Computing Partitions of Rectilinear Polygons with Minimum Stabbing Number*

Stephane Durocher and Saeed Mehrabi

Department of Computer Science, University of Manitoba, Winnipeg, Canada,
{durocher,mehrabi}@cs.umanitoba.ca

Abstract. The stabbing number of a partition of a rectilinear polygon P into rectangles is the maximum number of rectangles stabbed by any axis-parallel line segment contained in P. We consider the problem of finding a rectangular partition with minimum stabbing number for a given rectilinear polygon P. First, we impose a *conforming* constraint on partitions: every vertex of every rectangle in the partition must lie on the polygon's boundary. We show that finding a conforming rectangular partition of minimum stabbing number is NP-hard for rectilinear polygons with holes. We present a rounding method based on a linear programming relaxation resulting in a polynomial-time 2-approximation algorithm. We give an $O(n \log n)$-time algorithm to solve the problem exactly when P is a histogram (some edge in P can see every point in P) with n vertices. Next we relax the conforming constraint and show how to extend the first linear program to achieve a polynomial-time 2-approximation algorithm for the general problem, improving the approximation factor achieved by Abam, Aronov, de Berg, and Khosravi (ACM SoCG 2011).

1 Introduction

A polygon P is rectilinear if all of its edges are axis-parallel. A *rectangular partition* of a rectilinear polygon P is a decomposition of P into rectangles whose interiors are disjoint. Rectilinear polygon decomposition has several applications, including VLSI layout design [10] and image processing [6].

Let P be a rectilinear polygon and let R be a rectangular partition of P. Given a line segment ℓ inside P, we say that ℓ *stabs* a rectangle of R if ℓ passes through the interior of the rectangle. The (*rectilinear*) *stabbing number* of R is the maximum number of rectangles of R stabbed by any axis-parallel line segment inside P. Moreover, the *vertical* (resp., *horizontal*) stabbing number of R is defined as the maximum number of rectangles stabbed by any vertical (resp., horizontal) line segment inside P. We say an edge of a rectangle in a rectangular partition of P is *fully anchored* if both of its endpoints are on the boundary of P. Consequently, a rectangular partition of P is called *conforming*,

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

if all edges of its rectangles are fully anchored. A conforming rectangular (CR) partition of P is *optimal* if its stabbing number is minimum over of all such partitions of P.

De Berg and van Kreveld [3] prove that every n-vertex rectilinear polygon has a rectangular partition with stabbing number $O(\log n)$. They show that this bound is asymptotically tight, as the stabbing number of any rectangular partition of a staircase polygon with n vertices is $\Omega(\log n)$. De Berg and van Kreveld [3] and Hershberger and Suri [7] give polynomial-time algorithms that compute partitions with stabbing number $O(\log n)$. Recently, Abam et al. [1] consider the problem of computing an optimal rectangular partition of a simple rectilinear polygon P, that is, a rectangular partition whose stabbing number is minimum over all such partitions of P. By finding an optimal partition for histogram polygons in polynomial time (see Section Section 3), they obtain an $O(n^7 \log n \log\log n)$-time 3-approximation algorithm for this problem. As Abam et al. note, however, the computational complexity of the general problem is unknown.

De Berg et al. [2] studied a related problem in which the objective is to partition a given set of n points in $\mathbb{R}^d$ into sets of cardinality between $n/2r$ and $2n/r$ for a given r, where each set is represented by its bounding box, such that the stabbing number, defined as the maximum number of bounding boxes intersected by any axis-parallel hyperplane, is minimized. They show the problem is NP-hard in $\mathbb{R}^2$. They also give an exact $O(n^{4dr+3/2} \log^2 n)$-time algorithm in $\mathbb{R}^d$ as well as an $O(n^{3/2} \log^2 n)$-time 2-approximation algorithm in $\mathbb{R}^2$ when r is constant. Fekete et al. [5] prove that the problem of finding a perfect matching with minimum stabbing number for a given point set is NP-hard, where the (rectilinear) stabbing number of a matching is the maximum number of edges of the matching intersected by any (axis-parallel) line. They also show that, for a given point set, the problems of finding a spanning tree or a triangulation with minimum stabbing number are NP-hard.

