The Structure of Optimal Partitions of Orthogonal Polygons into Fat Rectangles

Joseph O'Rourke* Geetika Tewari†

November 17, 2003

Abstract

Motivated by a VLSI masking problem, we explore partitions of an orthogonal polygon of n vertices into isothetic rectangles that maximize the shortest rectangle side over all rectangles. Thus no rectangle is "thin"; all rectangles are "fat." We show that such partitions have a rich structure, more complex than what one might at first expect. For example, for partitions all "cuts" of which are anchored on the boundary, sometimes cuts are needed $\frac{1}{2}$ or $\frac{1}{3}$ of the distance between two polygon edges, but they are never needed at fractions with a larger denominator. Partitions using cuts without any restrictions seem especially complicated, but we establish a limit on the "depth" of cuts (roughly, how distant from the boundary they "float" in the interior) and other structural constraints that lead to both an $O(n)$ bound on the number of rectangles in an optimal partition, as well as a restriction of the cuts to a polynomial-sized grid. These constraints may be used to develop polynomial-time dynamic programming algorithms for finding optimal partitions under a variety of restrictions.

1 Introduction

VLSI masks are etched by electron beams of some fixed minimum width. Complex shapes can only be masked without unnecessary overexposure if they can be partitioned into rectangles all of which are wider than this minimum width. Thus it is of some interest to develop an algorithm that can find an optimal partition of a polygon into rectangles, in the sense of maximizing the shortest side of any rectangle in the partition.[1]

---

*Dept. Comput. Sci., Smith College, Northampton, MA 01063, USA. `orourke@cs.smith.edu`. Supported by NSF Distinguished Teaching Scholars award DUE-0123154.

†Dept. EECS, Harvard Univ., Cambridge, MA 02138, USA. `gtewari@eecs.harvard.edu`

[1] We thank Ulrich Finkler of IBM Research for suggesting this problem.

Although this was our original motivation, our algorithms are too complex to be of assistance in designing masking strategies. In this paper, we pursue the partitioning problem without regard to practical photolithography concerns. In particular, we study the structure of *optimal partitions* of simple, orthogonal polygons (simple polygons whose edges meet at right angles) into isothetic rectangles (rectangles whose sides are parallel to the polygon edges) satisfying two criteria:

1. The shortest side length δ of any rectangle in the partition is maximized over all rectangle partitions, i.e., there is no strictly "fatter" rectangle partition.
2. Among those partitions with the same δ, an optimal partition must in addition employ the fewest number of rectangles.

We will consistently use δ to represent the shortest side length over all rectangles in a given partition. The first and primary criterion (fatness) in general leaves the optimal partition highly underdetermined, for the presence of one unavoidably thin rectangle in the partition permits considerable freedom in the remaining rectangles as long as they are fatter than δ. The secondary criterion (fewest number) controls this somewhat, but still leaves the optimal partition underdetermined: in general there are many equally optimal partitions. Thus it is not possible to tightly characterize all optimal partitions. But the theme of our work is to show that there is always some optimal partition with particular structural properties, structural properties that lead to algorithms.

In two abstracts [OPT01, OT02] we claimed polynomial-time algorithms for finding optimal partitions under a variety of restrictions (including no restrictions). Although we repeat these claims here, we do not formally establish them, for the following reasons. First, the time complexities are very large and unlikely to be anywhere near optimal. Second, the algorithms are all complex dynamic programming algorithms. Third, careful proofs of correctness for the algorithms would require going well beyond our abstracts and the supporting details in student reports [Pas01, Tew02]. Instead we concentrate on the structural properties of optimal partitions, in the hope that they will lead to cleaner and more efficient algorithms.

1.1 Related Work

The two main optimization criteria that have been explored in the problem of partitioning orthogonal polygons are: minimizing the number of rectangles, and minimizing the total length of the cut segments needed to cut out the rectangles. Improving on several earlier results, Liou et al. found an $O(n)$ time algorithm to optimally partition an orthogonal polygon without holes into the minimum number of rectangles [LTL89]. Lingas et al. were the first to investigate the second, "minimum ink" optimization criterion. They presented a $O(n^4)$ time algorithm for optimally partitioning an orthogonal polygon without holes into rectangles [LPRS82]. For a polygon with holes, they showed that the problem is NP-complete.

There has also been research on covering orthogonal polygons by rectangles, but this work is less relevant to our concerns. See [Kei00] for a survey of polygon decomposition algorithms.

2 Cut Types and Results

Our analysis focuses on the cuts used to separate the rectangles in a partition. Roughly, a *cut* is a maximal segment of the partition whose relative interior is strictly interior to the polygon; see Section 3 for a more precise definition. It is natural to wonder if the cuts may be restricted in any way without altering the optimum δ of partitions. For example: Is there always an optimal partition such that at least one cut connects two boundary points, i.e., cuts all the way through? Fig. 1 shows that the answer is NO. Let us distinguish three types of cuts:

1. *Vertex cuts*: those incident to a polygon vertex.
2. *Anchored cuts*: those touching (anchored on) a point of the polygon's boundary. (Vertex cuts are special cases of anchored cuts.)
3. *Floating cuts*: those which are strictly interior to the polygon—they "float" in the interior.

(Later it will be useful to call all nonvertex cuts *movable* cuts.)

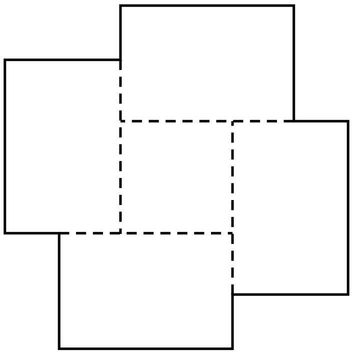


Figure 1: No cut in this optimal partition connects boundary-to-boundary.

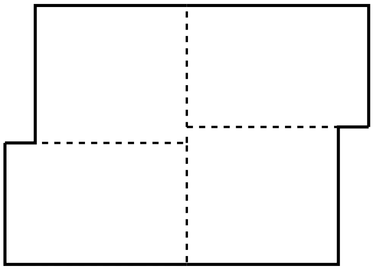


Figure 2: Not all cuts in the optimal partition are vertex cuts: the central cut is anchored but not incident to a vertex.

Is there always an optimal partition using only vertex cuts? Fig. 1 uses only vertex cuts, but Fig. 2 employs (necessarily) one cut that includes no vertex. Is there always an optimal partition with every cut lying on the *vertex grid* formed

Cut type	Time complexity	Reference
vertex cuts	$O(n^5)$	[OPT01]
anchored cuts	$O(n^{10})$	[OT02, Tew02]
unrestricted cuts	$O(n^{42})$	[OT02, Tew02]

Table 1: Dynamic programming algorithm time complexities.

by all horizontal and vertical lines through all polygon vertices? Fig. 2 again provides a counterexample. Could we say that no reflex vertex ever need be incident to two cuts? Fig. 3 shows this is too strong a claim. Is there always an optimal partition such that every cut is anchored? Figures 1-3 use only anchored cuts, but Fig. 4 employs (necessarily) one floating cut.

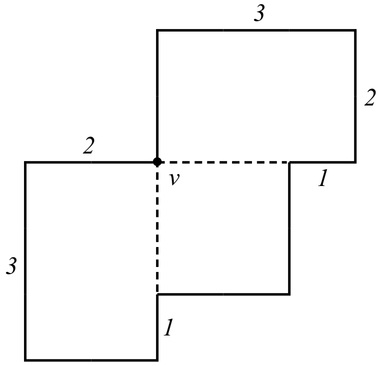


Figure 3: Two cuts incident to a reflex vertex v in the optimal partition. Edges are labeled with their lengths.

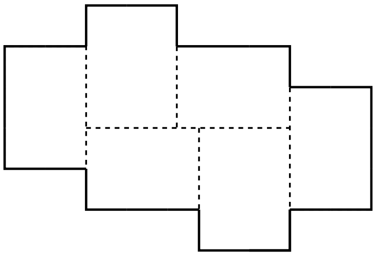


Figure 4: Not all cuts in the optimal partition are anchored: the central cut is floating.

