covered by L'' . Recall that $\{\ell\}$ was added to \mathcal{C}_2 because there were more than $2k^2$ sets in \mathcal{C}_1 that contain ℓ . By construction of \mathcal{C}_1 , no line $\ell' \neq \ell$ can cover more than $2k$ of these sets (otherwise, the pair $\{\ell, \ell'\}$ would have been added to \mathcal{C}_1). Thus, we conclude that L'' has more than k segments, a contradiction.

In a similar way we can prove that a subset L' , with $|L'| \leq k$, that is a minimum-size cover for \mathcal{C}_1 is also a minimum size cover for \mathcal{C}_2 , thus, concluding the proof. \square

Lemma 7. *If $|\mathcal{C}_2| > 2k^3$, then there is no cover of size at most k for \mathcal{C} .*

Proof. Proof of this claim follows from Lemmas 5 and 6 and the fact that each segment can only cover at most $2k^2$ sets of \mathcal{C}_2 . \square

Now we look at the computational aspect of generating \mathcal{C}_2 . Both reduction steps from \mathcal{C} to \mathcal{C}_1 and from \mathcal{C}_1 to \mathcal{C}_2 require counting subsets. Here we use the fact that the subsets are of size at most 4 to show that this can be done in linear time using hash tables. Note that linear time here is in the size of \mathcal{C} , the size of the arrangement, which may be quadratic with respect to the number of line segments.

Lemma 8. *The set \mathcal{C}_2 can be constructed in time $O(n \log n + C)$, where n is the number of segments in L and C is the number of cells in the arrangement induced by L .*

Proof. To reduce \mathcal{C} to \mathcal{C}_1 we count for every pair of line segments ℓ, ℓ' the number of subsets of \mathcal{C} that contain both. We do this by making a single pass over all subsets of \mathcal{C} and maintaining for each pair ℓ, ℓ' how many subsets contain them thus far. To avoid having to initialize counts for all pairs ℓ, ℓ' , which may be more than linear in the size of \mathcal{C} we use a hash table and create a new count whenever we encounter a new pair of line segments. Each subset contains at most 4 segments and at most 6 pairs of segments, so we can process it in $O(1)$ time. After counting we can go through all pairs of line segments with non-zero count and for each pair ℓ, ℓ' that is contained in more than $2k$ subsets we remove the sets (which is easily done by storing which sets contain the pair) and add a new subset $\{\ell, \ell'\}$. To compute \mathcal{C}_2 from \mathcal{C}_1 we can use a similar construction. \square

Theorem 9. *For the problem of finding a minimum-size cover for all rectangular cells in an arrangement of n axis-parallel line segments L we can either find a kernel of size $O(k^3)$ or conclude that no solution of at most size k is possible. Moreover, the algorithm runs in $O(n \log n + C)$ time, where C is the number of cells in the arrangement induced by L .*

Since we now have a kernel of size $O(k^3)$ it follows that the problem is fixed parameter tractable [4].

Corollary 10. *Given a set L of line segments the problem of finding a minimum size set $L' \subseteq L$ that covers the rectangular cells of the arrangement of L is fixed parameter tractable with respect to the size k of the optimal solution.*

4 Rectangular subdivisions

Although the problem of finding a minimum set of covering segments is NP-hard, there are special cases where the problem can be solved in polynomial time. One such case is that the input line segments form a special type of rectangular subdivision. The rectangular subdivisions we consider are those defined by a KD-tree [1]. That is, a recursive subdivision of a rectangle using horizontal or vertical splitting segments (see Fig. 5). Note that the segments that have only an endpoint on the boundary of a cell are not considered part of the boundary of that cell. From now on we refer to such a subdivision simply as a *rectangular subdivision*.

A rectangular subdivision provides a clear tree-structure which we can use to compute an optimal covering in a bottom up fashion. Each node in the tree is associated to some rectangle. For a leaf-node this rectangle is a cell of the arrangement, whereas for an interior node its associated rectangle r is formed by the union of the rectangles associated to its children. We also associate an interior node of the tree with a horizontal or vertical segment that splits its associated rectangle into two rectangles associated to its children.

Although the above FPT approach works we show there is a much faster exact algorithm for rectangular subdivision. Without loss of generality we assume

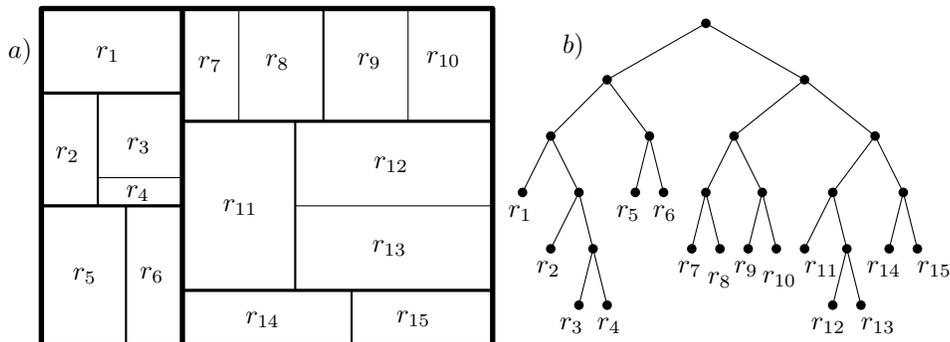


Figure 5: a) A rectilinear binary space partition within a rectangle. b) A possible tree representing the partition.

that the subdivision is given as a binary tree (the KD-tree structure)¹. We show that an optimal covering can then be computed in linear time. Note that in this definition the outer face is covered if and only if at least one of the edges of the bounding rectangle is in the cover, and any other cell is covered if one of the segments defining its boundary is chosen as a covering segment.