This paper examines the problem of finding an optimal rectangular partition of a given rectilinear polygon, both in the unrestricted version of the problem (partitions need not be conforming) considered by Abam et al. [1] and in the case of conforming rectangular (CR) partitions. For CR partitions, we give an $O(n \log n)$-time algorithm for computing an optimal partition when the input polygon is a histogram with n vertices in Section 3, a polynomial-time 2-approximation algorithm for arbitrary rectilinear polygons (possibly with holes) in Section 4, and we show NP-hardness for finding an optimal partition on rectilinear polygons with holes in Section 5. To the authors' knowledge, this is the first complexity result related to determining the minimum stabbing number of a rectangular partition of a rectilinear polygon, partially answering an open problem posed by Abam et al. [1]. For general (not necessarily conforming) rectangular partitions we give a polynomial-time 2-approximation algorithm in Section 6 that improves on the 3-approximation algorithm of Abam et al. [1]. Complete proofs for the results of Sections 5 and 6 are omitted due to space constraints.

2 Preliminaries

Let P be a simple rectilinear polygon and let R be a CR partition of P. We refer to any maximal line segment whose interior lies in the interior of P and on the boundary of some rectangle in R as a *partition edge*. That is, the partition edges of R correspond to the "cuts" that divide P into rectangles. A vertex u of P is a *reflex* vertex if the angle at u interior to P is $3\pi/2$. We denote the set of reflex vertices of P by $V_R(P)$. For each reflex vertex $u \in V_R(P)$, we denote the maximal horizontal (resp., vertical) line segment contained in the interior of P with one endpoint at u by H_u (resp., V_u) and refer to it as the *horizontal line segment* (resp., *vertical line segment*) of u. Observe that for every reflex vertex u of P, at least one of H_u and V_u must be present in R. The following observation allows us to consider only a discrete subset of the set of all possible rectangular partitions of P to find an optimal partition:

Observation 1. *Any rectilinear polygon P has an optimal rectangular partition in which every partition edge has at least one reflex vertex of P as an endpoint.*

Consequently, every partition edge is either H_u or V_u for some $u \in V_R(P)$.

Given an integer $k \geq 1$, a k-Sum Linear Program (KLP)[1] [11] consists of an $m \times n$ matrix A, an m-vector b, and an n-vector $X = (x_1, x_2, \ldots, x_n)$ for which the objective is to

$$\begin{aligned} \text{minimize} \quad & \max_{S \subseteq N : |S| = k} \sum_{j \in S} c_j x_j \\ \text{subject to} \quad & AX \geq b \\ & X \geq 0, \end{aligned} \tag{1}$$

where $N = \{1, 2, \ldots, n\}$. Observe that when $k = n$, the KLP is equivalent to a classical linear program (LP).

3 Finding an Optimal CR Partition of a Histogram

In this section, we present an algorithm for computing an optimal CR partition of a histogram. A *histogram* (polygon) H is a simple rectilinear polygon that has one edge e that can see every point in P. Equivalently, as defined by Katz and Morgenstern [8], a simple orthogonal polygon P is a vertical (resp., horizontal) histogram if it is monotone with respect to some horizontal (resp., vertical) edge e that spans P; we call e the *base* of H.

Abam et al. [1] give a polynomial-time algorithm for computing an optimal rectangular partition of a histogram. A rectangular partition of a histogram is not necessarily a CR partition. Figure 1(a) shows a histogram whose optimal

[1] Throughout the paper, we use KLP to abbreviate either k-Sum Linear Program or k-Sum Linear Programming. Similarly, we use LP to denote either Linear Program or Linear Programming.

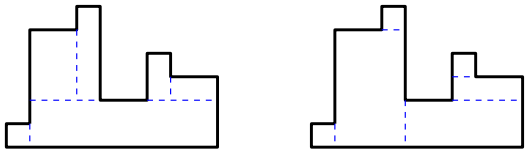

Fig. 1. A vertical histogram H. (a) An optimal rectangular partition of H with stabbing number 2. (b) Any CR partition of H has stabbing number at least 3.

rectangular partition has stabbing number 2. However, any CR partition of this histogram has stabbing number at least 3; see Figure 1(b). Without loss of generality, suppose each histogram is a vertical histogram.