Other optimality criteria permit some of these restrictions. For example, the minimum number of rectangles can be achieved via vertex cuts [LTL89]; and the minimum total cut length can be achieved by anchored cuts lying on the vertex grid [LPRS82]. To find optimal partitions independent of restrictions, we must include floating cuts. Nevertheless, it is interesting to explore optimal partitions under the restriction that all cuts are vertex cuts, or all are anchored cuts. In [OPT01] we showed that restriction to vertex cuts permits a dynamic programming algorithm similar to that in [LTL89] to find an optimal partition in $O(n^5)$ time. We implemented the algorithm, and discovered that, on random orthogonal polygons, it appears to run in $O(n^2)$ time. A sample of the code's output is shown in Fig. 5. In [OT02] more complex structural constraints led to increasingly worse but still polynomial-time algorithms for finding optimal partitions, as summarized in Table 1. All three algorithms follow a similar dynamic-programming structure, although we could only handle floating cuts

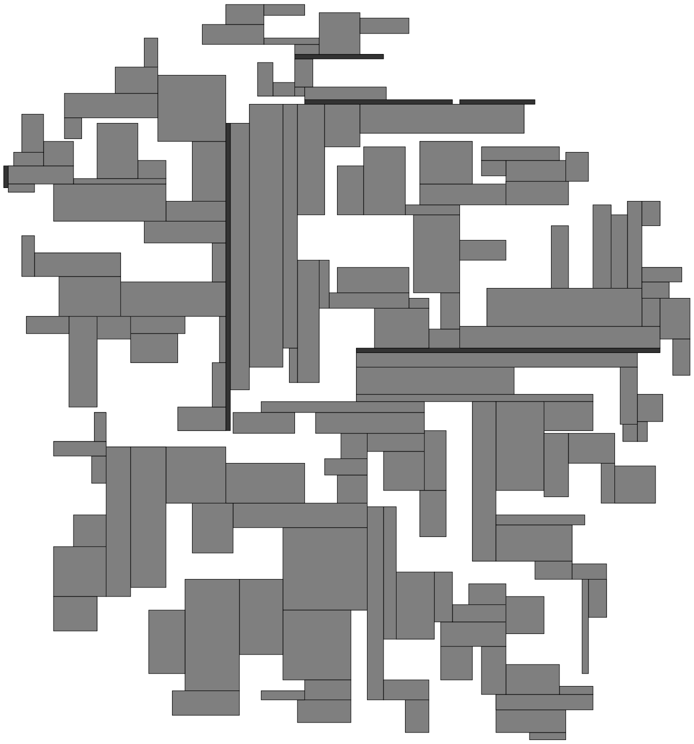


Figure 5: A polygon of $n = 348$ vertices optimally partitioned by vertex cuts into 155 rectangles. The tied-for-thinnest rectangles are shaded dark.

(unrestricted cuts) with an intricate algorithm [OT02, Tew02] that leads to the large upper bound indicated.

The support for these algorithms is a collection of geometric and combinatorial theorems on the structure of optimal partitions. Vertex cuts are already restricted to lie on the vertex grid. We show in Theorem 6 that anchored cuts fall on the vertex grid, or on midlines or thirds-lines between the lines of the vertex grid (but not fourths-, etc.). Our most complex structural result is that an (unrestricted) optimal partition never includes floating cuts that are too "deep" in the polygon interior (Theorem 12): cuts floating deep in a "sea" of floating cuts are never necessary. This leads to an $O(n)$ bound on the number of rectangles in an optimal partition (Theorem 14), and a restriction of floating cuts to lie on a grid of $O(n^4)$ lines (Theorem 17). As a consequence, we obtain a polynomial-size set of candidate cuts from which an optimal partition can be selected.

3 Cut Properties

Vertex cuts are already so restricted that little additional structure is needed for an algorithm. However, there are two issues concerning the definition of a cut that need clarification. Let S be the set of points of a rectangle partition of a polygon P that (a) lie on some rectangle boundary, and (b) are strictly interior to P. We define a set C of *cuts* for the partition to be a collection of closed segments satisfying these properties:

1. The union of all cuts in C is equal to the closure of S.
2. The relative interiors of the cuts (i.e., the cuts without their endpoints) are pairwise disjoint.
3. The cuts are maximal in the sense that no two collinear cuts in C can be merged without violating pairwise disjointness.

Criterion (2) rules out two cuts crossing by definition, for if they cross, they share an interior point. Thus, the partition shown in Fig. 6(a) is not a partition by vertex cuts, for the two boundary-to-boundary segments incident to a and b cross. This partition must be viewed as employing at least one anchored cut, say, the right half of the segment incident to a. The optimal partition using only vertex cuts results in a thinner rectangle, as shown in in Fig. 6(b). We defined cuts to be maximal and thus noncrossing because permitting them to cross introduces nonlocal effects that undermine the dynamic programming algorithm.

The second issue concerning the definition of cuts is that a given partition does not uniquely determine a set of cuts, due to criterion (3). Nonuniqueness occurs at +-junctions, such as at the intersection of the b- and c-cuts in Fig. 6(b). In this case, the b-cut must go through the intersection, leaving a c- and d-cut to either side, in order to remain a partition by vertex cuts. But in general there is a choice, e.g., if there were a vertex at the other end of the b-cut, or if the

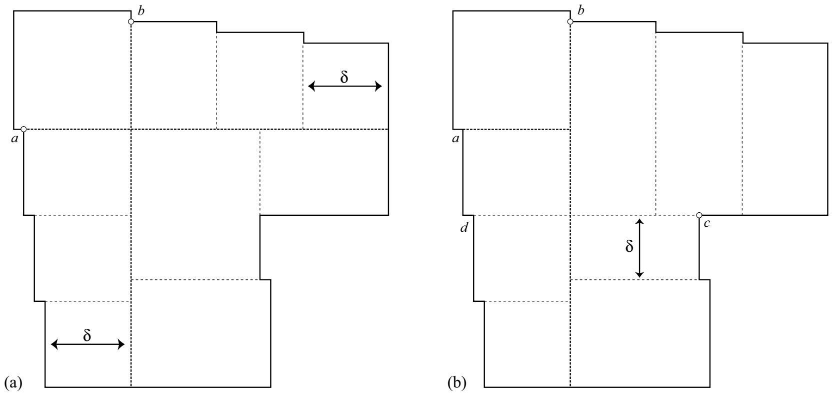


Figure 6: (a) The segments incident to a and b cross, so this is not a vertex-cut partition. (b) The optimal vertex-cut partition.

cuts were not restricted to vertex cuts. To facilitate later reference, we elevate this simple consequence of our definition to a lemma:

Lemma 1 *An endpoint of a cut is either at a vertex, on an edge of the polygon, or at an interior point of another cut.*
Proof: If a cut has an endpoint interior to the polygon but not at an interior point of another cut then, as Fig. 7(a) shows, there must be at least one other cut incident to the endpoint in order to surround that point by rectangles; but, as (b) shows, this would result in (at least) two collinear cuts, which violates the maximality of the definition of a cut. In the case illustrated, either $c_1 \cup c_2$ or $c_3 \cup c_4$ could be considered one cut, in which case the other two would terminate on an interior point of that cut. $\square$

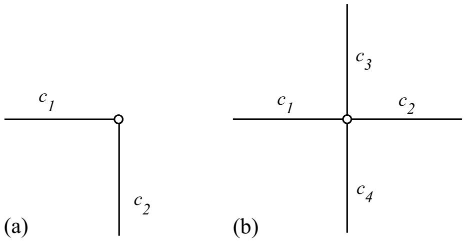


Figure 7: (a) Two such cuts cannot share an endpoint interior to the polygon; (b) Such collinear cuts are not maximal.

4 Anchored Cuts

The example in Fig. 2 suggests that anchored cuts can be chosen to lie on lines midway between two edges, for it would seem to be advantageous to slide such a cut to balance the dimensions of the rectangles supported to either side. We were therefore surprised to discover the example shown in Fig. 8, which has the property that the optimal partition needs to use two anchored cuts which are not midway between any pair of edges, but rather lie at one-third and two-thirds between two edges. (We will not pause to prove that the partition shown is the only optimal one [Tew02, pp. 22–25].) Call a partition that uses only anchored cuts an *anchored partition*. Call a rectangle in an optimal partition a *δ-rectangle* if at least one of its dimensions is δ, the minimum rectangle side length throughout the partition. Say that a cut c *supports* a rectangle R if one of R's sides shares a positive-length portion of c.

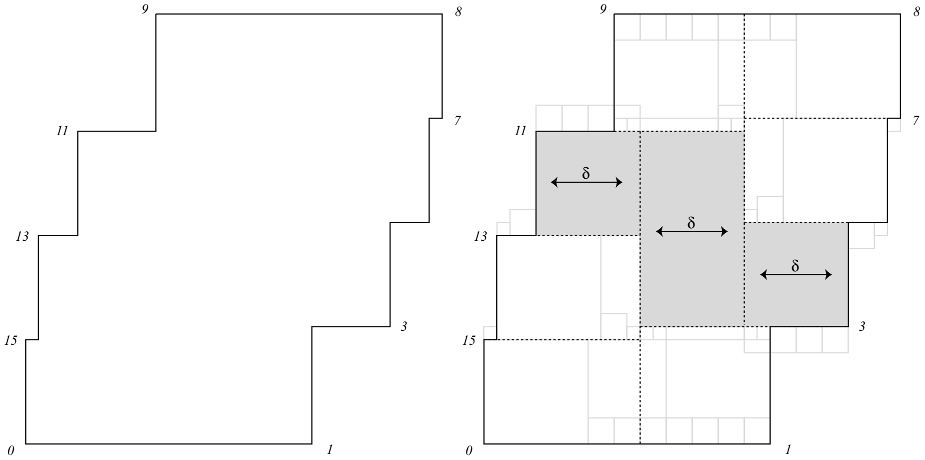


Figure 8: An $n = 16$ orthogonal polygon and its unique optimal partition. The three shaded rectangles are $\delta = 8$ wide. Light lines indicate grid dimensions. The two vertical anchored cuts are not midway between any pair of vertices, but lie $\frac{1}{3}$ and $\frac{2}{3}$'s between vertices 3 and 11.

