The Contiguous Art Gallery Problem is in $\Theta(n \log n)$

Sarita de Berg[1] Jacobus Conradi[2] Ivor van der Hoog[1] Eva Rotenberg[1]

Abstract. Recently, a natural variant of the Art Gallery problem, known as the *Contiguous Art Gallery problem* was proposed. Given a simple polygon P, the goal is to partition its boundary ∂P into the smallest number of contiguous segments such that each segment is completely visible from some point in P. Unlike the classical Art Gallery problem, which is NP-hard, this variant is polynomial-time solvable. At SoCG 2025, three independent works presented algorithms for this problem, each achieving a running time of $O(kn^5 \log n)$ (or $O(n^6 \log n)$), where k is the size of an optimal solution. Interestingly, these results were obtained using entirely different approaches, yet all led to roughly the same asymptotic complexity, suggesting that such a running time might be inherent to the problem.

We show that this is not the case. In the `realRAM`-model, the prevalent model in computational geometry, we present an $O(n \log n)$-time algorithm, achieving an $O(kn^4)$ factor speed-up over the previous state-of-the-art. We also give a straightforward sorting-based lower bound by reducing from the set intersection problem. We thus show that the Contiguous Art Gallery problem is in $\Theta(n \log n)$.

Acknowledgements. The authors wish to thank Frank Staals and Jack Spalding-Jamieson for their helpful discussions.

Funding. This work was supported by the the VILLUM Foundation grant (VIL37507) "Efficient Recomputations for Changeful Problems" and by the Carlsberg Foundation, grant CF24-1929.

1 Introduction

The Art Gallery problem is a classical problem in computational geometry. Given a simple polygon P with n vertices, the task is to compute the smallest set of guards such that every point $p \in P$ is visible to at least one guard. The problem was first posed by Victor Klee at a meeting of the Mathematical Association of America in 1973 and later formalized by Chvátal [17], who proved that $\lfloor n/3 \rfloor$ guards always suffice and that this combinatorial bound is tight.

Since its introduction, the problem has been extensively studied. O'Rourke and Supowit [35] showed that the problem is NP-hard if P is allowed to have holes, and Lee and Lin [30] strengthened this result by proving NP-hardness even when P is a simple polygon without holes. Eidenbenz, Stamm, and Widmayer [21] established APX-hardness, and Bonnet and Miltzow [10] showed that the problem is W[1]-hard when parametrized by the number of guards. Finally, Abrahamsen, Adamaszek, and Miltzow [2] proved that the problem is $\exists\mathbb{R}$-complete.

Problem variants. The Art Gallery problem has given rise to an exceptionally large number of problem variants. We briefly review three families of problem variants to illustrate two points: (1) there exists a remarkably large number of problem derivatives that are frequently studied, and (2) almost all of these versions are computationally intractable. When a polynomial-time algorithm exists, it typically has a high-degree polynomial running time.

The first family of variants restricts the structure of the polygon P. Krohn and Nilsson [28] showed that the problem remains NP-hard when P is xy-monotone. Schuchardt and Hecker [39] established NP-hardness for orthogonal polygons, and Tomás [42] extended this to the case of thin orthogonal polygons. Eidenbenz, Stamm, and Widmayer [21] proved NP-hardness even when P is restricted to be a 1.5D terrain (an x-monotone polygonal chain that is closed by a base edge).

[1]Department of Theoretical Computer Science, IT University of Copenhagen, Denmark
[2]Department of Computer Science, University of Copenhagen, Denmark

A second line of work constrains where guards may be placed. Lee and Lin [30] already proved NP-hardness when guards are restricted to the polygon vertices. Rieck and Scheffer [37] studied a dispersive variant where guards must maintain a minimum distance from one another, and showed NP-hardness as well. Related to this paradigm is the *watchman problem*, in which one or more guards move along paths inside the polygon. This problem also admits several sub-variants, the majority of which are NP-hard [16, 19, 34]. Those that can be solved in polynomial time require high-polynomial algorithms, with the best known running times of $O(n^4)$ [12, 15, 41].

A third family of variants modifies the notion of visibility. Bhattacharya, Kumar, and Bodhayan [5] considered *edge guards*, where a point of P is visible if it can be seen from some point on a guarding edge. Mahdavi, Seddighin, and Ghodsi [32] studied edge guards for rectilinear polygons in which new guarding edges may be introduced in P. Biedl and Mehrabi [7] analyzed rectilinear visibility, where two points see each other if there exists a rectilinear path between them contained in P. All of these variants were shown to be NP-hard. Worman and Keil [44] studied the same rectilinear visibility model but restricted P to be rectilinear also; they obtained an $\tilde{O}(n^{17})$ polynomial-time algorithm. Hoorfar and Bagheri [26] considered rectangle vision, where two points $p, q \in P$ see each other if there exists a rectangle contained in P having p and q as opposite corners. This problem can be solved in $\tilde{O}(n^{17})$ time for simple polygons but becomes NP-hard when holes are allowed [26]. Hillberg, Krohn, and Pahlow [25] showed that Art Gallery remains NP-hard when guards can only see to the right. Vaezi, Roy, and Ghodsi [43] studied visibility with mirrors that allow guards to see around corners, proving that several of these variants are also NP-hard.

