Lower bounds for approximate polygon decomposition and minimum gap

Joachim Gudmundsson [a] Thore Husfeldt [b]
Christos Levcopoulos [c]

[a] *Department of Computer Science, Utrecht University, PO Box 80.089, 35O8 TB Utrecht, the Netherlands.* joachim@cs.uu.nl

[b] *Department of Computer Science, Lund University, Box 118, 221 00 Lund, Sweden.* thore@cs.lth.se

[c] *Department of Computer Science, Lund University, Box 118, 221 00 Lund, Sweden.* christos@cs.lth.se

Abstract

We consider the problem of decomposing polygons (with holes) into various types of simpler polygons. We focus on the problem of partitioning a rectilinear polygon, with holes, into rectangles, and show an $\Omega(n \log n)$ lower bound on the time-complexity. The result holds for any decomposition, optimal or approximative. The bound matches the complexity of a number of algorithms in the literature, proving their optimality and settling the complexity of approximate polygon decomposition in these cases.

As a related result we show that any non-trivial approximation algorithm for the minimum gap-problem requires $\Omega(n \log n)$ time.

Key words: Lower bounds, Minimum gap, Polygon decomposition, Algebraic decision trees, Computational Geometry

1 Introduction

A decomposition of a polygon P is a collection $\mathcal{C}$ of shapes that are in some sense simpler than P, such that

$$P = \bigcup_{R \in \mathcal{C}} R.$$

The *covering* problem is to find such $\mathcal{C}$ given P, the *partition* problem stipulates that the union be disjoint. In the *rectilinear* case, which is the focus of this paper, P is an orthogonal polygon whose sides are rectilinear, and the R are rectilinear rectangles. The input size, denoted n throughout the paper, is the number of vertices of P. A measure of the quality of decomposition $\mathcal{C}$ is its size $\#\mathcal{C}$; for the partition problem the total edge length of all $R \in \mathcal{C}$ (the 'ink' used) has been suggested as well.

Finding an optimal solution to a decomposition problem is, in many cases, known to be difficult. For example even in the hole-free and orthogonal case, the covering problem has no PTAS unless P = NP [4]. (As a notable exception, a minimum size partition for a hole-free polygon can be found in linear time [12].)

Extensive work has been done on non-optimal decomposition algorithms. Some of these results can be grouped as follows (we refer to a recent survey by Keil [9] for a more complete coverage):

- *The fastest algorithms, including approximation algorithms, for polygons with holes run in time* $\mathrm{O}(n \log n)$, *e.g.*, minimum ink partition [10], minimum size cover [8].
- *Hole-free orthogonal polygons can be decomposed in linear time, e.g.*, minimum size partition [12], rectangular cover of four times optimal size [6], square cover [2], and many results on horizontally convex orthogonal polygons.

We prove the separation suggested by these results:

Theorem 1 *Every algorithm that computes a decomposition of a non-simple connected polygon with n vertices into L (orthogonal) rectangles requires $\Omega(n \log n + L)$ time.*

We stress that the theorem makes no assumptions on the quality of the solution, *e.g.*, the approximation ratio of the algorithm. However, for algorithms that produce coverings of large cardinality (for example, algorithms with approximation ratios worse than $O(\log n)$) the statement trivialises because the output must be reported.

The model underlying Theorem 1 is the algebraic computation tree [13], which is the standard model for low-level lower bounds in computational geometry. Interestingly, since the theorem is stated for any algorithm, exact or approximative, there seems to be no standard reduction that establishes the theorem. We believe that the present paper provides a canonical problem (computational prototype) that may be useful for lower bounds for other approximation algorithms.

By a reduction from sorting, Liou *et. al.* [11] show an $\Omega(n \log n)$ lower bound for the *optimal* partition problem assuming that the polygon has holes. The best upper bound for this problem is $O(n^{3/2} \log n)$ [14]. Aggarwal *et. al.* [1] present an $\Omega(n \log n)$ lower bound for the problem of determining the minimum distance between any pair of vertices of a simple polygon. Their lower bound proof does not apply, however, to the problem of approximating the minimum distance.

