International Journal of Computational Geometry & Applications

ON THE TIME BOUND FOR CONVEX DECOMPOSITION OF SIMPLE POLYGONS

MARK KEIL
Department of Computer Science, University of Saskatchewan, 57 Campus Drive, Saskatoon, Saskatchewan, Canada S7N 5A9, keil@cs.usask.ca

and

JACK SNOEYINK*
Department of Computer Science, University of British Columbia, and Department of Computer Science, University of North Carolina, Sitterson Hall, Chapel Hill, NC 27599-3175 USA, snoeyink@cs.unc.edu

Received received date
Revised revised date
Communicated by Editor's name

ABSTRACT

We show that a decomposition of a simple polygon having n vertices, r of which are reflex, into a minimum number of convex regions without the addition of Steiner vertices can be computed in $O(n + r^2 \min\{r^2, n\})$ time and space. A Java demo is available at `http://www.cs.ubc.ca/spider/snoeyink/demos/convdecomp`

Keywords: simple polygons, dynamic programming, convex decomposition

1. Introduction

Suppose that P is a simple polygon in the plane with n vertices, of which r are *reflex vertices* having interior angles greater than π. The *minimum convex decomposition problem* asks for a decomposition of the interior of P into the minimum number of convex regions.

There is an anomaly in the literature on minimum convex decomposition: If, as in Figure 1, segments of the decomposition can end at arbitrary points, often called *Steiner points*, then Chazelle and Dobkin[2,3] have shown that a minimum decomposition can be computed by dynamic programming in $O(n + r^3)$ time. On the other hand, if the segments must end at vertices of the polygon, as in Figure 2, then the best published bound is a dynamic programming algorithm of Keil[9,10] that computes a minimum decomposition in $O(nr^2 \log r)$ time. As Chazelle and Dobkin[3] noted in 1985, this is asymptotically slower for any non-constant r, even though allowing Steiner points usually makes optimization problems more difficult.

After some definitions, we show in Section 3 that Keil's dynamic programming algorithm can be implemented using stacks in place of a search structure, which removes a $\log r$ factor from the running time. Then we show in Section 4 how to reduce the input, in $O(n + r^2 \log n)$ time, to a polygon that has the same minimum decomposition but at most r^2 sides. Thus, a minimum decomposition of P can be computed in $O(n + \min\{nr^2, r^4\})$ time, matching Chazelle and Dobkin's running time at least when $r = O(\sqrt[4]{n})$. If we apply a similar reduction when Steiner points are allowed, we obtain a polygon that has the same minimum decomposition but only $O(r)$ sides. This may explain why the problem without Steiner points appears harder.

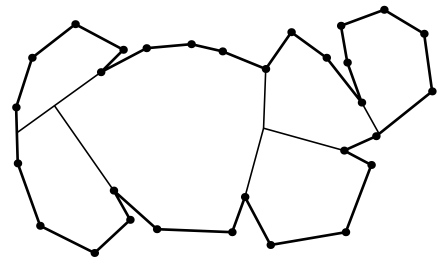

Figure 1: Minimum convex decomposition with Steiner points

A related problem is that of finding a convex decomposition with "minimum ink"—that is, having minimum total edge length. We can simplify the search structure from Keil's minimum ink algorithm[9] to achieve $O(n^2r^2)$ time, but cannot reduce the dependence on n. Greene[7] already achieved this time bound with a double dynamic programming algorithm that was a contemporary of Keil's.

2. Notation for simple polygons

Assume that we are given the n vertices of a polygon $P = \{p_0, p_1, \ldots, p_{n-1}\}$ in counter-clockwise (ccw) order. We assume that P is *simple:* that is, the only intersections between the polygon *edges,* the segments $\overline{p_i p_{i+1}}$ that form the boundary of P, are at the shared endpoint of adjacent edges.