We define a segment a to *hit* a segment b if an endpoint of a coincides with a nonendpoint of b. Note that this definition is not symmetric. For the symmetric notion, we use the term *touch* (or *incident*): two segments touch if they share a point.

We need to distinguish the anchored cuts that are not vertex cuts. Define all nonvertex cuts as *movable* cuts; these are the cuts that are "potentially movable."

The following lemma will show that the situation in Fig. 8 is in a sense the worst that can happen with movable anchored cuts:

Lemma 2 *No optimal anchored partition includes three parallel movable cuts supporting two δ-rectangles between them.*

If this lemma were false, then anchored cuts might need lie on, say, quarter-lines. But with only two parallel anchored cuts, a "$\delta + \delta + \delta$" configuration as illustrated in Fig. 8 is the most complex possible. This leads to the algorithm listed in Table 1, as we will show at the end of this section.

Because anchored partitions are of less intrinsic interest than unrestricted partitions, we choose to only sketch the proof of Lemma 2, via Lemmas 3 and 4 below, which we will apply again in Section 5.

Lemma 3 *A movable cut f in an optimal partition must have two cuts hitting it from opposite sides that are separated by strictly less than δ. We call such cuts* ($< \delta$)-staggered*.*

Proof: Suppose otherwise. Then remove cut f, assumed vertical without loss of generality. The rectangles hitting each side of f lengthen horizontally but may shorten each other vertically as their horizontal top and bottom cut through one another; see Fig. 9. Note that because f is movable, neither endpoint is a vertex, so the horizontal lengthening does not introduce any new rectangles. However, because any pair of cuts hitting f are separated by $\geq \delta$, this "slicing up" cannot produce a side length $< \delta$. Therefore the new partition does not diminish δ. It reduces the number of rectangles by 1. Therefore the original was not optimal, a contradiction. □

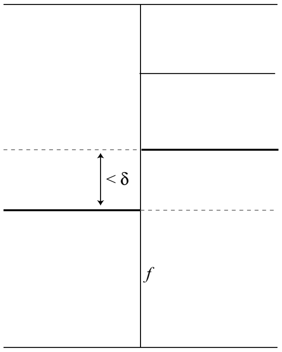
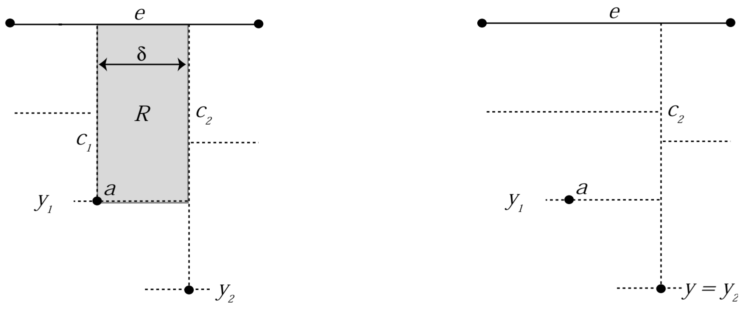


Figure 9: Opposing cuts hitting f must be staggered by $< \delta$. Removal of f replaces 5 rectangles by 4.

Lemma 4 *An optimal partition does not include a δ-rectangle supported on opposite sides by parallel δ-separated movable anchored cuts (i.e., anchored but nonvertex cuts) which both hit the boundary at their endpoints to the same side.*

Proof: *(Sketch.)* Let c_1 and c_2 be the movable anchored cuts, oriented vertically without loss of generality. Suppose in contradiction to the lemma, the cuts c_1 and c_2 terminate on edges above (i.e., to the same side). They can terminate on two distinct noncollinear edges, or on two distinct collinear edges, or on the same edge e. In this sketch we only prove the last case: Both c_1 and c_2 terminate in the same edge e above.

Let $y_1 \geq y_2$ be the y-coordinates of the lower endpoints of c_1 and c_2; see Fig. 10(a). Let a be the lower endpoint of c_1. It must be true that there are no vertices of the polygon on the horizontal closed segment between a and c_2. For if there were this vertex, it would squeeze in a rectangle of width $< \delta$ between it and c_2. Therefore c_1 is not supported by hitting ($< \delta$)-staggered cuts, as it can have no such cut from its right. This violates Lemma 3, as it would lead to the improved partition in Fig. 10(b). Therefore the assumed partition could not have been optimal, a contradiction.



(a) (b)

Figure 10: The partition is improved if c_1 is removed.

The other two cases lead to similar contradictions from the presence of a vertex that forces a thinner rectangle. ☐

We are now prepared to sketch a proof of Lemma 2, ruling out more than two parallel δ-separated movable anchored cuts.

Proof: *(Sketch.)* In an optimal partition, if parallel δ-separated anchored cuts exist, then:

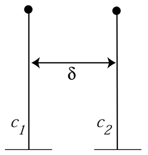


Figure 11: Contradiction to Lemma 4

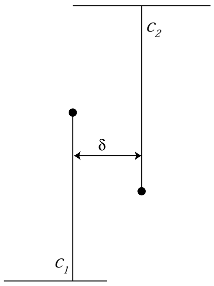


Figure 12: Two adjacent anchored cuts.

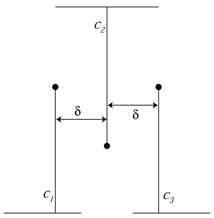


Figure 13: Three adjacent anchored cuts.

1. Each movable anchored cut must be supported by ($< \delta$)-staggered cuts hitting it, as per Lemma 3.

2. Lemma 4 excludes the situation shown in Fig. 11. So they must be aligned as shown in Fig. 12.

Therefore the structure of three parallel adjacent movable anchored cuts must be as shown in Fig. 13. Suppose, contrary to the lemma, that an optimal partition has three vertical parallel δ-separated movable anchored cuts. Let the three cuts c_i have upper and lower endpoints a_i and b_i respectively, and left and right staggered cuts l_i and r_i, for $i = 1, 2, 3$ hitting them. Without loss of generality let b_1 and b_3 hit the polygon boundary; then by the reasoning above, b_2 does not hit the boundary, and thus must hit another cut, d_2. Let $y(p)$ be the (vertical) y-coordinate of a point p. Without loss of generality let $y(b_1) \leq y(b_3)$. In this sketch we consider only one of three cases depending on the relation of $y(b_2)$ with respect to $[y(b_1), y(b_3)]$: $y(b_1) \leq y(b_3) \leq y(b_2)$ (See Fig. 14.)

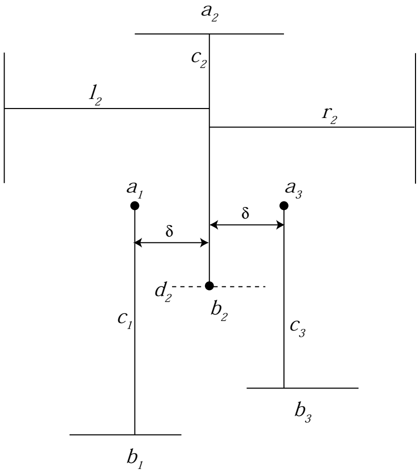


Figure 14: Case: $y(b_3) < y(b_2)$.

Consider the left staggered cut l_3 supporting c_3. Its left end must lie on the polygon boundary (because it is anchored). But c_3 must project leftwards onto c_2 (within the range $[y(b_2), y(a_3)]$ or c_1 (within the range $[y(b_3), y(b_2)]$ without any intervening boundary points. For suppose there were a boundary point p in this range, such that the horizontal segment from p to c_3 is interior to the polygon. Then p must fall somewhere horizontally between c_1 and c_3. It cannot lie on c_1 or c_2. Two subcases arise:

1. p is not collinear with c_2. Then p lies strictly between c_1 and c_2, or c_2 and c_3. This forces a subdivision of that δ-column, resulting in a rectangle partition $< \delta$, a contradiction.

2. p is collinear with c_2. Then the polygon boundary must enter the δ-column to the left or right somewhere between p and d_2, again forcing a subdivision of that column, resulting in the same contradiction.