The structure of the paper is as follows. In the next section we show how the problem ImproveGap, *i.e.* improving a non-tight lower bound on the minimum difference between two consecutive numbers in a set of n real numbers, can be reduced to the original problem of covering a polygon. In Theorem 4 we show, by using the results of Ben-Or [3], that the time needed to solve ImproveGap is $\Omega(n \log n)$. In Section 3 we make some comments on the input representation of the polygon since the lower bound construction crucially relies on a detail in the input representation. Finally, in Section 4 we generalise the results to other variants of approximating the minimum gap of a set of real numbers.

2 Reduction to a gap problem

Let R denote the real numbers. For $x = \{x_1, \ldots, x_n\}$, $x_i \in R$, let

$$MinGap(x) = \min_{i \neq j} |x_i - x_j|$$

be the *minimum gap* of x.

Problem 2 *(*ImproveGap*) Given a vector of reals $x \in R^n$ and a real number ε with $0 \leqslant \varepsilon < MinGap(x)$, find a real number b such that $\varepsilon < b \leqslant MinGap(x)$.*

The next lemma shows the reduction from ImproveGap to the original cover problem.

Lemma 3 *If there is an algorithm of time complexity $T(n)$ that decomposes a given polygon with n vertices into L orthogonal rectangles then* ImproveGap *can be solved in time $O(T(n) + n + L)$.*

PROOF. We assume without loss of generality that the values in a given instance x of ImproveGap are positive. We construct a polygon P as follows. Let $x_{\min} = \min_i x_i$ and $x_{\max} = \max_i x_i$. These values can be computed in linear time. The outer perimeter of P is a rectilinear rectangle with lower left corner at $(x_{\min} + \varepsilon, 0)$ and upper right corner at $(x_{\max}, 3)$. For each x_i, except $x_{\min}$ and $x_{\max}$, we place a rectangular hole of height 1 and width ε with lower left

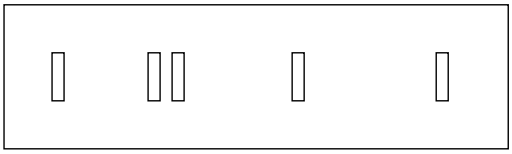

Fig. 1. A polygon constructed from the instance $S = \{0, 1, 3, 3.5, 6, 9, 10.5\}$.

corner at $(x_i, 1)$, Fig. 1. This construction takes time $O(n)$. The algorithm for the decomposition problem returns a collection $\mathcal{C}$ of L rectangles. We can assume that none of these has zero width. Let $w > 0$ be the width of the narrowest rectangle in $\mathcal{C}$, this can be found in time $O(L)$. Then $w + \varepsilon$ is a lower bound on $MinGap(x)$, and is greater than ε.

To prove Theorem 1 it remains to show that ImproveGap requires $\Omega(n \log n)$ time:

Theorem 4 *In the algebraic computation tree model, every algorithm for* ImproveGap *needs time* $\Omega(n \log n)$.

PROOF. Let A be an algorithm for ImproveGap with running time $T(n)$. For the lower bound we introduce a decision version of the minimum gap problem, whose definition depends on A. Let

$$N = \Big\{ (x_1, \ldots, x_n) \mid \{x_1, \ldots, x_n\} = \{1, \ldots, n\} \Big\}$$

be the set of permutations of $(1, \ldots, n)$. Let $B_A = \min_{x \in N} A(x, 0)$. Since N is finite and $A(x, 0) > 0$ we have $B_A > 0$. Let

$$Y_A = \{ x \in R^n \mid A(x, 0) < B_A \}.$$

We define the decision problem Decide_A as follows: Given $x \in R^n$, decide if $A(x, 0) < B_A$.

From the definition it is easy to see that Decide_A has an algorithm with running time $T(n) + O(1)$. On input $x \in R^n$ it runs $A(x, 0)$ and returns 'yes' if and only if $A(x, 0) < B_A$. Note that B_A depends only on A and need not be computed by the algorithm. Thus the running time is as stated.