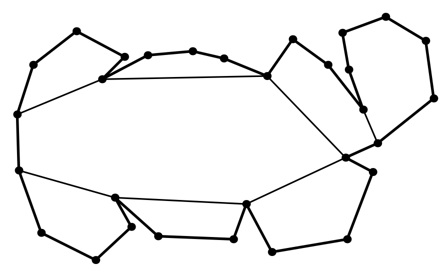

Figure 2: A minimum convex decomposition without Steiner points

A *diagonal* of P is a segment that joins two vertices of P and remains strictly inside P. We use the notation d_{ij} for the diagonal $\overline{p_i p_j}$ with $i < j$. Diagonals for a given vertex p_i can be found by computing the *visibility polygon* for p_i in linear time[6].

One important observation for visibility algorithms will also be important for us: that diagonals appear in the same order as vertices.

Observation 1 *The angular order of diagonals $d_{ij_1}, d_{ij_2}, \ldots d_{ij_k}$, counter-clockwise (ccw) around a vertex p_i is identical to the order of the vertices $p_{j_1}, p_{j_2}, \ldots p_{j_k}$ ccw around P.*

A vertex of P is *reflex* if its interior angle is greater than π. It is not hard to see that a minimum convex decomposition without Steiner points must use diagonals that have at least one reflex vertex, since removal of a diagonal from a minimum decomposition must result in a non-convex region. In fact, an easy way to obtain a decomposition into at most four times the minimum number of convex pieces[7,8] is to start with any triangulation of P, and consider removing each diagonal in turn unless doing so forms a reflex angle.

Note that a particular diagonal d that joins two reflex vertices of P is not necessarily present in a minimum convex decomposition—a decomposition of P may use several diagonals that intersect d instead.

3. Dynamic programming for convex decomposition

To find a minimum set of diagonals and solve the minimum convex decomposition problem we use *dynamic programming*, which is an algorithmic paradigm that finds the best solution to a problem by combining optimal solutions to subproblems[1].

We define a subproblem for each diagonal d_{ik} such that p_i or p_k is a reflex vertex: let P_{ik} denote the polygonal line $p_i, p_{i+1}, \dots, p_k$. The *size* of subproblem P_{ik} is the number of vertices in P_{ik}. We want to associate with each P_{ik} a *weight* w_{ik}, which is the minimum number of diagonals in a convex decomposition of P_{ik}. We make the convention that $d_{0(n-1)}$ is also a diagonal, so we compute the number of diagonals in a minimum convex decomposition by computing the weight of $P_{0(n-1)} = P$.

We can identify three approaches to apply dynamic programming to find the weight of P_{ik} from the weights of smaller subproblems.

1. Consider each convex polygon C that can be constructed adjacent to d_{ik} in P_{ik}. Removal of C from P_{ik} leaves a number of subproblems; the weight of using C is the sum of subproblem weights plus the number of subproblems, since each is cut off by a diagonal. The weight of P_{ik} is the minimum weight over all polygons C. This is the basis of Greene's approach[7], but is complicated by the fact that the polygons must be explored very efficiently.

2. Consider each triangle T that can be constructed adjacent to d_{ik} in P_{ik}. Removal of T leaves two subproblems; the weight of using T is the sum of subproblem weights plus the 0, 1, or 2 edges of T that may need to be introduced to avoid reflex vertices when merging the subproblem solutions. This approach of Keil[9] must keep several solutions for each subproblem, as sketched in Subsection 3.1.

3. Consider each triangle T that could be part of a "canonical triangulation," which will be defined in Subsection 3.2. The decomposition always uses 1 or 2 edges of these triangles, so the task of merging becomes easier. Multiple solutions must still be stored, but the following subsections show that the solutions can be stacked to be ready when needed.

Although we have described these three approaches as top-down recursive procedures, we prefer to implement them by bottom-up iteration, solving subproblems in order of increasing size to end with the weight of a minimum decomposition of P. We initialize by the convention that $w_{i(i+1)} = -1$.