Low-polynomial variants. Only a few Art Gallery variants are solvable in low-polynomial time, and all known results rely on very strong structural restrictions. Lee and Preparata [31] gave a linear-time algorithm to decide whether a polygon can be guarded by a single guard. Hoorfar and Bagheri [27] presented a linear-time algorithm for rectangle vision when P is a histogram. Palios and Tzimas [36] restrict each guard's visibility region to an orthogonal r-star and obtained a near-linear time algorithm. Biedl and Mehrabi [6] combined several restrictions—P must be orthogonal [39], thin [42], and vision rectilinear [7]—and achieved a linear-time solution. Chen, Estivill-Castro, and Urrutia [14] gave a linear-time algorithm for 1.5D terrains when guards can only see to the right.

Contiguous Art Gallery Among the many known variants, some are motivated by geometric simplicity or mathematical curiosity, while others arise naturally from physical considerations. Laurentini [29] suggested guarding only the boundary of the polygon, based on the observation that in most art galleries, artworks are displayed on walls. This leads to the question: 'Can we compute a smallest set of guards such that every point on ∂P is visible to at least one guard?'

Laurentini observed that this problem is NP-hard, and Stade [40] recently proved that it is $\exists\mathbb{R}$-complete. Thomas Shermer proposed the closely-related Contiguous Art Gallery problem. Here, each guard g can only guard a contiguous segment $[u, v]$ of ∂P. The goal is to compute the smallest set of such contiguous guards whose visibility intervals together cover ∂P (see Figure 1).

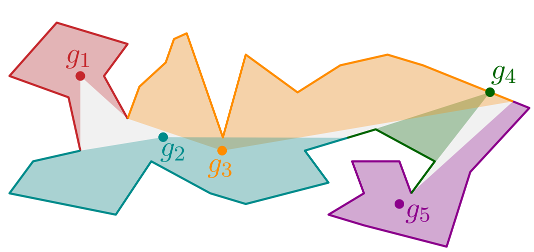

Figure 1: A simple polygon P and five contiguous guards that guard the entire boundary of P.

At SoCG 2025, three independent submissions presented `realRAM` algorithms for this problem, each with comparable high-polynomial running times but based on entirely different methods. Let k denote the size of the optimal solution. Biniaz et al. [9] analyzed the combinatorial structure of contiguous guards. They constructed a candidate set $\mathbb{C}$ of $O(n^4)$ guards and guarantee that there exists an optimal solution that uses one of the guards from $\mathbb{C}$. They show how to compute $\mathbb{C}$, and how to compute from these an optimal solution n $O(kn^5 \log n)$ total time.

Merrild, Rysgaard, Schou, and Svenning [33] gave an $O(kn^5 \log n)$-time greedy algorithm. For any point $u \in \partial P$, let $\texttt{next}(u)$ denote the farthest point v along ∂P such that there exists a guard $(g, [u, v])$. Starting from u, one can recursively apply `next` until the resulting set $\{[u, v]\}$ covers ∂P. Such a sequence defines a *revolution*, which they show yields a solution of size at most $k + 1$. They compute a revolution in $O(n^2 \log n)$ time and prove that after $O(kn^3)$ recursive revolutions, one must obtain an optimal solution, leading to an overall $O(kn^5 \log n)$-time algorithm.

Robson, Spalding-Jamieson, and Zheng [38] took an analytical approach based on the same $\texttt{next}(u)$ function. They parametrize the boundary ∂P by $[1, 2n + 1)$, where points i and $i + n$ correspond to the same vertex of P. Then we can treat `next` as a function from $[1, n+1)$ to $[1, 2n+1)$. They proved that ∂P can be partitioned into $O(n^3)$ intervals, where `next` is of constant-complexity in each interval. They extend this to an $O(n^6 \log n)$-time algorithm.

A subsequent joint version [8] presented an overview of these three approaches but did not provide a unified algorithm. Thus, there exist three independent approaches that use high polynomial time.

Contribution. At first sight, CONTIGUOUS ART GALLERY appears to follow the same pattern as most ART GALLERY variants: guarding ∂P in general is hard, and the restricted version admits high-polynomial solutions. The existence of three independent near-$O(kn^5)$ time solutions may even suggest that this is the intrinsic problem complexity. In this paper, we show that this is not the case as we present an $O(n \log n)$-time algorithm. Our `realRAM` algorithm uses linear space and achieves an $O(kn^4)$ factor speed-up over the previous state-of-the-art.[1] We further prove that this bound is optimal for all comparison-based models of computation in which sorting requires $\Omega(n \log n)$ time, which includes the `realRAM`. Our result thus provides the first near-linear optimal algorithm for a variant of the ART GALLERY problem that is neither artificially over-constrained nor degenerate in structure. Our algorithm combines ideas from all previous approaches, each of which contains a useful combinatorial component that can be exploited for efficiency. Prior works enumerated these components. We replace these brute-force steps with efficient sweep-line algorithms, and combine these insights into a unified efficient algorithm.

Biniaz et al. [9] constructed a candidate set $\mathbb{C}$ of $O(n^4)$ contiguous guards such that there exists an optimal solution that uses one guard from $\mathbb{C}$. We show that a linear-size set suffices. We compute in $O(n \log n)$ time a linear-size set X of points $u \in \partial P$ such that there exists an optimal solution containing a contiguous guard $(g, [u, v])$. Running the revolution algorithm of Merrild et al. [33] for each $x \in X$ would then yield an optimal solution in $O(n^3 \log n)$ time. However, their method effectively enumerates all interval pairs of a certain arrangement in P, leading to the high polynomial factor.