It remains to prove that Decide_A requires $\Omega(n \log n)$ time. This is a standard argument using the following result by Ben-Or:

Theorem 5 *[3] Let W be a set in Cartesian space R^n and let T be an algebraic decision tree of fixed order $d \geqslant 2$ that solves the membership problem in W. If h is the depth of T and $\#W$ is the number of disjoint connected components of W, then $h = \Omega(\log \#W - n)$.*

We consider the complementary problem of $\textsc{Decide}_A$. Thus let N_A denote the 'no'-instances of $\textsc{Decide}_A$. We show that $\#N_A = \Omega(n!)$ by proving that every point of N lies in a seperate component of N_A. First note that for $x \in N$ we have $A(x, 0) \geqslant \min_{x \in N} A(x, 0) = B_A$, so x is a 'no'-instance and therefore $N \subseteq N_A$. Now consider two different instances $x, x' \in N$. A standard argument shows that any continuous curve from x to x' passes through a point y with $\textsc{MinGap}(y) = 0$. Thus $A(y, 0) = 0$ and therefore $y \notin N_A$. The theorem follows.

One might be tempted to use a problem simpler than ImproveGap for the reduction, for example MinGap or Uniqueness. However, a reduction from MinGap is not appropriate, since we do not need an exact distance for approximation algorithms. MinGap might be a suitable problem if we would consider an optimal algorithm, and not any approximation algorithm. The Uniqueness-problem cannot be considered since we do not allow the gap between two holes to be 0, and if the distance between two holes were zero then this "gap" would not be considered. Hence, neither the Uniqueness-problem nor the MinGap-problem is suitable for the reduction.

3 Comments on input representation

Our results reinforce the literature's dichotomy between polygons with holes and without holes. However, we want to emphasize the fact that this is mainly a question of how the instance is represented. Indeed, the literature's linear-time algorithms exploit a feature in the representation of the input, not in the topology of the polygon.

First, it may be noted that our reduction relies crucially on a detail in the input representation, namely that the holes in P are not ordered from left to right. Were they ordered, the gap problem presented in our reduction could be easily solved in linear time.

The presence of an ordering in hole-free polygons and its absence in the general case is a well-accepted assumption in the literature; the common input representation is a doubly-connected edge list of the edges constituting the perimeter of the polygon, sorted by their ordering on the perimeter. Thus there is a natural ordering of the edge list of a hole-free polygon. Indeed, the known linear-time algorithms for decomposing hole-free polygons rely on the input being presented as a sorted edge list, for example by using Chazelle's linear-time triangulation algorithm [5].

On the other hand, for polygons with holes there is no such ordering. Our

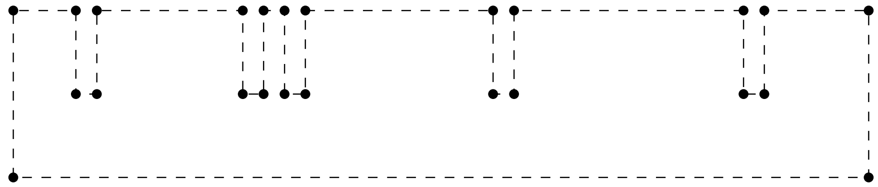

Fig. 2. Reduction from PartitionFromPoints using S as in Fig. 1.

reduction exploits this lack of information to solve a gap problem.

To illustrate this further we consider the covering problem with a different input representation.

Problem 6 (PartitionFromPoints) *For a set of n points that are the corners of a rectilinear, simple polygon P, compute a partition of P.*

We remark that indeed such a polygon is uniquely described by its corners, so the problem is well-defined (assuming that every edge is maximal in the sense that no two edges can be replaced by a single edge). We show that even though this polygon is *hole-free*, it does not admit a linear time decomposition, no matter how bad. Contrast that with the result that the exact partition problem for hole-free polygons can be solved in linear time [11], if they are represented as an ordered list.

Proposition 7 *Every algorithm for* PartitionFromPoints *runs in* $\Omega(n \log n)$ *time.*

PROOF. (Sketch) Given an input $x_1, \ldots, x_n, \varepsilon$ to PartitionFromPoints, let (as in the proof of Lemma 3) $x_{\min} = \min_i x_i$ and $x_{\max} = \max_i x_i$. Let $S = \{1, \ldots, n\} \setminus \{\min, \max\}$. The points of P are

$$\{(x_{\min} + \varepsilon, 0), (x_{\min} + \varepsilon, 2), (x_{\max}, 0), (x_{\max}, 2)\} \cup$$

$$\bigcup_{\forall i \in S} \{(x_i, 2), (x_i, 1), (x_i + \varepsilon, 1), (x_i + \varepsilon, 2)\},$$

as shown in Fig. 2.