3.1. Equivalent decompositions and narrowest pairs

Since there may be exponentially-many decompositions of P_{ik} that attain weight w_{ik}, Keil[9] suggests storing only certain equivalence classes of decompositions. Associate a pair of vertex indices $[a, b]$ with each decomposition of P_{ik} by noting that one convex polygon of the decomposition will be incident on diagonal d_{ik} and will have vertices a, i, k, b in clockwise order, where possibly $a = b$. Two decompositions of P_{ik} are considered *equivalent* if they have the same weight and the same associated pair of indices.

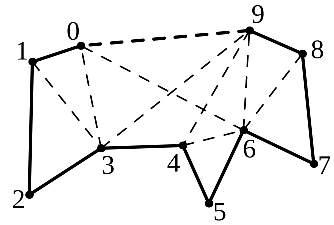

Figure 3: Useful diagonals of P_{09}

Consider the example of polygon P_{09}, shown in Figure 3, Its minimum convex decompositions, shown in Figure 4, are labeled with their associated pairs. To observe that these are all the minimum decompositions, note that any convex decomposition of P_{09} must use diagonals to eliminate the reflex vertices 3, 4, and 6; there are eleven ways to do so with only three diagonals.

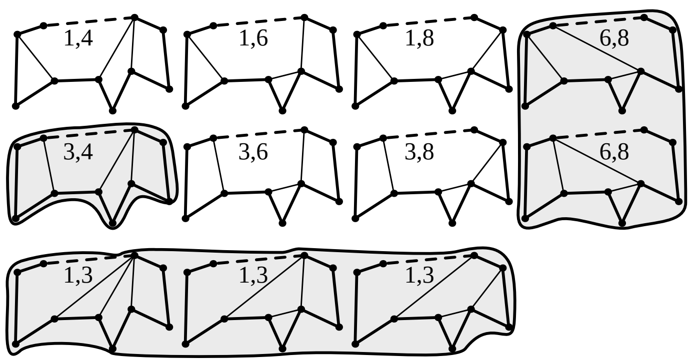

Figure 4: The eleven minimum convex decompositions of polygon P_{09} with *narrowest pairs* circled

In this example, certain "narrowest" pairs have been circled. The *narrowest pairs* are those whose convex region in a small neighborhood of d_{ik} does not contain the convex region of any other minimum decomposition of P_{ik}. Keil observed that only subproblems with narrowest pairs need to be used to assemble a minimum convex decomposition.[9]

According to Observation 1, we can test for narrowest pairs for P_{ik} by simply comparing indices—we discard any interval $[a, b]$ that contains a smaller interval. Notice that this means that if a set of narrowest pairs is ordered by their first indices, then they will also be ordered by their second indices. As we compute indices of narrowest pairs for subproblem P_{ik}, we will store them on a stack $\mathcal{S}_{ik}$ in order, so that the segments from bottom to top are in ccw order around p_i and p_k. Stack $\mathcal{S}_{09}$ for Figure 4 would contain $[1, 3]$, $[3, 4]$, and $[6, 8]$, from bottom to top. Thus, we would know that diagonal d_{06} and edge d_{89} formed the narrowest pair that was furthest counter-clockwise.

3.2. Canonical triangulations

One way to decompose the subproblem P_{ik} into smaller subproblems would be to determine a convex polygon C that is incident on d_{ik} in some minimum convex decomposition of P_{ik}. Removing that C leaves smaller subproblems, whose optimal decompositions could have already been computed. This is essentially the approach taken by Greene[7]. Since this approach can involve turning one subproblem into many, we instead extend any minimum convex decomposition to a canonical triangulation by adding extra diagonals. Removing the triangle incident on d_{ik} leaves at most two smaller subproblems, each of which have canonical triangulations.