The other cases we are not including here ([Tew02, pp.28–33]) lead to similar contradictions from forced splittings of δ-columns. □

Compacting the rectangles in a partition leftwards leads to this:

Lemma 5 *There exists an optimal partition such that every (vertical) movable anchored cut supports a δ-rectangle to its left.*
Proof: *(Sketch).* Otherwise, an unsupported cut could be slid to the left with no resulting change in the optimality of the partition. □

Finally, the following theorem is now straightforward:

Theorem 6 *There exists an optimal anchored partition such that every movable anchored cut has coordinate:*

$$\tfrac{1}{2}a + \tfrac{1}{2}b \quad \text{or} \quad \tfrac{1}{3}a + \tfrac{2}{3}b$$

where a and b are (either x or y) coordinates of two vertices.
Proof: The proof proceeds by separately examining the x-coordinates of one, two, and three or more parallel movable anchored cuts in an optimal partition (The argument is identical for y-coordinates). Let $x(p)$ be the x-coordinate of a point p, and $x(s)$ the x-coordinate of a vertical segment s.

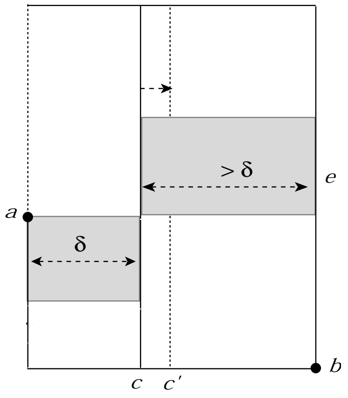


Figure 15: Case 1: c can be moved to a midline c'.

1. A single movable anchored cut.

   Without loss of generality, let a polygon edge e be to the right of a vertical movable anchored cut c, as in Fig. 15. By Lemma 5, c must support a δ-rectangle to its left that is bounded by a polygon edge or vertex cut. Therefore $x(c) - \delta$ must equal $x(a)$ for some vertex a. Let a vertex b lie on e. Let the x-coordinate of an anchored cut c' be:

   $$x(c') = \frac{1}{2}x(a) + \frac{1}{2}x(b)$$

   c can be replaced by c', the midline of the horizontal distance between a and b, without any change in the optimal partition.

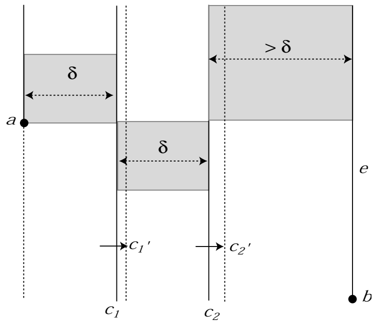


Figure 16: Case 2: c_1 and c_2 can be moved to c_1' and c_2' that lie on third lines.

2. Two parallel δ-separated movable anchored cuts.

   Let c_1 and c_2 be two movable anchored cuts, and let e be the polygon edge to the right of c_2. As established by Lemma 5, c_1 must support a δ-rectangle to its left that is bounded by a polygon edge or vertex cut. Therefore $x(c_1) - \delta = x(a)$ for a vertex a. Let a vertex b lie on e. Let a movable anchored cut c_1' satisfy

   $$x(c_1') = [x(a) + 2x(b)]/3 = \frac{1}{3}x(a) + \frac{2}{3}x(b)$$

   Let a movable anchored cut c_2' satisfy

   $$x(c_2') = [2x(a) + x(b)]/3 = \frac{2}{3}x(a) + \frac{1}{3}x(b)$$

   c_1 and c_2 can be replaced by c_1' and c_2', which lie on third-lines between a and b, without any change in the optimality of the partition. See Fig. 16.

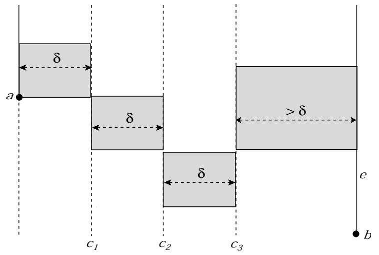


Figure 17: Case 3: Contradiction to Lemma 2.

3. Three or more parallel δ-separated movable anchored cuts

   This is a contradiction to Lemma 2. See Fig. 17.

□

Theorem 6 leads to an $O(n^2)$ set of possible anchor points along the boundary of the polygon: at all the vertices, and at all the $\{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}\}$ coordinates between each pair of edges. Treating each anchor point as a "pseudovertex," and applying the $O(n^5)$ vertex-cut algorithm [OPT01], yields the $O(n^{10})$-time algorithm in Table 1.

5 Unrestricted Cuts

Lemma 2 led us to wonder if a similar result might not hold for floating cuts. Fig. 18 shows that no easy generalization is possible.

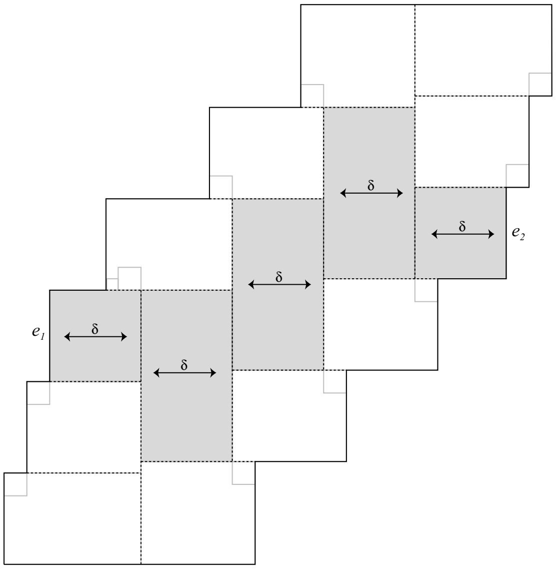


Figure 18: The unique optimal partition employs four vertical cuts at multiples of one-fifth between polygon edges e_1 and e_2. The shaded rectangles are each δ wide.

Here the positioning of four cuts—two anchored and two floating—is determined by a chain of five δ-rectangles. Generalizing the example shows that $1/k$-th lines might be necessary for any $k = O(n)$.

Although in such an example, the position of some central cuts is determined by distant edges, none of these cuts is "far" from the boundary in another sense. To make this sense clear, we introduce the notion of the *depth* of a cut. Cut depth is defined inductively:

1. Vertex cuts have depth 0. Polygon edges are also defined to have depth 0, being viewed as cuts that touch their vertex endpoints.
2. Otherwise, a cut has depth 1 plus the minimum depth over all segments that touch it.

Thus, every point on a cut of depth k can reach a vertex by a "cut path" with no more than k turns, a fact that is used centrally in the dynamic programming algorithm [OT02]. Movable (nonvertex) anchored cuts have depth 1, because they touch a polygon edge. Floating cuts have depth ≥ 1. See Fig. 19. Notice that cut depth does not distinguish between anchored and floating, because a floating cut could have both *endcaps* (the cuts it hits, i.e., on which its endpoints lie) as vertex cuts, i.e., depth-0 cuts. All cuts of depth ≥ 1 are movable cuts, for they, unlike vertex cuts, might slide perpendicularly without changing the structure of the partition.

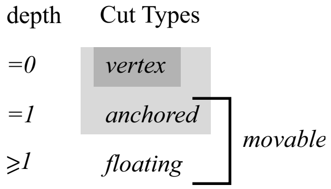


Figure 19: Classification of cut types.

We already know from Figures 4 and 18 that cuts of depth 1 are sometimes necessary, for those examples require floating cuts, which are always depth ≥ 1. It is, however, not entirely obvious that cuts of depth 2 (or larger) are ever needed. The simplest example we could find is shown in Fig. 20, which requires a single cut f of depth 2. (That the partition shown is the unique optimal one is not evident, established in [Tew02, pp.43–51].) In order to achieve a polynomial-time algorithm, we found it necessary to derive a limit on how deep into the partition the constraints of the polygon boundary can propagate. The remainder of our effort is focussed on establishing that cuts of depth 3 or more are never needed in an optimal partition. This will lead to a polynomial-sized grid on which the cuts may be placed.

5.1 Movable Cuts Structure

The plan is to detail what must be the local structure of an optimal partition surrounding a cut of vertex-depth 3 (or more), and then show that this structure is in fact not optimal: repartitioning will improve it. Let f be a cut of depth ≥ 3. The endcaps l and r of f, and all the cuts that hit f, must be of depth ≥ 2. Moreover, all the cuts touching these hitting cuts must be of depth ≥ 1 and therefore movable. We now show that such collections of movable cuts in an optimal partition must have a very restricted structure. The intuition is that too much of the structure is movable to be essential, thus eliminating possibilities.

We start with a simple constraint on interior rectangles.

Lemma 7 *In an optimal partition, every rectangle that is bounded by two parallel movable cuts, is bounded by the cuts in a "pinwheel" pattern (in either orientation), as illustrated in Fig. 21(a).*

This structure is evident in both Figs. 8 and 18.