We avoid this enumeration by showing how to compute all values $\texttt{next}(x)$ for $x \in X$ in $O(n \log n)$ total time using a second sweep around ∂P. Applying this second sweepline algorithm k times yields an $O(kn \log n)$-time algorithm. To remove the remaining factor of k, we compute the revolutions for all $x \in X$ *implicitly*. To achieve this, we use the algebraic view of the `next` function introduced by Robson, Spalding-Jamieson, and Zheng [38], who showed that `next` can be represented as a piecewise function with $O(n^3)$ constant-complexity pieces. Their approach constructs these pieces explicitly in $O(n^3)$ time by partitioning ∂P into intervals where `next` is defined by three vertices.

[1] A preliminary version of this paper appeared online with an $\tilde{O}(kn^2)$-time algorithm.

We revisit this construction and show that the number of distinct function pieces is in fact much smaller. This part of the paper is the most technical: we prove that the `next` function can be represented as a piecewise function with only $O(n)$ constant-complexity pieces, and that this representation can be computed in $O(n \log n)$ total time. Using a lazy segment tree evaluation technique due to Aggarwal et al. [4], we can then evaluate all revolutions from $x \in X$ simultaneously and obtain the optimal solution in $O(n \log n)$ time using $O(n)$ space. A particular point of emphasis for our approach is that the running time uses a single logarithmic factor. We show that our algorithms must maintain a convex hull under several types of updates. The best known general-purpose dynamic convex hull structure supports updates in $O(\log^2 n)$ time. We can get by with an existing first-in-first-out structure to avoid any additional log-factors.

2 Preliminaries

Our input is a simple polygon P with n edges. Formally, P is the closed region bounded by a simple closed curve consisting of n vertices defining n edges. A vertex of P is a *reflex vertex* if its interior angle exceeds π. We denote for two points $s, t \in P$ the shortest path from s to t, which is a polygonal chain, by $S(s,t)$. We denote by ∂P the boundary of P and assume that ∂P is given in counter-clockwise order. Each edge is an ordered pair $\overline{u\,v}$ following this counter-clockwise ordering, i.e., it is directed from u to v. A point lies *strictly to the right* of $\overline{u\,v}$ if it lies in the open half-plane to the right of the directed line through u and v. A point *to the right* of $\overline{u\,v}$ may lie on this line.

Parameterizing ∂P. We assume that for any integer $i \in [1, n-1]$, the consecutive vertices v_i and v_{i+1} of P appear counter-clockwise. Consequently, the interior of P lies immediately left of $\overline{v_i\,v_{i+1}}$. For convenience, we define a continuous surjective function $[1, 2n+1) \to \partial P$ such that for every vertex v_j, both j and $n+j$ map to v_j. Thus, each point on ∂P can be represented by two real values: one in $[1, n+1)$ and one in $[n+1, 2n+1)$.

Using this parametrization, we define (open) chains as follows. A *chain* is a sequence of edges. For two points $u, v \in \partial P$, we denote by $[u, v]$ the chain obtained by traversing ∂P from u to v in counter-clockwise order. We denote by (u, v) the *open chain*, consisting of all points $x \in [u, v]$ with $x \neq u$ and $x \neq v$; that is, $(u, v) := \{\, x \in [u, v] \mid x \neq u,\ x \neq v \,\}$. Observe that the chain between any two points on ∂P can be described as $[u, v]$ with $u \in [1, n+1)$ and $v \in [u, 2n+1)$.

Problem statement. A point x *sees* a point y if the segment $\overline{x\,y}$ is contained in P. A *(contiguous) guard* is a tuple $(g, [u, v])$ consisting of a point $g \in P$ and a chain $[u, v] \subset \partial P$ such that every point on the chain is visible from g.

Problem Statement 1 (Contiguous Art Gallery). Given a simple polygon P with n vertices, compute a minimum-size set of contiguous guards G such that their corresponding chains cover the entire boundary interval $[1, n+1]$.

We say that a guard $(g, [u, v])$ *dominates* another guard $(g', [u', v'])$ if $[u', v'] \subset [u, v]$. It *strictly dominates* $(g', [u', v'])$ if $[u', v'] \subsetneq [u, v]$. For a fixed $u \in [1, n+1)$, we frequently compute a *maximal* $v \in [u, 2n+1)$ such that there exists a point $g \in P$ for which $(g, [u, v])$ is a guard. By this we mean that for all other guards $(g', [u, v'])$, it holds that $v' \in [u, v]$.

Visibility core. For any chain $[u, v]$ between two points $u, v \in \partial P$, we define the *visibility core* $\mathcal{E}[u, v]$ as the set of points $p \in \mathbb{R}^2$ that lie left of all edges in $[u, v]$. We can use the visibility core to decide whether a single guard suffices.

Observation 1. If $(g, [u, v])$ is a guard, then g lies left of all edges of P intersecting the open chain (u, v). In particular, g lies in $\mathcal{E}[u, v]$. Moreover, there exists a guard that can see all of ∂P if and only if $\mathcal{E}[1, n+1]$ is non-empty.