Consider any convex decomposition in which every diagonal is incident on a reflex vertex. We assume that vertex p_0 is reflex either by convention, or by renumbering vertices of P if P is not already convex. We can complete such a decomposition to a *canonical triangulation* as follows: each convex region C has at least one vertex that was reflex in P—connect the reflex vertex in C with lowest index to all vertices with higher index in C. If C is not yet triangulated after this step, then the vertex with highest index was reflex in P; connect it to the remaining vertices in that region. In Figure 5 shows an example in which vertices between p_8 and p_{12} and between p_{20} and p_{26} connect to the highest index in their region; all other vertices connect to the lowest.

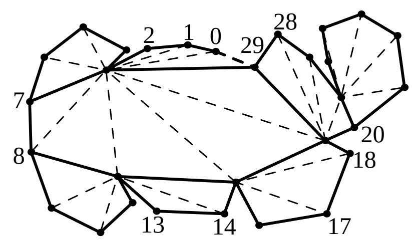

Figure 5: Canonical triangulation from minimum decomposition

We can make the following observations about the diagonals of a canonical triangulation and the subproblems that they form.

Observation 2 *In a canonical triangulation, each diagonal d_{ik}, with $i < k$, has three properties:*

1. *The diagonals with endpoints in P_{ik} define a canonical triangulation of P_{ik}.*
2. *If p_i is reflex in P, then the adjacent triangle $\triangle p_i p_j p_k$, with $i < j < k$, either has $j = k-1$ or d_{jk} is a diagonal used in the convex decomposition.*
3. *If p_i is not reflex in P, then p_k must be. The adjacent triangle $\triangle p_i p_j p_k$, with $i < j < k$, either has $j = i+1$ or d_{ij} is a diagonal used in the convex decomposition.*

Now, the minimum convex decompositions of P_{ik} can be constructed by considering which vertices p_j can form a canonical triangle with d_{ik}, and whether diagonals d_{ij} and d_{jk} must be added to the decompositions of P_{ij} and P_{jk}, or whether those diagonals are merely in the canonical triangulation. In support of that decision, we will also associate with each P_{ik} a stack, $\mathcal{S}_{ik}$, that contains equivalence classes of solutions of P_{ik}.

3.3. Solving subproblems

We can use these properties to systematically explore the canonical triangulations of the minimum decompositions of P_{ik} that have narrowest pairs. We assume that we have, for each subproblem P_{xy} that is smaller than P_{ik}, the narrowest pairs for all minimum convex decompositons of P_{xy}. These are in ccw (increasing) order in stack $\mathcal{S}_{xy}$ and in cw (decreasing) order in stack $\mathcal{T}_{xy}$. (The data structure can be implemented as a single list with two independent "stack" pointers that start on either end.) We use stack $\mathcal{T}_{ij}$ or $\mathcal{S}_{jk}$, depending on whether p_i is a reflex vertex or not, to produce the narrowest pairs for minimum decompositions of P_{ik} in ccw order on stack $\mathcal{S}_{ik}$. Then we make $\mathcal{T}_{ik}$ from $\mathcal{S}_{ik}$, so that both are available for subsequent computation.

A. p_i **reflex:** Minimum decompositions of P_{ik} use, for some $i < j < k$, the diagonal or edge d_{jk}, a decomposition of P_{jk}, and a decomposition of P_{ij}, perhaps with the diagonal d_{ij}. That is, for subpolygon P_{ik} we are using the following dynamic programming recurrence.

$$w_{ik} = \min_{\substack{i < j < k \\ \& \; d_{ij},\, d_{jk} \text{ exist}}} \begin{cases} w_{ij} + w_{jk} + 2 & \text{if } d_{ij} \text{ must be included} \\ w_{ij} + wjk + 1 & \text{otherwise} \end{cases}$$

To compute all narrowest decompositions, we consider, in increasing order, the vertices j with $i < j < k$ and d_{ij} and d_{jk} in the visibility graph. If vertex j were not visible from vertex i, then diagonal d_{jk} would not be one of a narrowest pair for P_{ik}.