Let H be a histogram with n vertices and let H^- denote the set of horizontal edges of H. Recall that every CR partition of H must include at least one of the edges H_u or V_u for every reflex vertex u in H. The algorithm begins with an initial partition of H, consisting exclusively of horizontal partition edges, that will be modified to produce an optimal CR partition of H by greedily replacing horizontal edges with vertical edges. The initial partition of H is obtained by adding the edge H_u for each reflex vertex u.

Observation 2. *For any* CR *partition of any vertical histogram H and any reflex vertex u in H, the vertical partition edge V_u may be included at u if and only if no horizontal partition edge is included directly below u (otherwise it would intersect V_u).*

Observation 2 suggests a hierarchical tree structure that determines a partial order in which each horizontal partition edge can be removed and replaced by a vertical partition edge, provided it does not intersect any horizontal partition edge below it. Thus, we construct a forest (initially a single tree denoted T_0) associated with the partition; the algorithm proceeds to update the forest and, in doing so, modifies the associated partition as horizontal partition edges are replaced by vertical ones. Define a tree node for each edge in $H^- \cup S$, where $S = \{H_u \mid u \in V_R(H)\}$. Add an edge between two vertices u and v if some vertical line segment intersects both edges associated with u and v, but no other edge of $H^- \cup S$. When the polygon H is a histogram, the resulting graph, T_0, is a tree. See the example in Figure 2(a). We now describe how to construct T_0 in $O(n \log n)$ time. Note that the set S need not be known before construction.

Each edge in H^- is adjacent to two vertical edges on the boundary of H, which we call its left and right neighbours, respectively. Sort the edges of H^- lexicographically, first by y-coordinates and then by x-coordinates. The algorithm sweeps a horizontal line ℓ across H from bottom to top. Initially, ℓ coincides with the base of H; root the tree T_0 at a node u that corresponds to the base of H. The construction refers to a separate balanced search tree that archives the set of vertical edges of H on or below the sweepline, indexed by x-coordinates. Initially, only the leftmost and rightmost vertical edges of H are in the search tree,

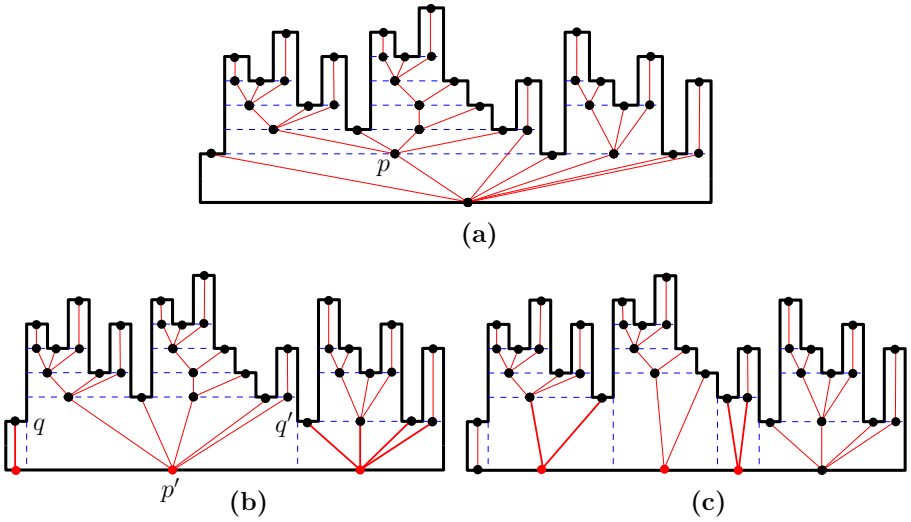

Fig. 2. **(a)** A histogram H and the tree T_0 that corresponds to the initial partition of H. **(b)** The edge associated with node p is removed from the partition and is replaced by two vertical edges anchored at the reflex vertices q and q'. The red vertices denote the roots of the three new resulting trees. **(c)** The algorithm terminates after one more iteration, giving an optimal CR partition of H (with stabbing number 5) along with the corresponding forest.

i.e., the base's neighbours. The construction of the tree T_0 proceeds recursively on u as follows.