3 Technical Overview

Our primary contribution is a tight analysis of the Contiguous Art Gallery problem in the `realRAM`-model. We present an $O(n \log n)$-time algorithm, achieving a speedup by a factor of $\tilde{O}(kn^4)$ compared to previous works [8, 9, 33, 38]. We complement this by a relatively simple lower bound of $\Omega(n \log n)$ in the `realRAM`-model.

Our algorithmic improvement arises from three key ideas. For any $x \in [1, n+1)$, let $\texttt{next}(x)$ denote the maximal value v such that there exists a guard $(g, [u, v])$. Our first idea is to construct a set X of points along ∂P with the property that there exists some $x \in X$ for which the greedy algorithm—obtained by recursively applying $\texttt{next}(\cdot)$—yields an optimal solution. We construct a set X of linear size and compute it in $O(n \log n)$ time. This result alone improves the running time of [8, 33] to $O(n^3 \log n)$, as it suffices to compute what they call a revolution for each $x \in X$.

Our second idea is a sliding-window algorithm that, for an ordered pair of indices (i, j), repeatedly increments i or j until $i = n+1$. We use the sliding window to compute $\texttt{next}(x)$ for all $x \in X$ in $O(n \log n)$ total time using only $O(n)$ space. By recursively applying the sliding-window procedure, we obtain an $O(kn \log n)$-time algorithm.

Our final and most technical idea refines this approach. We partition ∂P into intervals such that, for each interval $[x_1, x_2]$, the function $\texttt{next}(x)$ on the domain $[x_1, x_2]$ can be described by a constant-complexity function. This observation was present in [8, 38], where they use $O(n^3)$ intervals which they compute in near-cubic time. We show that $O(n)$ such intervals suffice, and compute them in $O(n \log n)$ time. We can combine these functions with our sliding-window procedure to create an algorithm that computes, for all $x \in X$, their full greedy sequences under the `next` function in $O(n \log n)$ total time—producing an optimal solution to the Contiguous Art Gallery problem.

3.1 Upper bound

We begin with a concept that appears in several previous approaches [8, 9, 33, 38]. For any $u \in \partial P$, let $v \in [u, 2n+1)$ be the maximal value such that there exists a guard $(g, [u, v])$. This defines a function $\texttt{next} : \partial P \to \partial P$ mapping u to v. Let k denote the minimum number of guards required to cover ∂P. If we choose any point $x \in \partial P$ and apply `next` a total of $k+1$ times, the resulting point lies beyond $x + n$. Our goal is to compute a set $X \subset \partial P$ such that for at least one $x \in X$ there exists an optimal solution of k guards containing a guard $(g, [x, v])$. If we recursively apply the `next` function k times to all points in X, this yields an optimal solution using k guards. Hence, if X has linear size and if we can evaluate `next` in $O(\log n)$ time, we obtain an $O(kn \log n)$-time algorithm.

Classifying guards. Section 4 introduces the structural results to compute X and the `next` function. We define three collections of highly structured guards such that every guard is dominated by one of these three types. Specifically, we partition the set of all guards into *good* and *bad* guards. Intuitively, a guard $(g, [u, v])$ is *good* if the angle $\sphericalangle(u, g, v)$ is at most π, and *bad* otherwise. We then define three collections of *dominators*: the good, the bad, and the ugly.

Good dominators are very well-structured: for any ordered pair of indices (i, j) there exists at most one good dominator, implying that there are $O(n^2)$ such dominators overall. Moreover, every good guard is dominated by a good dominator. Bad dominators are slightly less structured: for each ordered pair (i, j), we consider the polygon $\mathcal{E}[i-1, j+1]$ and define a guard $(g, [u_{\max}, v_{\max}])$ for each vertex g of $\mathcal{E}[i-1, j+1]$, where $[u_{\max}, v_{\max}]$ is some maximal chain visible to g. Since $\mathcal{E}[i-1, j+1]$ has $O(n)$ vertices, there are $O(n^3)$ bad dominators. The set of ugly dominators is not as susceptible to discretization. Instead, we prove that each bad guard is dominated by either a bad or an ugly dominator, where the relevant ugly dominator can be computed on the fly.

We can now provide the high-level intuition for why $\texttt{next}(x)$ can be evaluated in logarithmic time. If we precompute all good and bad dominators $(g, [u, v])$, then we can store their corresponding intervals $[u,v]$ in a segment tree T. For any $x \in \partial P$, we performing a stabbing query on T and we return the maximal right endpoint r among the intervals stabbed by x. Let $v = \texttt{next}(x)$. If there exists a guard $(g,[u,v])$ dominated by a good or bad dominator, then $v = r$. We then compute an ugly dominator for x on the fly and compare its result to r to obtain the correct value of $\texttt{next}(x)$.

For this approach to run in $O(n \log n)$ time, the segment tree must contain only $O(n)$ dominators. To ensure this, we define the *reduced good dominators* $\mathcal{D}$ as those good dominators that are not *strictly* dominated by another good dominator. Similarly, we define the *reduced bad dominators* $\mathcal{B}$ as the bad dominators that are not *strictly* dominated by *any* other guard. These distinct definitions matter because the reduced good dominators serve a dual purpose:

Defining a set X. The reduced good dominators $\mathcal{D}$ are not only essential for computing $\texttt{next}$, but they also induce the set X. Specifically, we prove that there always exists an optimal solution Opt to the Contiguous Art Gallery problem such that at least one guard $(g,[u,v]) \in$ Opt either belongs to $\mathcal{D}$ or has u as a vertex of P. We define X as the set of all vertices of P, together with all points $u \in \partial P$ such that there is a reduced good dominator $(g,[u,v])$.