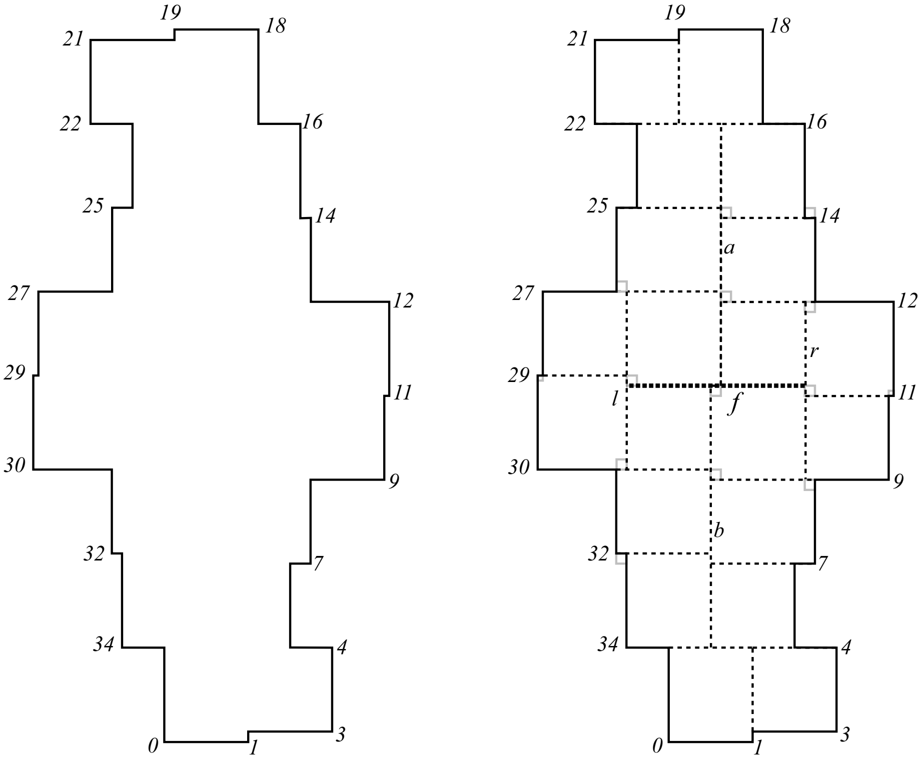


Figure 20: A polygon that requires a floating cut f of depth 2. Cuts l, r, a, and b are all depth-1, as they touch vertex cuts (which have depth 0).

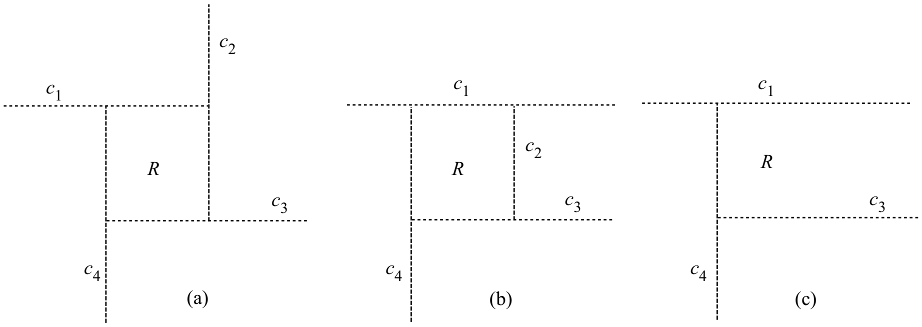


Figure 21: (a) A pinwheel of cuts surrounding rectangle R. (b) A non-pinwheel pattern leads to an improved partition (c).

Proof: As established by Lemma 1, two cuts must meet in a 'T', one cut hits the other. Let R be bounded by c_i, $i = 1, 2, 3, 4$, with c_1 and c_3 the two movable cuts. The intuition is that, because they are movable, they each need to be "stopped" by hitting cuts, to require R. By Lemma 3, c_3 must have a cut c_2 hitting it from above. If, contrary to the lemma, c_2 also hits c_1, then, as shown in Fig. 21(b), removal of c_2 improves the partition (c), so it could not have been optimal. □

Now we start to build up the constraints on deep movable cuts.

Lemma 8 *If a movable cut has only movable cuts hitting it from one side, then it has exactly one such cut hitting that side.*

Proof: Let f be the "base" movable cut, horizontal without loss of generality. Lemma 3 establishes that f must have at least one cut hitting it on each side. In contradiction to the lemma, assume f has two adjacent movable cuts c_1 and c_2 hitting f from above. As established by Lemma 3, each movable cut must be supported by hitting ($< \delta$)-staggered cuts. Thus, c_1 must have a cut hitting it on its right; let a be the lowest such. Finally, let R_i be the rectangle covering f from above and c_i from the right, for $i = 1, 2$. See Fig. 22(a).

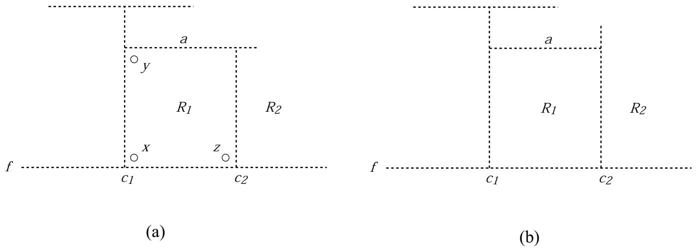


Figure 22: Two movable cuts c_1 and c_2 hitting another movable cut f.

Let x be a point near the lower left corner of R_1. Because a is lowest, R_1 extends vertically from f to a, including a point y near the intersection of c_1 and a. Because c_1 and c_2 are adjacent, R_1 extends horizontally along f to c_2, including a point z near the c_2-f intersection. Because R_1 includes $\{x, y, z\}$, the cut a, which includes the top of R_1, must extend rightwards at least to c_2. Thus we know that a meets c_2.

As a result of Lemma 1 there are now only two possibilities. We show that that each is contradictory:

1. c_2 hits a; see Fig. 22(a). Then the cuts forming R_1 are not aligned in a pinwheel pattern, violating Lemma 7.
2. a hits c_2; see Fig. 22(b). Again the pinwheel lemma is violated.

These contradictions establish the lemma. □

The structure at the heart of Fig. 20 is necessary:

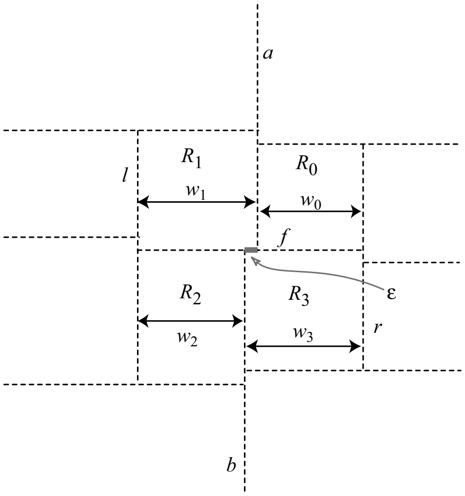


Figure 23: Structure of a movable cut f with all touching cuts movable.

Lemma 9 *In any optimal partition, every movable cut f that touches only movable cuts, has the structure indicated in Fig. 23: there are just two hitting cuts a and b, staggered on f by less than δ, and the endcaps l and r terminate on cuts as illustrated. Additionlly, the partition must, without loss of generality, have the following dimensional constraints, where $\epsilon = w_1 - w_2 = w_3 - w_0$:*

$$w_1 > w_2 \geq \delta$$

$$w_3 > w_0 \geq \delta$$

with

$$0 < \epsilon < \delta$$

Proof: Lemma 3 establishes that a movable cut f must be supported by hitting cuts staggered by $< \delta$. If these are all movable cuts, then by Lemma 8, f must have just one on each side (a and b). Both endcaps l and r must hit the nearest cuts supporting a and b; otherwise, the pinwheel Lemma 7 is violated.

We now establish the dimension constraints. Assume without loss of generality that $w_2 \leq w_1 \geq w_0$. (For if $w_2 > w_1$, reflect the figure about f, and relabel. If $w_1 < w_0$, reflect the figure about a and relabel.) If $w_1 = w_2$, then a and b merge and the left half of f is unnecessary. Therefore, the structure of f must be as illustrated in Fig. 23, and must have the following properties:

$$w_1 > w_2 \geq \delta$$

$$w_3 > w_0 \geq \delta$$

Let $\delta = \min\{w_i\}$, $i = 0, \ldots, 3$. If δ were less than ϵ, then f would not be supported by $(< \delta)$-staggered cuts; a violation of Lemma 3, hence, $0 < \epsilon < \delta$.$\Box$

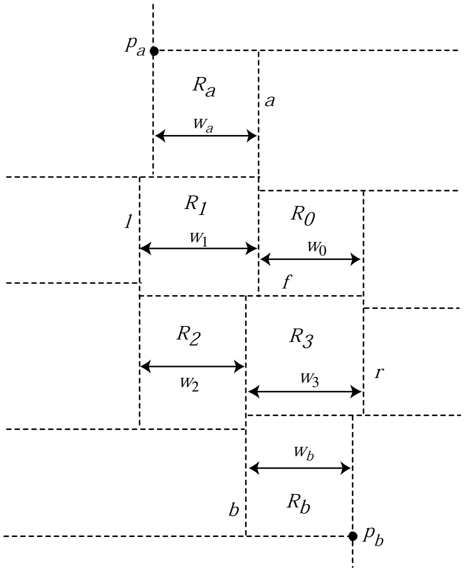


Figure 24: The necessary structure surrounding a cut f of depth ≥ 3 in an optimal partition.