For a given j, popping the cw-ordered stack $\mathcal{T}_{ij}$ will go through the pairs in ccw order: find the last pair $[s, t]$ such that d_{tj} and d_{jk} do not form a reflex angle at p_j. If, as in the upper half of Figure 6, there is no such pair $[s, t]$, or if d_{is} and d_{ik} form a reflex angle at p_i, then we must use diagonal d_{ij} to obtain a convex decomposition of P_{ik} with weight $w_{ij} + w_{jk} + 2$ and narrowest pair $[j, j]$. (Recall our convention that the weight of any polygon edge $w_{i(i+1)} = -1$.) Otherwise, as in the lower half of Figure 6, we obtain a convex decomposition of P_{ik} with weight $w_{ij} + w_{jk} + 1$ and narrowest pair $[s, j]$.

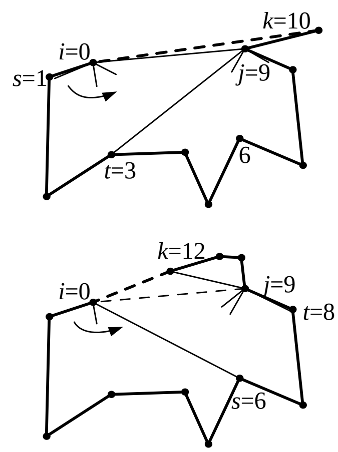

Figure 6: p_i reflex: use d_{jk} and perhaps d_{ij}

To build the ccw-ordered stack $\mathcal{S}_{ik}$ of narrowest pairs for P_{ik} is easy since the second element is always the loop index j. For each pair $[x, j]$ that achieves the minimum weight, push $[x, j]$ if the first element of the pair on top of $\mathcal{S}_{ik}$ is $< x$. Otherwise the pair on top of the stack $\mathcal{S}_{ik}$ is narrower.

Because we use d_{jk} in the decomposition, either $j = k-1$ so that d_{jk} is a polygon edge or at least one of p_j and p_k is reflex.

B. p_i **not reflex:** This case is symmetric except that we know that p_k is reflex. Since minimum decompositions use d_{ij}, either p_j is reflex or d_{ij} is a polygon edge. We consider, in increasing order, the index $j = i+1$ and indices of reflex vertices p_j with $i+1 < j < k$ for which d_{ij} and d_{jk} are in the visibility graph.

Popping the ccw-ordered stack $\mathcal{S}_{jk}$ will go through the pairs in cw order: find the last pair $[s,t]$ such that d_{ij} and d_{js} do not form a reflex angle at p_j. If there is no such pair $[s,t]$, or if d_{tk} and d_{ik} form a reflex angle at p_k, then we must use diagonal d_{jk} to obtain a convex decomposition of P_{ik} with weight $w_{ij} + w_{jk} + 2$ and narrowest pair $[j,j]$. Otherwise, we obtain a convex decomposition of P_{ik} with weight $w_{ij} + w_{jk} + 1$ and narrowest pair $[j,t]$.

To build the ccw-ordered stack $\mathcal{S}_{ik}$ of narrowest pairs for P_{ik} is again easy since now the first element is the loop index j. For each pair $[j,x]$ that achieves the minimum weight, while the second element of the pair on top of $\mathcal{S}_{ik}$ is $\geq x$, pop $\mathcal{S}_{ik}$. Then push $[j,x]$.

Given polygon $P = \{p_0, p_1, \ldots, p_{n-1}\}$, compute a minimum convex decomposition by dynamic programming. This procedure shows the flow of control; Subroutines TypeA and TypeB are described in text.