Sliding windows. We use a sliding window over indices (i,j) to compute (a superset of) $\mathcal{D}$. The remainder of our $O(n \log n)$-time algorithms rely on a different but very specific sliding window:

Definition 1. We define a *sliding sequence* as an ordered set of index pairs $\{(i,j)\}$ of linear size where, for every pair of consecutive elements $((i,j),(i',j'))$, we have $(i'-i,\, j'-j) \in \{(1,0),(0,1)\}$. Given a polygon P, a sliding sequence σ is said to be *conforming*, if for each $u \in [1,n+1)$, with $u \in [i-1,i)$ and $\texttt{next}(u) \in (j,j+1]$, then $(i,j) \in \sigma$.

Let σ be a conforming sliding sequence. Each bad dominator has a defining index pair (i,j). Since reduced bad dominators are not strictly dominated by *any* other guard, every guard in $\mathcal{B}$ has its corresponding index pair $(i,j) \in \sigma$. From this observation we upper bound $|\mathcal{B}|$. Consider iterating over all $(i,j) \in \sigma$. By point-line duality, maintaining the convex visibility core $\mathcal{E}[i-1,j+1]$ during this iteration corresponds to maintaining the convex hull of a point set under first-in–first-out updates. Chan, Hershberger, and Pratt [13] show that the convex hull of such an update sequence of length n has $O(n)$ vertices. This implies an algorithm to dynamically maintain $\mathcal{E}[i-1,j+1]$ subject to incrementing i and j, and that $|\mathcal{B}| \in O(n)$. It remains to compute such a σ, and derive the sets $\mathcal{D}$ and $\mathcal{B}$, in $O(n \log n)$ time.

Computing σ and the reduced dominators. In Section 6 we compute linear-size sets $\mathcal{D}'$ and $\mathcal{B}_\sigma$ where $\mathcal{D} \subset \mathcal{D}'$ and $\mathcal{B} \subset \mathcal{B}_\sigma$. We first compute $\mathcal{D}'$ using a sliding-window algorithm. Note that we cannot use a conforming sliding sequence σ for this task, as a reduced good dominator may be strictly dominated by a guard that is not a good dominator. Consequently, the defining pair (i,j) for a guard $(g,[u,v]) \in \mathcal{D}$ need not belong to σ.

For all subsequent algorithms, we use a conforming sliding sequence σ, which we compute in an online fashion starting from $(i,j) = (1,1)$ and repeatedly incrementing either i or j. The choice of which index to increment follows a simple rule: while there exists a guard $(g,[u,v])$ such that $[i,j+1] \subset [u,v]$, we increment j and add the new pair (i,j) to σ; otherwise, we increment i and add the new pair (i,j) to σ. By the result of [13], we can maintain $\mathcal{E}[i-1,j+2]$ as a balanced binary tree while incrementing i and j. We show that, given i, j, and $\mathcal{E}[i-1,j+2]$, this decision can be made in $O(\log n)$ time, yielding an overall $O(n \log n)$ algorithm for constructing σ. From σ, we derive a linear-size set $\mathcal{B}_\sigma$ containing $\mathcal{B}$. We store the visibility of guards in $\mathcal{D}'$ and $\mathcal{B}_\sigma$ in an segment tree T. Thus, we may obtain for any point $x \in \partial P$ the guard $(g,[u,v]) \in \mathcal{D}' \cup \mathcal{B}_\sigma$ that has $x \in [u,v]$ and maximizes v.

An $O(kn \log n)$ algorithm. In Section 7, we consider the set X and iterate over all $(i, j) \in \sigma$. For each $x \in X$, we perform a stabbing query on T to find the maximum value r such that there exists a guard $(g, [x, r]) \in \mathcal{D}' \cup \mathcal{B}_\sigma$. The value $\texttt{next}(x)$ differs from r only if it is realised by an ugly dominator. The value $\lfloor r \rfloor$ serves as a lower bound for the index j such that $\texttt{next}(x) \in (j, j+1]$, and by definition $(i, j) \in \sigma$. These observations yield an algorithm that iterates over σ and computes $\texttt{next}(x)$ for all $x \in X$ in $O(n \log n)$ total time. Repeating this process k times produces an optimal solution in $O(kn \log n)$ time. This is already a major improvement over the state-of-the-art [8, 9, 33, 38] and serves as a crucial subroutine for our $O(n \log n)$-time algorithm.

An $O(n \log n)$ algorithm. We compute a representation of $\texttt{next}$ as a piecewise, possibly degenerate, Möbius transformation. This approach was first proposed in [8, 38], where the resulting function consisted of $O(n^3)$ pieces and was computed in $\tilde{O}(n^5)$ time. In contrast, we use our dominators to define a function with only $O(n)$ pieces, which can be computed in $O(n \log n)$ time. Specifically, in Section 8, we partition the boundary ∂P into intervals I such that either:

- a reduced good dominator $(g, [u, v])$ sees $[x, \texttt{next}(x)]$ for all $x \in I$, so $\texttt{next}(x) = v$;
- a reduced bad dominator $(g', [u', v'])$ sees $[x, \texttt{next}(x)]$ for all $x \in I$, so $\texttt{next}(x) = v'$; or
- for all $x \in I$, $\texttt{next}(x) = \frac{A+Bx}{C+Dx}$ for constants A, B, C, D depending only on I.