5.2 Depth-3 Cut Structure

We now specialize the situation to depth-3 cuts. The key to the construction is that the cuts a and b just illustrated in Fig. 23 are themselves subject to the constraints of Lemma 9.

Lemma 10 *The structure of a cut f of depth ≥ 3 in an optimal partition must be as shown in Fig. 24:*

1. *Only one cut hits a and b on both sides.*
2. *$w_a \leq w_1$ and $w_b \leq w_3$.*

Proof: A depth-3 cut f is movable and touches only movable cuts, so Lemma 9 applies and establishes the local structure around f (Fig. 23). Because a and b are themselves movable, Lemma 8 shows they each can have only be hit by one movable cut to each side. Therefore, there can only be two rectangles to each side of a and b, as illustrated.

Suppose $w_a \geq w_1$ in the optimal partition, in contradiction to the claim of the lemma. Then, as illustrated in Fig. 25, a could slide to the left by ϵ and merge with b. As a result of this sliding, the width of the rectangles incident to a from the right is increased by ϵ, and the width of the rectangles incident to a from the left is reduced by ϵ, but is $\geq w_2$ (which is unchanged), and thus $\geq \delta$. Now a, b, and f could be removed and replaced by a single vertical floating cut f', reducing the number of rectangles in the original partition, and providing

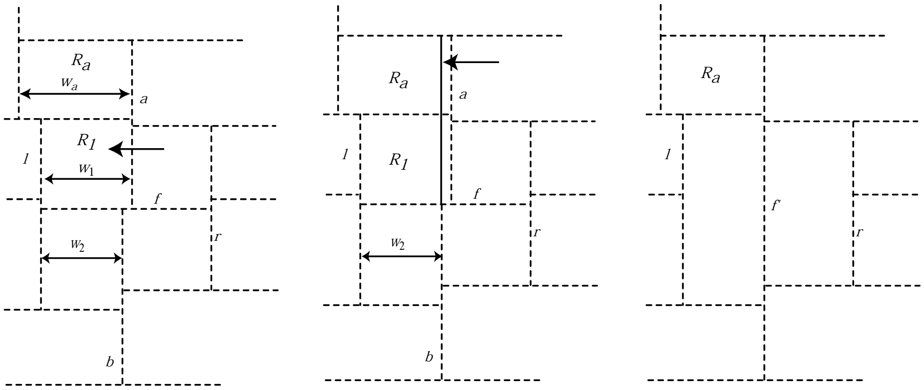


Figure 25: If $w_a \geq w_1$, then the partition is not optimal.

a superior partition, contradicting the assumed optimality of the partition. An analogous contradiction arises if $w_b \geq w_3$. Therefore, $w_a < w_1$ and $w_b < w_3$.

Therefore, the structure of a movable cut f of depth ≥ 3 in an optimal partition must be as shown in Fig. 24. □

5.3 Repartitioning Lemma

Finally, we show that the necessary structure surrounding a depth-(≥ 3) cut just established is suboptimal.

Lemma 11 *An optimal partition never includes the structure shown in Fig. 24, with* $w_a \leq w_1$ *and* $w_b \leq w_3$*, for it can always be improved from Fig. 26(a) to (c).*

Proof: We rely on Fig. 26 to define p_a, c_a, s_a, p_b, c_b, and s_b. The cut c_a can be vertically extended down from p_a until it meets s_b, the bottom horizontal side of R_2. Similarly, a cut c_b can be vertically extended upward from p_b until it meets s_a, the top horizontal side of R_0. s_a can be horizontally extended leftward until it meets c_a, and s_b can be horizontally extended rightward until it meets c_b. This provides a new partition, where the four rectangles covering f can be replaced by a single rectangle R bounded by $\{s_a,\ c_a,\ s_b,\ c_a\}$. See Fig. 26. The width of R, w_R, is:

$$w_R = w_a + w_b - \epsilon$$

Because $w_a \geq \delta$ and $w_b \geq \delta$,

$$\begin{aligned} w_R &\geq 2\delta - \epsilon \\ w_R &\geq \delta + (\delta - \epsilon) \end{aligned}$$

From Lemma 9, $\delta > \epsilon$, so it follows that $w_R > \delta$, and R is an improvement.

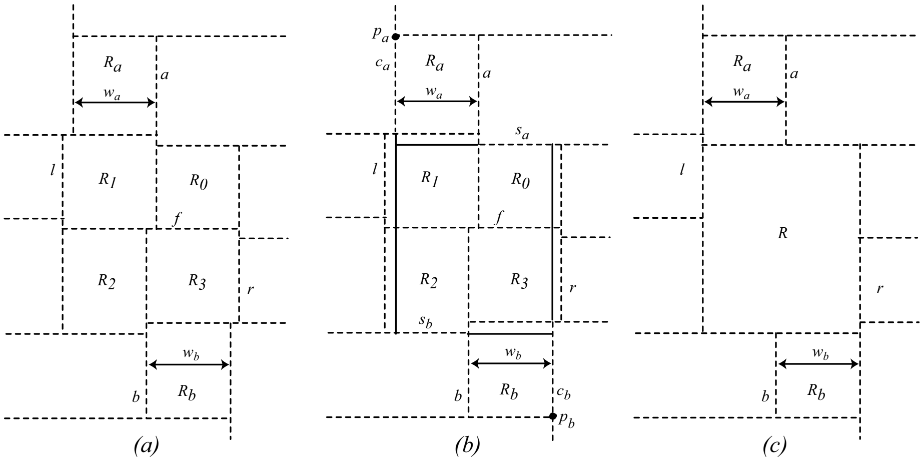


Figure 26: The structure in (a) can be repartitioned as indicated in (b) to the improved structure shown in (c).

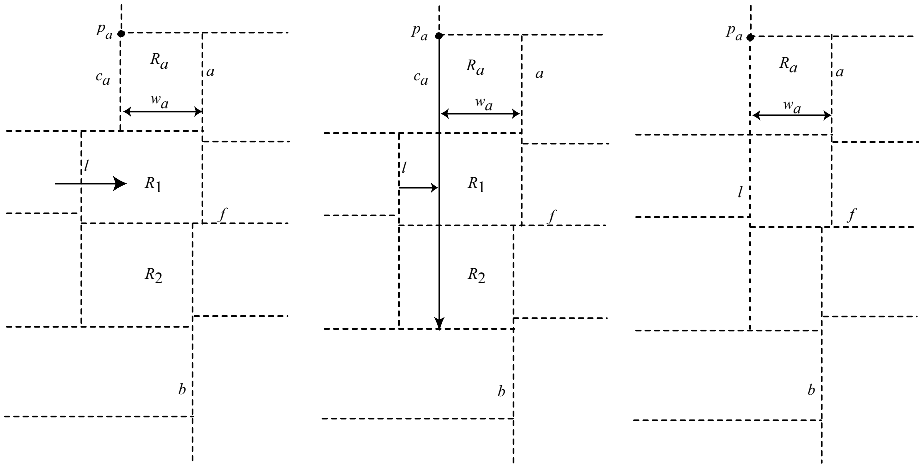


Figure 27: Extending a vertical cut (such as c_a) either increases or leaves the width of incident rectangles (to the vertical cut) unchanged.

We now verify that the illustrated partition is indeed an improvement, by inspecting the effect of each extended cut on the dimensions of the surrounding rectangles.

Because $w_a \leq w_1$ and $w_1 > w_2$, the vertical extension of c_a must cut across the width of two rectangles; R_1 and R_2. After the repartitioning, these rectangles are replaced by R. Since an endcap of f is a movable cut, it can slide horizontally toward f and merge with the closer of c_a or c_b. Rectangles supported by the endcap on the side that faces f are replaced by R, and rectangles supported on the other side become larger. The width of rectangles incident to R_a, from whose corner c_a is extended, remains unchanged. See Fig. 27.