Procedure `MCD`(P)
    Initialize weights of edges $w_{i(i+1)} = -1$;
    for `visible`$(i, i+2)$ { $w_{i(i+2)} = 0$; push $[i+1, i+1]$ on $\mathcal{S}_{i(i+2)}$; }

    for *size* $= 3$ **to** n **do** {
        for reflex vertices p_i with $i + \mathit{size} \leq n$ **do**
            $k = i + \mathit{size}$; **if** `visible`(i,k) {
                if (p_k reflex) **for** $j = i+1$ **to** $k-1$ **do** `TypeA`(i,j,k);
                else {   /* *Need* p_j *reflex or* $j = k-1$ */
                    for reflex p_j with $i < j < k-1$ **do** `TypeA`(i,j,k);
                    `TypeA`$(i, k-1, k)$; }
            }
        for reflex vertices p_k with $\mathit{size} \leq k < n$ **do**
            $i = k - \mathit{size}$;
            if (p_i not reflex and `visible`(p_i, p_k)) {
                `TypeB`$(i, i+1, k)$;   /* *Need* $j = i+1$ *or* p_j *reflex* */
                for reflex p_j with $i+1 < j < k$ **do** `TypeB`(i,j,k);
            }
        Stack $\mathcal{S}_{ik}$ is complete in ccw order; form stack $\mathcal{T}_{ik}$ in cw order;
    }

Algorithm 1: MCD algorithm

3.4. Correctness and analysis

Algorithm 1 shows the flow of control for the dynamic programming.

Theorem 3 *Given a simple polygon with n vertices, r of which are reflex, we can* *solve the minimum convex decomposition problem in $O(nr^2)$ time and space.*

Proof: It is not difficult to bound the total running time of Algorithm 1: the algorithm calls subroutines `TypeA`(i, j, k) or `TypeB`(i, j, k) for triples $i < j < k$ that are indices of at least two reflex vertices, or one reflex vertex and polygon edge. Thus, there are less than nr^2 calls. The work done in each subroutine is constant plus the number of pairs popped from stacks; since each subroutine adds at most one pair to two stacks, at most $O(nr^2)$ elements can be popped. Thus, $O(nr^2)$ bounds the total time. The memory requirements in the worst case are dominated by the $O(nr^2)$ space for the stacks.

To prove the correctness of Algorithm 1 we can argue by induction that we inspect the canonical triangulations and find the narrowest pairs in ccw order for each minimum convex decomposition of P_{ik}, where p_i or p_k is reflex. The key is that by our flow of control—solving subproblems from smallest to largest—pairs popped from a stack will never be needed again. For example, if while solving P_{ik} we pop the ccw-ordered stack $\mathcal{S}_{jk}$, then Observation 1 implies that any $P_{i'k}$ with $i < i'$ that uses subproblem P_{jk} will have $d_{i'j}$ clockwise of d_{ij}, and thus would also require popping $\mathcal{S}_{jk}$. ∎

4. Biased convex decompositions

To reduce the dependence on n, the input size, we can look for a decomposition of a special form. We say that a minimum convex decomposition is *biased* if diagonals that end at a convex vertex can neither be moved to the next vertex ccw, nor deleted and replaced by a reflex-reflex diagonal (RR-diagonal) while maintaining a convex decomposition.

We single out two special types of diagonals that end at convex vertices; these definitions are easier to illustrate (Figure 7) than to write down. A diagonal $\overline{rp}$, with reflex vertex r and convex vertex p, is a *reflex extension*, or *RE-diagonal,* if the extension through r of the edge after r in ccw order first hits vertex p or the edge after p. Similarly, diagonal $\overline{rp}$ is a *diagonal extension,* or *DE-diagonal,* if the extension through r of an RR-diagonal or RE-diagonal incident on r first hits vertex p or the edge after p.