Suppose the next edges of H^- encountered by the sweepline ℓ are $e_1, \ldots, e_k$, each of which has equal y-coordinate. Add the respective left and right neighbours of $e_1, \ldots, e_k$ to the search tree. Let l_1 and r_1 denote the x-coordinates of the respective left and right endpoints of edge e_1. Add a node representing e_1 to T_0 as a child of u. Check whether the left neighbour of e_1 (indexed by l_1) lies below ℓ. If not, then find the predecessor of l_1 in the search tree and let y denote its x-coordinate. Let u' denote the line segment on line ℓ with respective endpoints at the x-coordinates y and l_1. Add a node representing u' to T_0 as a child of u. Recursively construct the subtree of u'. Apply an analogous procedure to the right neighbour of e_1 (indexed by r_1). Repeat for each edge $e_i \in \{e_2, \ldots, e_k\}$. Upon completion, the tree T_0 is constructed storing a representation of the initial horizontal partition (see Figure 2(a)). Finally, each tree node stores its height and links to its children in order of x-coordinates; the tree can be updated accordingly after construction. The running time for constructing T_0 is bounded by sorting $O(n)$ edges and a sequence of $O(n)$ searches and insertion on the search tree, resulting in $O(n \log n)$ time to construct T_0.

We now describe a greedy algorithm to construct an optimal CR partition of H using T_0. Observe that the horizontal stabbing number of the initial partition is initially one, whereas its vertical stabbing number corresponds to the height of

T_0. The algorithm stores the forest's trees in a priority queue indexed by height. While the vertical stabbing number of H remains greater than its horizontal stabbing number, split the tree of maximum height, say T. To do this, remove the horizontal partition edge stored in a tree node p, where p is a child of the root of T on a longest root-to-leaf path in T. The choice of T and p is not necessarily unique; it suffices to select any tallest tree T and any longest path in T. Observe that p has at least one and possibly two reflex vertices as endpoints, denoted a and b. Remove the horizontal partition edge associated with p and add a vertical partition edge (V_a or V_b) for each neighbour of p that lies above p on the boundary of H. The tree T is then divided into up to three new trees: a) the subtrees of the root of T to the left of p, b) the subtree rooted at p, and c) the subtrees of the root of T to the right of p. The root of each new tree corresponds to the base edge of H. See Figure 2(b). The following observation is straightforward:

Observation 3. *The horizontal stabbing number of the partition associated with the forest corresponds to the number of trees in the forest, whereas its vertical stabbing number corresponds to the height of the tallest tree in the forest.*

Once the height of the tallest tree becomes less than or equal to the number of trees in the forest, we return either the current partition or the previous partition, whichever has lower stabbing number. The number of steps is $O(n)$, where each step requires $O(\log n)$ time to determine the tree with maximum height using the priority queue.

The algorithm's correctness follows from Observations 2 and 3, and the fact that reducing the vertical stabbing number requires reducing the height of the tallest tree, which is exactly how the algorithm proceeds, decreasing the height of a tallest tree by one on each step. Therefore, we have the following theorem:

Theorem 1. *Given a histogram H, an optimal* CR *partition of H can be found in $O(n \log n)$ time, where n is the number of vertices of H.*

4 An Approximation Algorithm for Rectilinear Polygons

In this section, we present an LP relaxation for the problem of finding an optimal CR partition of a rectilinear polygon, possibly with holes. We show that a simple rounding of the LP relaxation leads to a 2-approximation algorithm for this problem. Our algorithm works even when the input polygon has holes.