4 Approximating the minimum gap

In this section we show that approximating $MinGap(S)$ requires $\Omega(n \log n)$ time.

Denote by $MinGap_2(S)$ the second smallest gap of S. Obviously, $MinGap(S)$ is in the interval $[0, MinGap_2(S)]$. The next problem is to find a smaller interval containing $MinGap(G)$.

Problem 8 *(*IntervalMinGap*) Given a set S of real numbers and the value $MinGap_2(S)$, find an interval I such that $MinGap(S) \in I \subset [0, MinGap_2(S)]$.*

The argument from the previous section can be applied to prove the following result.

Proposition 9 *In the algebraic computation tree model, every algorithm for the* IntervalMinGap *requires time $\Omega(n \log n)$.*

The above result is very strong and implies the lower bound of the well-studied MinGap problem in almost every approximative sense. As an example of a natural approximation formulation of MinGap we propose the following.

Problem 10 *((c_1, c_2)-*ApproximateMinGap*) Given a set S of real numbers find a value δ such that*

$$\delta / c_1 \leqslant MinGap(S) \leqslant \delta c_2$$

Proposition 11 *In the algebraic computation tree model, every algorithm for* (c_1, c_2)-ApproximateMinGap *requires time $\Omega(n \log n)$, for every $c_1, c_2 > 1$.*

Acknowledgments

The authors thank the referees for the Information Processing Letters for their helpful comments and suggestions.

References

[1] A. Aggarwal, H. Edelsbrunner, P. Raghavan and P. Tiwari. Optimal time bounds for some proximity problems in the plane. *Information Processing Letters*, 42(1):55–60, 1992.

[2] R. Bar-Yehuda and E. Ben-Hanoch. A linear-time algorithm for covering simple polygons with similar rectangles. *International Journal of Computational Geometry & Applications*, 6(1):79–102, 1996.

[3] M. Ben-Or. Lower bounds for algebraic computation trees, *Proceedings of the 15th ACM Annual Symposium on Theory of Computation*, pages 80–86, 1983.

[4] P. Berman and B. DasGupta, On the complexities of efficient solutions of the rectilinear polygon cover problems, *Algoritmica*, 17:331–356, 1997.

[5] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Computational Geometry*, 6(5):485–524, 1991.

[6] Y. Cheng, S. Iyengar and R. Kashyap. A new method of image compression using irreducible covers of maximal rectangles. *IEEE Transactions on Software Engineering*, 14:651–658, 1988.

[7] J. C. Culberson and R. A. Reckhow. Covering polygons is hard. *Journal of Algorithms*, 17:2–44, 1994.

[8] D. S. Franzblau. Performance guarantees on a sweep-line heuristic for covering rectilinear polygons with rectangles. *SIAM Journal of Discrete Mathematics*, 2(3):307–321, 1989.

[9] J. M. Keil. Polygon decomposition. In *Handbook of Computational Geometry*, chapter 11, pages 491–518, Elsevier, 1999.

[10] C. Levcopoulos. Improved bounds for covering general polygons with rectangles. *Proceedings of the 7th Foundations of Software Technology and Theoretical Computer Science*, volume 287 of LNCS, 1987.

[11] 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 *Proceedings 5th Annual ACM Symposium on Computational Geometry*, pages 344–353, 1989.

[12] W. T. Liou, J. J. M. Tan and R. C. T. Lee. Covering convex rectilinear polygons in linear time. *International Journal of Computational Geometry & Applications*, 1(2):137–185, 1991.

[13] F. P. Preparata and M. I. Shamos. Computational Geometry: An Introduction. Springer-Verlag, New York, NY, 1985.

[14] V. Soltan and A. Gorpinevich. Minimum dissection of rectilinear polygon with arbitrary holes into rectangles. *Discrete Computational Geometry*, 9:57–79, 1993.