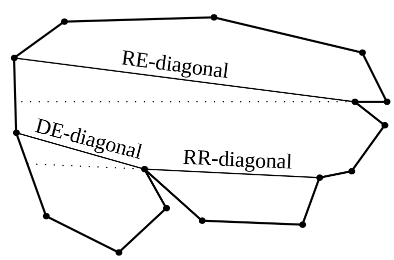

Figure 7: Types of diagonals

Note that an RE-diagonal or DE-diagonal cannot be moved in a convex decomposition; doing so would create a reflex angle with a polygon edge (in the RE-diagonal case) or with an RR-diagonal or RE-diagonal (in the DE-diagonal case) that is incident to its reflex vertex. In fact, these are the only possible obstructions to moving the *lead diagonal* at any convex vertex of P—the diagonal that bounds the same face as the next polygon edge ccw from that vertex.

Lemma 1 *In a biased decomposition of a polygon P, the lead diagonal at any convex vertex $p_i \in P$ must be an RE-diagonal or DE-diagonal.*

Proof: Let $\overline{rp_i}$ be the lead diagonal under consideration in the convex decomposition. By the definition of lead diagonal, the next vertex p_{i+1} is also on the boundary of the convex region on the left of $\overline{rp_i}$. Therefore, $\overline{rp_{i+1}}$ is a diagonal of P that does not intersect any other diagonals of the decomposition.

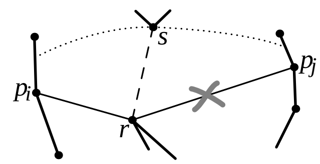

Figure 8: Replace $\overline{rp_j}$ with $\overline{rs}$

On the other hand, because the decomposition is biased, we cannot replace diagonal $\overline{rp_i}$ with $\overline{rp_{i+1}}$. Because the replacement cannot cause non-convexity at p_i or at p_{i+1}, it must do so at r.

We argue that if the non-convexity at r is not caused by a polygon edge or an RR-diagonal, then it is caused by an RE-diagonal at r. Assume, therefore, that non-convexity is caused by a diagonal $\overline{rp_j}$, with p_j a convex vertex. If the convex region that has p_i, r, and p_j on its boundary also has another reflex vertex s, then adding the RR-diagonal $\overline{rs}$ as in Figure 8, would allow us to delete $\overline{rp_i}$ or $\overline{rp_j}$—one of these deletions will maintain the convex decomposition. Thus, by the definition of biased decompositions, we conclude that the portion of the boundary from p_j ccw to p_i is a convex chain of polygon edges. Since p_j cannot move ccw, it must be an RE-diagonal.

This completes the proof that $\overline{rp_i}$ must be an RE-diagonal or DE-diagonal. ■

As an easy corollary, any convex decomposition can be converted to a biased decomposition by deleting, moving and replacing diagonals. We are not concerned about the time complexity of this process, just that it is sufficient to look for a minimum convex decomposition among those that are biased.

Corollary 4 *Any convex decomposition can be converted to a biased decomposition by a finite number of steps that delete, move and replace diagonals.*

Proof: Each operation either decreases the number of non-RR-diagonals, or advances a non-RR-diagonal's endpoint. No endpoint is revisited. ■

Notice that we now know a subset of vertices of P that can be used as endpoints of diagonals in a minimum, biased, convex decomposition—the set of vertices that can be endpoints of RE-diagonals or DE-diagonals. We can explicitly construct this set by ray shooting,[4,5] which takes $\log n$ time for each ray extended in P. A tempting idea, therefore, is to reduce P to a polygon that uses just a few convex vertices, and run the dynamic programming algorithm on the reduced polygon.

This idea works well for the decomposition that allows Steiner points; Chazelle and Dobkin's algorithm for convex decomposition with Steiner points makes use of "RE-diagonals" and *X-configurations*, which are embedded trees that join reflex vertices and have all angles bounded by π. Since no DE-diagonals are needed, the only convex vertices that are relevant are potential endpoints of RE-diagonals.