Let P be a rectilinear polygon. We define two binary variables u_h and u_v for every reflex vertex $u \in V_R(P)$ that correspond to H_u and V_u, respectively. Each variable's value (1 = present, 0 = absent) determines whether its associated partition edge is included in the partition. If two reflex vertices align, then they share a common variable. For each reflex vertex u in $V_R(P)$, let ℓ_u^- and $\ell_u^|$ be respective maximal horizontal and vertical line segments that pass through $f_\epsilon(u)$ and are completely contained in P, where $f_\epsilon(u)$ denotes an ϵ translation of the point u along the bisector of the interior angle determined by the boundary of P

locally at u, for some ϵ less than the minimum distance between any two vertices of P. This perturbation ensures that ℓ_u^- and $\ell_u^|$ lie in the interior of P, as in the definition of stabbing number. See Figure 3. Let S_u^- (resp., $S_u^|$) be the set of reflex vertices in $V_R(P)$, like v, such that V_v (resp., H_v) intersects ℓ_u^- (resp., $\ell_u^|$). For each reflex vertex $u \in V_R(P)$, let

$$u_{\Sigma^-} = 1 + \sum_{p \in S_u^-} p_v, \quad \text{and} \quad u_{\Sigma^|} = 1 + \sum_{p \in S_u^|} p_h.$$

Thus, u_{Σ^-} and $u_{\Sigma^|}$ denote the number of rectangles stabbed by ℓ_u^- and $\ell_u^|$, respectively, and their maximum values among all reflex vertices u in P correspond to one less than the respective horizontal and vertical stabbing numbers of P. Consequently, the stabbing number of the partition of P determined by the binary variables is

$$1 + \max_{u \in V_R(P)} \{\max\{u_{\Sigma^-}, u_{\Sigma^|}\}\}. \tag{2}$$

A partition divides the polygon into convex regions (more specifically, rectangles) if and only if at least one partition edge is rooted at every reflex vertex. Thus, a CR partition of P corresponds to an assignment of truth values to the set of binary variables such that (i) no two edges of the partition cross, and (ii) for every reflex vertex u, at least one of V_u and H_u is present in the partition.

Therefore, the problem of finding an optimal CR partition can be formulated as a *k-sum integer linear program* as follows:

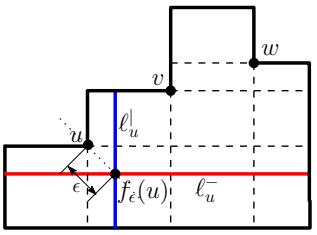

Fig. 3. The maximal line segments ℓ_u^- and $\ell_u^|$ that pass through the point $f_\epsilon(u)$ are shown in red and blue, respectively. In this example, $u_{\Sigma^-} = 1 + u_v + v_v + w_v$ and $u_{\Sigma^|} = 1 + u_h$.

$$\begin{aligned} &\text{minimize } (2) && && (3)\\ &\text{subject to } u_h + u_v \geq 1, && \forall u \in V_R(P),\\ &\qquad v_h + u_v \leq 1, && \text{if } H_v \text{ intersects } V_u,\\ &\qquad u_h, u_v \in \{0,1\}, && \forall u \in V_R(P). \end{aligned}$$

To obtain an integer linear program, we introduce an additional variable y. The following integer linear program is equivalent to the above KLP (see Section 2):

$$\begin{aligned} &\text{minimize } y && && (4)\\ &\text{subject to } y - u_{\Sigma^-} \geq 0 && \forall u \in V_R(P),\\ &\qquad y - u_{\Sigma^|} \geq 0 && \forall u \in V_R(P),\\ &\qquad u_h + u_v \geq 1, && \forall u \in V_R(P),\\ &\qquad -v_h - u_v \geq -1, && \text{if } H_v \text{ intersects } V_u,\\ &\qquad u_h, u_v \in \{0,1\}, && \forall u \in V_R(P). && (5) \end{aligned}$$

Since the number of sums in (2) is $O(n^2)$, the size of the integer linear program above is polynomial in n. Next, we relax the above program by replacing (5) with $u_h, u_v \in [0,1], \forall u \in V_R(P)$ and obtain the following LP:

$$\begin{aligned} &\text{minimize } y && && (6)\\ &\text{subject to } y - u_{\Sigma^-} \geq 0 && \forall u \in V_R(P),\\ &\qquad y - u_{\Sigma^|} \geq 0 && \forall u \in V_R(P),\\ &\qquad u_h + u_v \geq 1, && \forall u \in V_R(P),\\ &\qquad -v_h - u_v \geq -1, && \text{if } H_v \text{ intersects } V_u,\\ &\qquad u_h, u_v \geq 0, && \forall u \in V_R(P). \end{aligned}$$