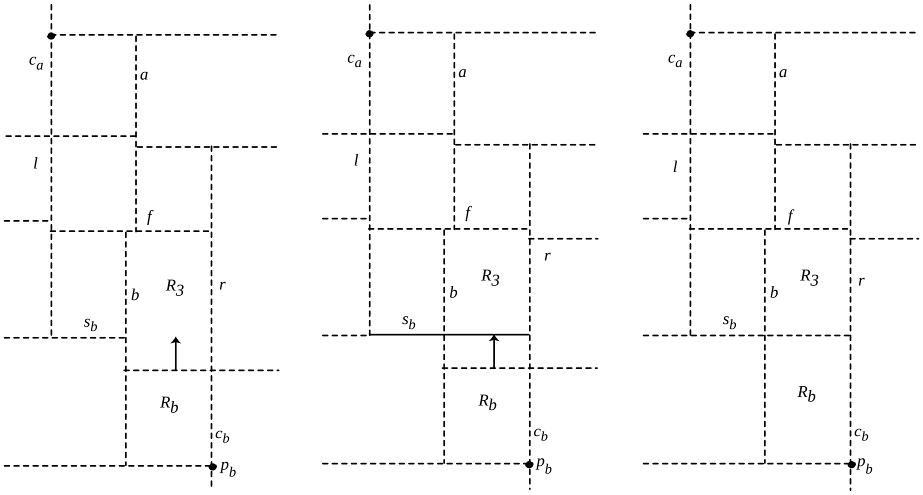


Figure 28: Case 1: Extending a side (such as s_b) that cuts across a rectangle that covers f does not diminish the optimality of the partition.

When s_b is horizontally extended rightwards to a point on c_b, it must cut across the width of one adjacent rectangle: this could either be a rectangle R_3 covering f, or a rectangle touching a rectangle R_b that covers f. In the former case, shown in Fig. 28, R_b becomes larger by the extension, and R_3, which gets shortened, is eventually replaced by R during the repartitioning process. In the latter case, R_b's height will be diminished, but as shown in Fig. 29, it will remain as large as the height of an adjacent rectangle, R_k in the figure, whose height δ_k is $> \delta$ and is unaffected by the repartitioning. Therefore there is no change in the optimality of the partition.

Similar tedious checking shows that the repartioning does not diminish any rectangle below δ. But it reduces the total number of rectangles by 3 (4 are replaced by 1). Therefore the original was not optimal. □

This lemma obviates the need for depth-(≥ 3) cuts:

Theorem 12 *An optimal partition never includes a (floating) cut of depth greater than* 2.

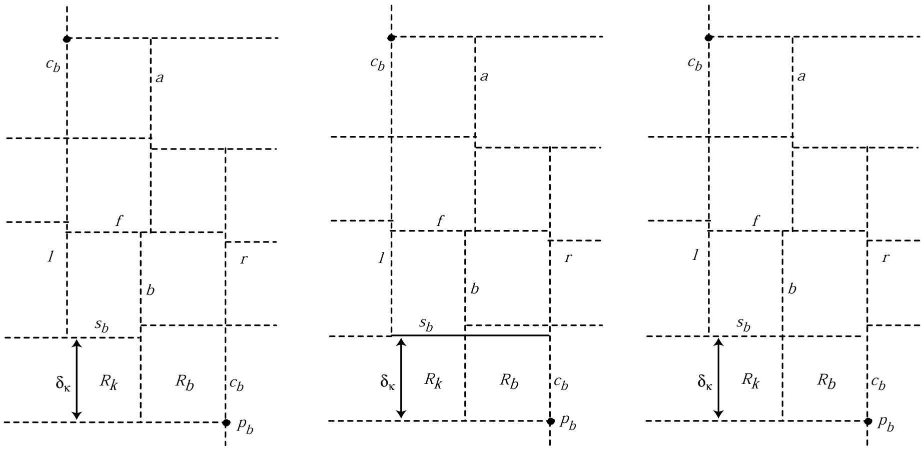


Figure 29: Case 2: Extending a side across a rectangle that touches a rectangle that covers f, does not diminish the optimality of the partition.

Lemma 11 establishes that there can never be a cut of depth ≥ 3 in an optimal partition employing floating cuts. Fig. 20 shows an optimal partition that needs a cut of depth 2, thus establishing that this is the best result possible.

5.4 Combinatorial Lemmas

We need to bound the number of rectangles employed in any optimal partition. Secondly, we need to bound the number of possible values of δ, and from that define a finite grid on which the partition may be drawn.

The first bound relies crucially on the depth bound of Theorem 12. Define k_d as the number of cuts of depth exactly d. Define the *source endpoint* of a cut as the lower endpoint of a vertical cut, and the left endpoint of a horizontal cut.

Lemma 13 *In an optimal partition, the total number of cuts is $O(n)$: $< 12n$.*
Proof: More precisely, we prove:

$$\begin{aligned} k_0 &< 2n \\ k_1 &\leq 4n \\ k_2 &\leq 6n \end{aligned}$$

The number of reflex vertices in a polygon is $< n$, and at most 2 cuts (see Fig. 3) can emanate from each. Therefore $k_0 < 2n$.

To count depth-1 cuts, we charge each to a vertex, and bound the number of charges received by each vertex. A depth-1 cut c_1 must touch a depth-0 cut: either it is hit by, or it hits such a cut. If c_1 is hit by one or more depth-0 cuts (as are the vertical cuts in Fig. 8), then charge c_1 to the vertex of the depth-0 cut that hits c_1 closest to c_1's source endpoint. Because each depth-0 cut can hit only one cut, a vertex receives a most one charge this way. If c_1 is hit only by cuts of depth > 0, then it must itself terminate on a depth-0 cut c_0. We cannot simply charge c_0's vertex v, however, as many depth-1 cuts could be hitting this same vertex cut c_0. If c_1 is the only cut hitting c_0 from one side, then charge c_0's vertex; v can receive at most two charges this way (one from each side of c_0). If there are several (parallel) depth-1 cuts incident to c_0 from the same side, then between each pair there must reside at least one vertex w; otherwise the pinwheel lemma is violated. See Fig. 30 for one possible structure. Charge c_1 to the vertex w nearest c_0 and between c_1 and the next parallel depth-1 cut hitting c_0 (b_1 in the figure). Because w cannot be closest to more than two vertex cuts, each vertex could be charged at most twice this way. Counting these along with the charges to hitting vertex cuts, each vertex receives at most 4 charges from depth-1 cuts. Thus $k_1 \leq 4n$.

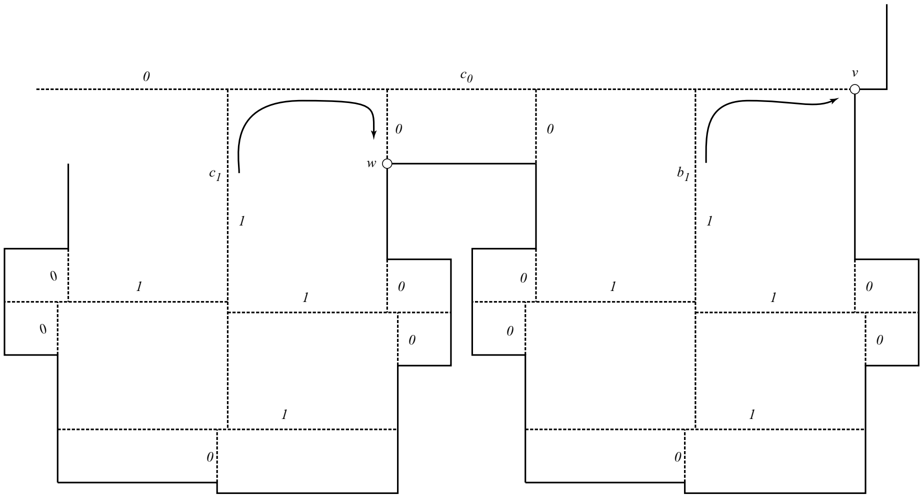


Figure 30: Depth-1 cuts b_1 and c_1 both hit c_0, and neither is hit by any depth-0 cut. In this example, b_1 charges v and c_1 charges w. Numbers adjacent to cuts indicate their depth.

To bound depth-2 cuts, we charge each either to a depth-1 cut, or to a vertex. A depth-2 cut c_2 must touch a depth-1 cut: either be hit by, or hit such a cut. If c_2 is hit by one or more depth-1 cuts (Lemma 8 establishes that there can be multiple cuts hitting a side of c_2 if at least one of them is not movable), then charge c_2 to the depth-1 cut c_1 that hits c_2 closest to c_2's source. Each depth-1 cut can receive at most one charge this way, as it may terminate on only one cut. If c_2 is only hit by depth-2 cuts then one of its endcaps must be a depth-1 cut c_1. Now we are in a situation analogous to that just considered: It could be that many (parallel) depth-2 cuts hit the same depth-1 cut c_1 from the same side, but again there must be vertices between these parallel cuts. (One could alter Fig. 30 with additional structure so that there are depth-2 cuts in the two "wells"). In this case, we charge c_2 to the vertex w between

c_2 and the adjacent parallel depth-2 cut, just as in the above argument. Each vertex can again receive at most two charges this way. So, the total number of charges generated by depth-2 cuts is $k_1 + 2n$. Therefore, $k_2 \le 4n + 2n = 6n$, and $k = k_0 + k_1 + k_2 < 12n$. $\square$

There is overestimation at several points in the above argument; we do not believe the quoted upper bounds can be achieved. It would be of interest to establish tight bounds.