Theorem 11. *Given the tree structure of a rectangular subdivision, where each node stores the rectangle it represents and its splitting segment, we can compute in linear time a segment-cover of the cells with a minimum size.*

Proof. Starting at the leaves, we compute an optimal covering in a bottom up fashion. For each node v of the tree we compute the solution to sixteen different subproblems and store these solutions in the corresponding nodes. Let R be the rectangle associated to a given node, and let $\{s_1, \dots, s_4\}$ be the four segments that define its boundary. For any subset $S' \subseteq \{s_1, \dots, s_4\}$, we consider the subproblem of finding the smallest covering of all the cells within R that contains the segments of S' . For each such subproblem its cardinality as well as how it is constructed from solutions of its children (this second part is needed to reconstruct the optimal solution).

Clearly, if v is a leaf the optimal cover is simply S' (unless $S' = \emptyset$ in which case there is no solution). For an interior node we proceed as follows: let v be an interior node of the tree and R the rectangle stored at v . Without loss of generality assume that R is split by a vertical line ℓ . The children of v are v_{left} and v_{right} with corresponding rectangles R_{left} and R_{right} . We must compute a minimum-size cover for each possible choice of the top, left, bottom and right edges of R . Note that a fixed choice of boundary edges for r already forces a choice of three edges for R_{left} and R_{right} . Only their shared edge ℓ is not fixed by the choice of boundary edges of r . However, we can simply try both options and see which results in the overall better cover of R . That is, we first assume ℓ

¹If the structure is not given as a binary tree, but a more general subdivision structure such as a double-connected edge list, we can construct the tree in linear time.

is not part of the cover and retrieve our already computed solutions for R_{left} and R_{right} for the current selection of boundary edges. Then we do the same when we do pick ℓ and choose the solution with the smallest number of edges. This results in an optimal solution since the only edges shared by R_{left} and R_{right} are ℓ and the top and bottom edge of R . We consider all possible selections of these edges. For each selection the two subproblems of finding an optimal cover for the R_{left} and R_{right} are independent, so we can reuse our previously computed solutions.

We now show that the algorithm indeed runs in linear time. Observe that only a constant number of subproblems are considered at each node of the tree. Moreover, each of these subproblems is solved in constant time by accessing the solution to a constant number of subproblems. Thus, overall we spent a constant amount of time per node of the tree, giving the desired bound. \square

5 Conclusions and open problems

Our results show that covering cells in an arrangement, similar to the original set-cover problem, may be NP-hard or polynomial-time solvable depending on various restrictions. It may be interesting to investigate further variants of the problem to see which restriction makes the problem polynomial-time solvable—this may be due to the lack of intersections between line segments or due to the tree-structure of the subdivision. It would also be interesting to see if good approximations are possible for the more general case or even the case with line segments of arbitrary orientations.

References

- [1] M. de Berg, O. Cheong, M. van Kreveld, M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag Berlin Heidelberg, 3rd edition, 2008.
- [2] P. Bose, J. Cardinal, S. Collette, F. Hurtado, M. Korman, S. Langerman and P. Taslakian. Coloring and Guarding Line Arrangements. *Discrete Mathematics & Theoretical Computer Science*, 15(3):139–154, 2013.
- [3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. McGraw-Hill Science/Engineering/Math, 2nd edition, 2001.
- [4] R.G. Downey and M.R. Fellows, *Fundamentals of Parameterized Complexity*. Springer-Verlag, London, 1st edition, 2013.
- [5] T. Feder and D. Greene. Optimal Algorithms for Approximate Clustering. in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC'88)*, pages 434–444, 1988.
- [6] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM* 45(4):634–652, 1998.
- [7] M.R. Gibson. *Clusters and covers: geometric set cover algorithms*. PhD Thesis, University of Iowa, 2010.

- [8] M. Korman, S.-H. Poon, M. Roeloffzen. *Line Segment Covering of Cells in Arrangements*. in *Proceedings of the Twentieth Annual International Conference on Combinatorial Optimization and Applications (COCOA'15)*, pages 152–162, 2015.
- [9] D. Lichtenstein. Planar formulae and their uses, *SIAM Journal on Computing*, 11, 329–343, 1982.
- [10] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing (STOC'97)*, pages 475–484, 1997.