We observe that the constraints $u_h, u_v \leq 1$ are redundant since we can reduce any $u_h > 1$ (resp., $u_v > 1$) to u_h=1 (resp., u_v=1) without increasing the value of the objective function for any feasible solution. Let s^* be a solution to the above LP. We round s^* to a feasible solution for our problem as follows. For each vertex $u \in V_R(P)$, let

$$u_h = \begin{cases} 0, & \text{if } s^*(u_h) \leq 1/2,\\ 1, & \text{if } s^*(u_h) > 1/2, \end{cases} \quad \text{and} \quad u_v = \begin{cases} 0, & \text{if } s^*(u_v) < 1/2,\\ 1, & \text{if } s^*(u_v) \geq 1/2. \end{cases} \tag{7}$$

We first show that, for every reflex vertex u, at least one of V_u and H_u is present in the partition.

Lemma 1. *For each vertex $u \in V_R(P)$, at least one of u_h and u_v is equal to 1 after rounding a solution to* (6)*.*

Proof. We give a proof by contradiction. Suppose that after rounding a solution to (6), $u_h = u_v = 0$ for some $u \in V_R(P)$. Since $u_h = 0$ by (7) we have $s^*(u_h) \leq 1/2$ and, similarly, since $u_v = 0$ we have $s^*(u_v) < 1/2$. Therefore, $s^*(u_h) + s^*(u_v) < 1$, which contradicts the constraint $u_h + u_v \geq 1$ of (6). □

The next lemma proves that no two edges of the partition obtained by the LP cross each other.

Lemma 2. *If H_v intersects V_u, for two vertices $u, v \in V_R(P)$, then at most one of the variables v_h and u_v is 1 after rounding a solution to the LP.*

Proof. We give a proof by contradiction. Suppose that for two vertices $u, v \in V_R(P)$: (i) H_v intersects V_u, and, (ii) both v_h and u_v are 1 after rounding. Since after rounding v_h=1 by (7) we have $s^*(v_h) > 1/2$. Similarly, since after rounding u_v=1 we have $s^*(u_v) \geq 1/2$. Therefore, $s^*(v_h) + s^*(u_v) > 1$, which contradicts the constraint $v_h + u_v \leq 1$ (or equivalently $-v_h - u_v \geq -1$) of the LP. □

By combining Lemmas 1 and 2, we get the following result:

Corollary 1. *The partition determined by a feasible solution to the LP after rounding is a* CR *partition.*

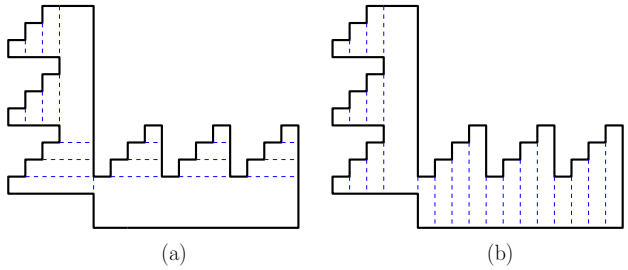

Fig. 4. A simple rectilinear polygon P for which (a) the optimal partition has stabbing number 4 while (b) assigning V_u (or H_u) to every reflex vertex u of P results in a partition with stabbing number at least 10.

By (7), the value of each variable after rounding is at most twice the value of the corresponding variable in the LP solution. Moreover, it is easy to see that the number of constraints in (6) is polynomial in $V_R(P)$, allowing a 2-approximate solution to be found in polynomial time. Therefore, we have the following theorem:

Theorem 2. *There exists a polynomial-time algorithm that constructs a* CR *partition of any given rectilinear polygon P with stabbing number at most twice that of any* CR *partition of P.*

Remark. A preliminary attempt at obtaining a 2-approximation might be to assign to each reflex vertex u its vertical partition edge, V_u (or, equivalently, assigning the horizontal partition edge H_u to each u). Unfortunately, this is not the case: Figure 4 shows a rectilinear polygon for which the optimal CR partition has stabbing number 4. However, the partition obtained by assigning V_u (or H_u) consistently to every vertex $u \in V_R(P)$ has stabbing number at least 10. In fact, the polygon in this example can be extended to show that this heuristic does not provide any constant-factor approximation.