As there are only $O(n)$ reduced good and bad dominators, the first two cases introduce at most $O(n)$ interval boundaries. For our final partitioning, we first prove a weaker statement: there are at most $O(n)$ intervals $\mathcal{U}$ where for all $x \in \mathcal{U}$, no reduced good or bad dominator sees $[x, \texttt{next}(x)]$, i.e., only an ugly dominator does.

We first prove that the vertices of the shortest path from x to $\texttt{next}(x)$ behave in a structured way, unless the optimal solution consists of at most 3 guards. These vertices play a key role in defining the ugly dominator g_u, and the resulting structural properties guarantee both the existence and computability of the $O(n)$ Möbius transformations in $O(n \log n)$ total time. Finally, in Section 9, we combine all components to obtain our $O(n \log n)$ algorithm. We apply our $O(kn \log n)$ algorithm to check whether a solution with $k \leq 3$ guards exists, in $O(n \log n)$ time. If not, we use the techniques of Aggarwal et al. [4] to compute the smallest k such that there exists a point $x \in X$ for which $\texttt{next}^k(x) \geq x + n$, or conversely, $[x, \texttt{next}^k(x)] = \partial P$, where $\texttt{next}^k(\cdot)$ is the composition of $\texttt{next}$ with itself k times, all within $O(n \log n)$ time.

3.2 Lower bound

In Section 10, we reduce from Set Disjointness, where we are given two sets $A, B \subset [0, n^3]$, each of size n, and must decide whether $A \cap B = \emptyset$. While in the $\texttt{wordRAM}$-model there is a deterministic $O(n \log \log n)$-time algorithm for solving this problem (via sorting [23]), in comparison-based models of computation such as the $\texttt{realRAM}$-model, it has a worst-case lower bound of $\Omega(n \log n)$, even if A is already sorted [45]. Given $A, B \subset [1, n^3]$ with A sorted, we can in linear time construct a simple polygon P such that $A \cap B = \emptyset$ if and only if P can be guarded with $2|B| + 2$ guards (see Figure 2).

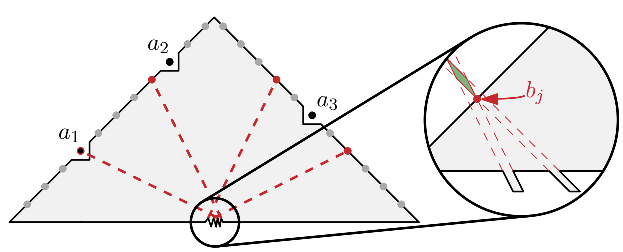

Figure 2: The lower-bound construction. Black points belong to A, and red points to B.

We begin with a triangle T whose top edges, $t_1 : (0,0)$ to $(\frac{n^3}{2}, \frac{n^3}{2})$ and $t_2 : (\frac{n^3}{2}, \frac{n^3}{2})$ to $(n^3, 0)$, represent the interval $[0, n^3]$. We construct the polygon in cyclical clockwise order, starting from $(0,0)$. For each of the increasing values $a_i \in A$, we make an indent in $t_1 \cup t_2$ excluding the point corresponding to a_i from P. Next, we partition a small segment on the bottom edge ($(n^3, 0)$ to $(0,0)$) into $|B|$ intervals I_j. For each $b_j \in B$, we introduce a *pocket*: two consecutive indentations of I_j such that a guard can see I_j if and only if it stands at the point of t_1 or t_2 corresponding to b_j. Hence, there exists a guard that can cover I_j if and only if $b_j \notin A$. Finally, we place a *blocker* between consecutive intervals I_j and I_{j+1} such that each I_j and each blocker requires a unique guard. The resulting polygon P can be constructed in linear time and guarded with $2|B| + 2$ contiguous guards if and only if $A \cap B = \emptyset$, establishing an $\Omega(n \log n)$ lower bound for the Contiguous Art Gallery problem in comparison-based models of computation such as the `realRAM`-model.

4 A combinatorial classification of guards

We define three types of guards such that every guard is dominated by one of these three types. To this end, we first partition the set of all guards into *good* and *bad* guards. In Figure 1, g_2, g_3, and g_4 are good, while g_1 and g_5 are bad.

Definition 2. Let u and v be points on ∂P with $u < v$. A contiguous guard $(g, [u, v])$ is *good* if $g \neq u$ and $g \neq v$ and the angle $\sphericalangle(u, g, v) \leq \pi$, and *bad* otherwise.