Theorem 14 *An optimal partition consists of $O(n)$ rectangles: $< 18n$.*
Proof: Let R be the number of rectangles, and k be total the number of cuts. Create $4R$ charges, and distribute each rectangle's charges to its four corners. Each corner coincides with a cut endpoint (treating polygon edges as cuts). Each cut endpoint can be shared by at most 4 rectangles. A closer analysis shows a cut endpoint can be shared by at most 3 rectangle corners. First, three is possible when the endpoint is a vertex, as seen in Fig. 3. Second, Lemma 1 shows that otherwise the endpoint is on the interior of an edge or another cut, in which case it is shared by two rectangle corners (e.g., in Fig. 6b, the endpoint of cut d is on the interior of cut b, and is shared by two rectangle corners above and below, left of b). So each cut endpoint receives at most 3 charges. So $4R \le 3(2k)$. By Lemma 13, $k < 12n$, and the conclusion follows. $\square$

5.5 The δ-Graph

The bound on the number of possible values of δ is obtained from the "δ-graph." Let $\mathcal{P}$ be an optimal partition of polygon P with minimum width δ. Define G_δ, the *δ-graph*, to be a graph whose nodes are the segments of the partition (both cuts and polygon edges), with two nodes connected by an arc iff the corresponding segments mutually support a rectangle of $\mathcal{P}$ of width δ to either side. A *monotonic path* in G_δ is a path whose node segments are sorted horizontally or vertically.

Lemma 15 *The δ-graph G_δ of an optimal partition $\mathcal{P}$ with minimum width δ contains a monotonic path both of whose end nodes are either polygon edges or vertex cuts (i.e., both are cuts of depth 0).*

Lemma 15 is manifest in Figures 8, 18, and 20.

Proof: Suppose otherwise. Because $\mathcal{P}$ is optimal with width δ, it must contain some δ-rectangles, which must be supported on both sides. So G_δ contains at least one arc. Take any connected component of G_δ, and call its leftmost and rightmost node segments s_l and s_r, supporting δ-rectangles R_l and R_r. By assumption, it cannot be that both s_l and s_r are either polygon edges or vertex cuts. Without loss of generality, let s_l be a nonvertex cut, and so movable. Slide s_l leftward slightly, enlarging R_l; see Fig. 31. A slide distance can be chosen so that none of the rectangles left of s_l become δ wide (which would create a new arc in G_δ); meanwhile R_l (and other rectangles supported by s_l) becomes wider than δ. The effect is to delete the arc from G_δ corresponding to R_l,

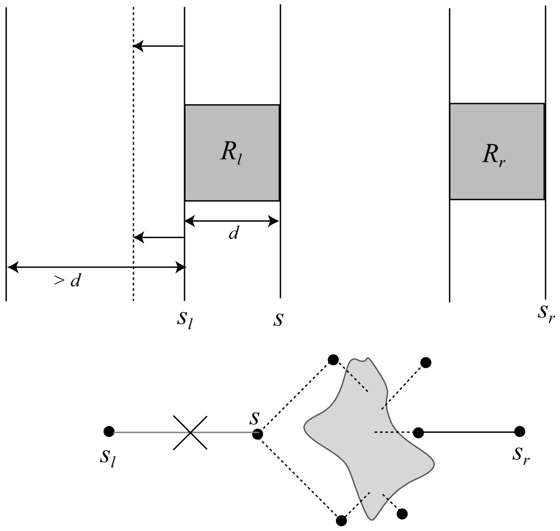


Figure 31: After sliding s_l leftwards, the δ-graph of does not contain an arc connected to s_l.

creating a new, optimal partition with one fewer δ-rectangle. Repeating this process permits removing all δ-rectangles, which contradicts the premise that $\mathcal{P}$ is optimal. $\square$

Lemma 16 *Let an optimal partition of an n-vertex polygon have minimum side length δ. Then $\delta \in \Delta$, where Δ is a set of rational numbers of cardinality $|\Delta| = O(n^3)$, and which can be computed in time $O(n^3)$.*
Proof: By Lemma 15, there is a monotonic path in G_δ between two segments that are either polygon edges or vertex cuts. Both are on lines containing polygon edges. Thus the path spans some distance d_{ij}, where d_{ij} is the distance between two parallel edges of the polygon e_i and e_j. By Theorem 14, an optimal partition contains only $O(n)$ rectangles. Thus δ must be $1/r$ of d_{ij}, with $r = O(n)$. The set of d_{ij} values has cardinality $O(n^2)$, and for each we need only compute d_{ij}/r with $r = O(n)$, for each r. $\square$

Theorem 17 *There is an optimal partition whose cuts fall on a subset of $O(n^4)$ gridlines.*
Proof: All the cuts can be pushed to the left, and down, so that every cut is, for some k, $k\delta$ to the right (or above) some polygon edge. So for each edge, as illustrated in Fig. 32, we can put a line at every δ to its right. Lemma 16 establishes that there are $O(n^3)$ choices for δ for each cut. The result is $O(n^3)$ gridlines per cut, resulting in $O(n^4)$ gridlines overall. $\square$

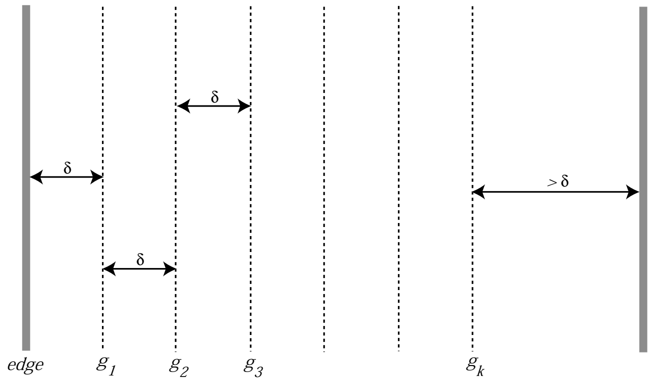


Figure 32: A gridline g_k can be placed $k\delta$ to the right of an edge.

6 Discussion

As mentioned previously, our structural properties, particularly Theorem 12, which limits cut depth, and Theorem 14, which limits the number of rectangles in a optimal partition, can be used to support a complex polynomial-time dynamic programming algorithm for unrestricted cuts. The idea is to show that the polygon can be partitioned by a chain of cuts that connect pairs of vertices. The cut-depth limit permits bounding the length of this chain to at most 10 cuts [Tew02]. This constant-size upper bound, together with the polynomially-sized grid, leads to the complexity listed in Table 1.

The simplicity of the linear bound of Theorem 14 leads us to suspect there should be simpler approaches, both to establishing this bound, and for constructing an optimal partition. A first step might be to establish a tight constant for the linear bound. We have no example that needs as many as n rectangles in an optimal partition, let alone the $18n$ bound established in Theorem 14. However, any new insights must still lead to the complicated uniquely optimal partitions of Figures 18 and 20.

Two natural directions we did not explore, both with practical import, are: orthogonal polygons with holes (we suspect the problem becomes NP-hard, as it does in [LPRS82]), and approximation algorithms.

Acknowledgments. We thank Irena Pashchenko, our coauthor on [OPT01], for her initial work on this project and for many useful subsequent comments. We are grateful to remarkably insightful referee reports, which led to the correction of several errors, and to patient editors.

References

[Kei00] J. M. Keil. Polygon decomposition. In Jörg-Rüdiger Sack and Jorge Urrutia, editors, *Handbook of Computational Geometry*, pages 491–

518. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[LPRS82] A. Lingas, R. Pinter, R. Rivest, and A. Shamir. Minimum edge length partitioning of rectilinear polygons. In *Proc. 20th Allerton Conf. Commun. Control Comput.*, pages 53–63, 1982.

[LTL89] W. T. Liou, J. J. M. Tan, and R. C. T. Lee. Minimum partitioning simple rectilinear polygons in $O(n \log \log n)$ time. In *Proc. 5th Annu. ACM Sympos. Comput. Geom.*, pages 344–353, 1989.

[OPT01] J. O'Rourke, I. Pashchenko, and G. Tewari. Partitioning orthogonal polygons into fat rectangles. In *Proc. 13th Canad. Conf. Comput. Geom.*, pages 133–136, August 2001.

[OT02] J. O'Rourke and G. Tewari. Partitioning orthogonal polygons into fat rectangles in polynomial time. In *Proc. 14th Canad. Conf. Comput. Geom.*, pages 97–100, August 2002.

[Pas01] I. Pashchenko. Optimal orthogonal decompositions of rectilinear polygons. Special studies report, Smith College, 2001.

[Tew02] G. Tewari. Partitioning orthogonal polygons into fat rectangles. Undergraduate thesis, Smith College, 2002.