5 Hardness for Rectilinear Polygons with Holes

In this section we present an overview of a reduction showing that the following problem is NP-hard; the complete details of the reduction are omitted due to space constraints.

Optimal CR partition
Input: A rectilinear polygon P possibly with holes
Output: An optimal CR partition of P

We show that Optimal CR Partition is NP-hard by a reduction from Planar Variable Restricted 3SAT (Planar VR3SAT). The Planar VR3SAT problem is a constrained version of 3SAT in which each variable

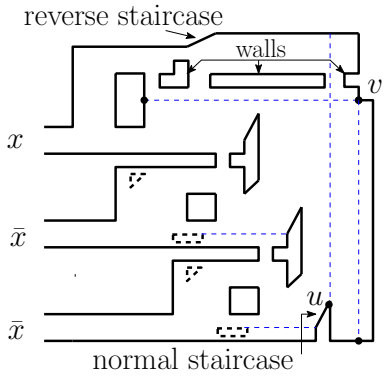

Fig. 5. An example of a variable gadget X linked by three respective corridors to its occurrences (x, $\overline{x}$ and $\overline{x}$) in clauses. Each pair of dashed triangular and rectangular holes form a negation gadget that negates the truth value of x in the associated clause linked by the adjacent corridor. Each staircase consists of c steps. Full details of the variable gadget appear in the complete version of this paper.

can appear in at most three clauses and the corresponding *variable-clause graph* must be planar. Efrat et al. [4] show that Planar VR3SAT is NP-hard.

Let $I = \{C_1, C_2, \ldots, C_k\}$ be an instance of Planar VR3SAT with k clauses and n variables, $X_1, X_2, \ldots, X_n$. We construct a polygon P with holes such that P has a CR partition with stabbing number at most $5c$ if and only if I is satisfiable, where c is a constant that does not depend[2] on I. Given I, we first construct the variable-clause graph of I in the non-crossing comb-shape form of Knuth and Raghunathan [9]. Without loss of generality, we assume that the variable vertices lie on a vertical line and the clause vertices are connected from left or right of that line. Then, we replace each variable vertex X_i with a polygonal variable gadget to which three connecting corridors are attached from its left. The corridors are then connected to the clause gadgets whose associated clauses contain that variable. Figure 5 shows an example of a variable gadget; note the vertex v. Due to the structure of the variable gadget, any CR partition must contain exactly one of the edges V_v or H_v; including V_v (resp., H_v) in the partition corresponds to a truth assignment of *true* (resp., *false*) for the variable x. Moreover, choosing V_v or H_v imposes constraints on how the rest of the variable gadget and its associated clause gadgets can be partitioned. Due to space constraints, we omit detailed descriptions of variable gadgets and clause gadgets. The overall construction implies the following lemma whose proof appears in the complete version of this paper:

Lemma 3. *P has a CR partition with stabbing number at most $5c$, for some constant c, if and only if I is satisfiable.*

By Lemma 3, we obtain the following theorem:

Theorem 3. Optimal CR Partition *is* NP*-hard.*

[2] The definition of the precise value of c refers to specific details of the reduction that have been omitted due to space constraints. Note, however, that the value of c can be specified in polynomial time.

6 Generalizing the Approximation Algorithm

In this section we relax the conforming constraint and consider the problem originally examined by Abam et al. [1], of finding an optimal rectangular partition that is not necessarily conforming. That is, partition edges need not be fully anchored; equivalently, vertices of partition rectangles may lie in the polygon's interior. We extend the LP relaxation presented in Section 4 to achieve a 2-approximation algorithm for this generalized problem for rectilinear polygons. In addition to improving the 3-approximation algorithm of Abam et al. [1], we present a simple algorithmic solution that also works when the input rectilinear polygon has holes. We present a brief overview of the LP relaxation in this section; the details of the algorithm are omitted due to space constraints.