Here are the steps to construct an $O(r)$-size polygon that contains all relevant vertices, and shows that the time complexity for minimum convex decomposition is of the form $O(n + r\log n + T(r))$. In polygon P, mark the edges incident on reflex vertices, then shoot inside P from every reflex vertex along the extensions of the incident edges, and mark the edges hit. Form polygon P' by omitting from P all vertices not incident to marked edges; P' has at most $7r$ vertices.

To prove that P' is simple, consider deleting vertices one by one and stop when the first intersection occurs. This must be by an edge passing over a reflex vertex. As illustrated in Figure 9, however, the shortest path that joins the extensions of edges at any reflex vertex is the same in P' as it is in P since it turns only at reflex vertices. The extensions and this path certify that no edge of P' crosses a reflex vertex.

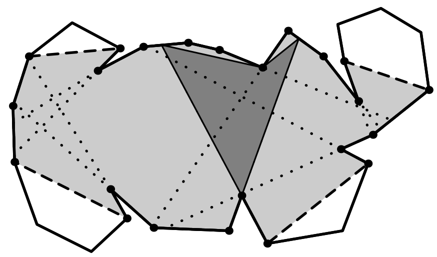

Figure 9: Forming P'

A biased minimum convex decomposition of P with Steiner points will move the RE-diagonals so that they end on edges of P that are included in P'. The X-configurations will not be affected, but will remain in P'. Thus, a minimum decomposition of P' gives a minimum decomposition of P.

To perform a similar reduction for the non-Steiner problem, we would have to add, in the worst case, $\min\{n, r^2\}$ endpoints for DE-diagonals, since there are potentially $r(r-1)$ RR-diagonals and r RE-diagonals that must be extended. Thus, after $O(n + \min\{n, r^2\} \log n)$ time for ray shooting and other preprocessing, the dynamic programming algorithm runs in $O(\min\{n, r^2\}r^2)$ time, giving $O(n + \min\{nr^2, r^4\})$ time overall.

Theorem 5 *Given a simple polygon with n vertices, r of which are reflex, we can solve the minimum convex decomposition problem in $O(n + \min\{nr^2, r^4\})$ time and space.*

Acknowledgment

We thank Mariette Yvinec and Sylvan Lazard for discussions, and the anonymous referees for their encouragement to give more details.

References

1. R. E. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1987.
2. B. Chazelle and D. P. Dobkin. Decomposing a polygon into its convex parts. In *Proc. 11th Annu. ACM Sympos. Theory Comput.*, pages 38–48, 1979.
3. B. Chazelle and D. P. Dobkin. Optimal convex decompositions. In G. T. Toussaint, editor, *Computational Geometry*, pages 63–133. North-Holland, Amsterdam, Netherlands, 1985.
4. B. Chazelle, H. Edelsbrunner, M. Grigni, L. J. Guibas, J. Hershberger, M. Sharir, and J. Snoeyink. Ray shooting in polygons using geodesic triangulations. *Algorithmica*, 12:54–68, 1994.
5. B. Chazelle and L. J. Guibas. Visibility and intersection problems in plane geometry. *Discrete Comput. Geom.*, 4:551–581, 1989.

6. H. ElGindy and D. Avis. A linear algorithm for computing the visibility polygon from a point. *J. Algorithms*, 2:186–197, 1981.
7. D. H. Greene. The decomposition of polygons into convex parts. In F. P. Preparata, editor, *Computational Geometry*, volume 1 of *Adv. Comput. Res.*, pages 235–259. JAI Press, London, England, 1983.
8. S. Hertel and K. Mehlhorn. Fast triangulation of the plane with respect to simple polygons. *Inform. Control*, 64:52–76, 1985.
9. J. M. Keil. Decomposing a polygon into simpler components. *SIAM J. Comput.*, 14:799–817, 1985.
10. J. M. Keil and J.-R. Sack. Minimum decompositions of polygonal objects. In G. T. Toussaint, editor, *Computational Geometry*, pages 197–216. North-Holland, Amsterdam, Netherlands, 1985.