We will define three collections of guards called *dominators*: the good, the bad, and the ugly. We prove that all good guards are dominated by a good dominator, which will induce a set X of $O(n)$ points along ∂P such that there exists an optimal solution that includes a guard $(g, [x, v])$ for $x \in X$. We prove that all bad guards are dominated by either a bad or an ugly dominator. The flowchart in Figure 3 illustrates our approach. To this end, we first make some observations:

Lemma 2. *Let $(g, [u, v])$ be a guard where, for the fixed vertex u, v is maximal. Then either v is a reflex vertex of P or $\overline{v\,g}$ contains a reflex vertex of P in its interior.*

Proof. By maximality of v, $g \neq v$ and there exists an $\varepsilon^* > 0$ such that $\forall \varepsilon \in (0, \varepsilon^*]$ the shortest path from g to $v + \varepsilon$ visits a reflex vertex x. If $x = v$ then v is a reflex vertex of P. Otherwise, $\overline{v\,g}$ contains x in its interior. ◻

Lemma 3. *Let $(g, [u, v])$ be a bad guard where, given g, $[u, v]$ is inclusion-wise maximal. Then the shortest path from u to v in P is a left-turning chain. Furthermore, if u or v is not a vertex of P, then this left-turning chain has at least one interior vertex. Moreover, if neither u nor v is a vertex, this chain has two interior vertices.*

If instead both g and u are given, and v is maximal, then the shortest path from u to v in P is still a left-turning chain. Furthermore, if v is not a vertex of P, then this left-turning chain has at least one interior vertex.

Proof. For a guard that is inclusion-wise maximal it cannot be that $g = u$ or $g = v$. That the shortest path from u to v is left-turning follows from the fact that g can see u and v and $\sphericalangle(u, g, v) > \pi$. If u (respectively v) is not a vertex of P, then Lemma 2 implies that the edge $\overline{u\,g}$ (respectively $\overline{v\,g}$) contains a reflex vertex of P in its interior, which must appear as an interior vertex on the shortest path from u to v.

If only v is maximal, then it again cannot be that $g = v$, but it may be that $g = u$. In this case the shortest path from u to v is the segment $\overline{u\,v}$, which is also left-turning. If v is not a vertex of P, Lemma 2 still implies there is at least one vertex on this shortest path. ◻

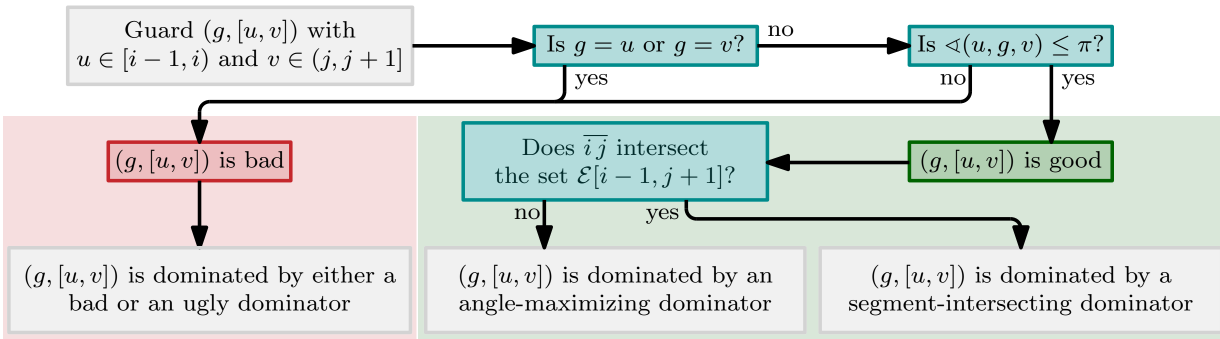

Figure 3: Flowchart illustrating how a guard is dominated. We prove the green block in Lemma 5 and the red block in Lemma 6.

4.1 The good dominators

We show that there exists a set $\mathcal{D}$ of $O(n)$ guards, called the *reduced good dominators*, such that every good guard is dominated by one of them. We begin by defining two categories of *good dominators*, each of size $O(n^2)$.

Observation 4. For any fixed $g \in P$ and indices i and j, with $i \leq j$, there exists at most one maximal chain $[u_{\max}, v_{\max}]$ visible from g that contains $[i, j]$.

Proof. For any fixed point $g \in P$, the set of points in P visible from g forms the *visibility polygon* of g. Its intersection with ∂P induces a fixed set of pairwise disjoint maximal chains, and only one of these chains can contain $[i, j]$. □

Definition 3. Consider an index pair (i, j) with $i \in [2, n+1]$ and $j \in [i, 2n]$. We define for (i, j) two types of guards that we call *good dominators*. Specifically, a segment-intersecting dominator (**A**) and angle-maximizing dominator (**B**):

- **A (segment-intersecting dominator):** See Figure 4. If $\mathcal{E}[i-1, j+1]$ intersects the interior of $\overline{i\,j}$, we define the guard $(g, [u_{\max}, v_{\max}])$, where g is the last point of $\overline{i\,j}$ intersecting $\mathcal{E}[i-1, j+1]$, and $[u_{\max}, v_{\max}] \subset [i-1, j+1]$ is the maximal chain containing $[i, j]$ that is visible to g.
- **B (angle-maximizing dominator):** See Figure 5. If $\mathcal{E}[i-1, j+1]$ lies left of $\overline{i\,j}$, we define the guard $(g, [u_{\max}, v_{\max}])$. We define g as i (or j) if $\mathcal{E}[i-1, j+1]$ intersects i (or j). Otherwise, g is the point in $\mathcal{E}[i-1, j+1]$ that maximizes the angle $\sphericalangle(i, g, j)$. We define $[u_{\max}, v_{\max}] \subset [i-1, j+1]$ as its maximal visible chain containing $[i, j]$.