The idea is to consider the arrangement of line segments induced by the intersection of vertical line segments (e.g., V_u for some reflex vertex u) with horizontal line segments (e.g., H_v for some reflex vertex v) inside the polygon P. We refer to these shorter line segments as *fragments* and associate a binary variable with each fragment (as opposed to associating a binary variable with each potential partition edge as in Section 4). By Observation 1 it suffices to consider rectangular partitions for which each partition edge is anchored at some reflex vertex. Thus, a rectangular partition corresponds to an assignment of binary values that observes the following constraints:

1. Every reflex vertex is adjacent to at least one fragment included in the partition.
2. A fragment may be included in a partition if and only if each of its endpoint meets either a) the polygon boundary, b) the continuation of a partition edge along an adjacent fragment, or c) two fragments that form a perpendicular partition edge.
3. At most three of four fragments with a common endpoint can be included in a partition.

A partition's stabbing number is represented as before by the maximum sum of binary variables crossed by any line segment ℓ_u^- or $\ell_u^|$, where u is a reflex vertex in P. The constraints 1–3 can be expressed as a linear program; details are omitted due to space restrictions. Rounding a solution to the linear program as in Section 4 gives the following theorem:

Theorem 4. *There exists a polynomial-time algorithm that constructs a rectangular partition of any given rectilinear polygon P with stabbing number at most twice that of any rectangular partition of P.*

7 Conclusion

This paper considers the problem of finding an optimal partition of a rectilinear polygon P (i.e., a partition with minimum stabbing number over all such partitions of P) for two different types of partitions.

For the first type, CR partitions (in which no partition edge can end in the interior of another edge of the partition) we first described an $O(n \log n)$-time algorithm when P is a histogram polygon with n vertices. Next we presented a LP relaxation of the problem to achieve a polynomial-time 2-approximation algorithm for any given rectilinear polygon with n vertices (possibly with holes). We also proved that the problem is NP-hard for rectilinear polygons with holes. The complexity of the problem for simple rectilinear polygons (without holes) remains open.

For the second type, in which endpoints of partition edges may lie in the interior of P, we gave a polynomial-time 2-approximation algorithm for rectilinear polygons. Our algorithm, based on the extension of our LP relaxation for CR partitions, not only improves the 3-approximation algorithm of Abam et al. [1], but also provides a simple solution that works when polygons have holes. The complexity of the general problem remains open for both simple rectilinear polygons and rectilinear polygons with holes, providing another interesting direction for future research.

References

1. M. A. Abam, B. Aronov, M. de Berg, and A. Khosravi. Approximation algorithms for computing partitions with minimum stabbing number of rectilinear and simple polygons. In *Proc. ACM SoCG*, pages 407–416, 2011.
2. M. de Berg, A. Khosravi, S. Verdonschot, and V. van der Weele. On rectilinear partitions with minimum stabbing number. In *Proc. WADS*, volume 6844 of *LNCS*, pages 302–313. Springer, 2011.
3. M. de Berg and M. van Kreveld. Rectilinear decompositions with low stabbing number. *Inf. Proc. Let.*, 52(4):215–221, 1994.
4. A. Efrat, C. Erten, and S. Kobourov. Fixed-location circular arc drawing of planar graphs. *J. Graph Alg. & Applications*, 11(1):165–193, 2007.
5. S. Fekete, M. Lübbecke, and H. Meijer. Minimizing the stabbing number of matchings, trees, and triangulations. *Disc. Comp. Geom.*, 40:595–621, 2008.
6. K. Gourley and D. Green. A polygon-to-rectangle conversion algorithm. *IEEE Comp. Graphics & App.*, 3(1):31–36, 1983.
7. J. Hershberger and S. Suri. A pedestrian approach to ray shooting: shoot a ray, take a walk. *J. Alg.*, 18(3):403–431, 1995.
8. M. J. Katz and G. Morgenstern. Guarding orthogonal art galleries with sliding cameras. *Inter. J. Comp. Geom. & App.*, 21(2):241–250, 2011.
9. D. Knuth and A. Raghunathan. The problem of compatible representatives. *SIAM J. Disc. Math.*, 5(3):422–427, 1992.
10. M. Lopez and D. Mehta. Efficient decomposition of polygons into L-shapes with application to VLSI layouts. *ACM Trans. Design Automation Elec. Sys.*, 1(3):371–395, 1996.
11. A. Punnen. K-sum linear programming. *J. Oper. Res. Soc.*, 43(4):359–363, 1992.