For $i = j$, we define $(g, [u_{\max}, v_{\max}]) = (i, [i-1, i+1])$, which is a dominator of both type **A** and **B**. If there does not exist a type **A** and **B** guard for (i, j) that adheres to the stated conditions, then there is no good dominator for (i, j).

Note that by general position if $\mathcal{E}[i-1, j+1]$ is left of $\overline{i\,j}$, then it can intersect $\overline{i\,j}$ in at most a single point, so the angle-maximizing dominators are well-defined.

Definition 4. The *reduced good dominators* $\mathcal{D}$ are the good dominators not strictly dominated by another good dominator.

Next, we show that any good guard is dominated by a reduced good dominator. Our proof follows the approach illustrated in the green block of the flowchart in Figure 3.

Lemma 5. *For every good guard* $(g, [u, v])$ *there exists a reduced good dominator* $(g', [u', v']) \in \mathcal{D}$ *that dominates it.*

Proof. The guard $(g, [u, v])$ is good and thus $g \neq u, v$ and $\sphericalangle(v, g, u) \leq \pi$ by Definition 3. We prove that a good dominator (and thus a reduced good dominator) dominates $(g, [u, v])$. Let $u \in [i-1, i)$ and $v \in (j, j+1]$. If $i = j$ or $i-1 = j$, then the dominator for (i, j), which is the guard $(i, [i-1, j+1])$, dominates $(g, [u, v])$. So, assume $i < j$.

We define by P_g the polygon formed by the edges $\overline{v\,g}$, $\overline{g\,u}$ and $[u, v]$. Since $(\overline{u\,g}, \overline{v\,g})$ forms a convex wedge, and P_g contains no points of ∂P in its interior, all points in $\mathcal{E}[u, v] \cap P_g$ see all of $[u, v]$. By general position, $\mathcal{E}[u, v]$ either is left of $\overline{i\,j}$ and intersects $\overline{i\,j}$ in at most one point, or contains at least one point strictly right of $\overline{i\,j}$. We define g^* as i (or j) if $\mathcal{E}[u, v]$ contains i (or j) and is left of $\overline{i\,j}$, or as the point in $\mathcal{E}[u, v]$ that maximizes the angle $\sphericalangle(i, g^*, j)$, otherwise.

Case 1: $\boldsymbol{\sphericalangle(i, g^*, j) > \pi}$**.** The point g^* must lie strictly right of the edge $\overline{i\,j}$ and, in particular, cannot serve as an angle-maximising dominator. Instead, we construct the segment-intersecting dominator (**A**) and show that it dominates $(g, [u, v])$. Because $\sphericalangle(u, g, v) \leq \pi$, we also have $\sphericalangle(i, g, j) \leq \pi$, which, together with $\sphericalangle(i, g^*, j) > \pi$, implies that $\overline{i\,j}$ lies in P_g. In particular, g^* also lies in P_g, with $\overline{i\,j}$ separating g and g^*. By definition, both g and g^* lie in $\mathcal{E}[u, v]$. Since $\mathcal{E}[u, v]$ is convex, and $\overline{i\,j}$ separates g and g^*, $\mathcal{E}[u, v]$ must intersect $\overline{i\,j}$. Moreover, every edge of $\mathcal{E}[u, v]$ supports an edge in $[i-1, j+1]$, and as no three vertices of P are collinear, there must exist an intersection point in the interior of $\overline{i\,j}$.
Any point of intersection between $\overline{i\,j}$ and $\mathcal{E}[u, v]$ lies in both $\mathcal{E}[u, v]$ and P_g, and thus sees all of $[u, v]$. It follows that the last point of intersection g' between $\overline{i\,j}$ and $\mathcal{E}[u, v]$ sees $[u, v]$ and so the segment-intersecting dominator $(g', [u_{\max}, v_{\max}])$ corresponding to (i, j) is well-defined and it dominates $(g, [u, v])$.

Case 2: $\boldsymbol{g^* = i}$ **or** $\boldsymbol{g^* = j}$ **or** $\boldsymbol{\sphericalangle(i, g^*, j) \leq \pi}$**.** If we can show that g^* is in P_g, then it lies in $P_g \cap \mathcal{E}[u, v]$ and thus sees all of $[u, v]$. So, the angle-maximizing dominator $(g^*, [u_{\max}, v_{\max}])$ (**B**) corresponding to (i, j) dominates $(g, [u, v])$. If $g^* = i$ or $g^* = j$ then g^* in P_g, concluding the proof. So, assume that $g^* \neq i, j$ and for the sake of contradiction that $g^* \notin P_g$, refer to Figure 6. Let $\alpha = \sphericalangle(i, g^*, j)$. The points $x \in \mathbb{R}^2$ such that $\sphericalangle(i, x, j) = \alpha$ form a circular arc passing through i and j. As no other point in $\mathcal{E}[u, v]$ has larger angle than α, and $\mathcal{E}[u, v]$ is convex, the tangent t of this circular arc at g^* separates the plane into two half-planes: a closed half-plane H_g that contains $\mathcal{E}[u, v]$ and an open half-plane $H_{i,j}$ containing the points i and j (if i or j lies on t then they coincide with g^*, contradicting our assumptions).

Now, consider the edges of $\mathcal{E}[u, v]$ that intersect g^*. As $\mathcal{E}[u, v]$ is contained in H_g, and $g \in \mathcal{E}[u, v]$, the supporting line ℓ of one of these edges must pass between g and t. As P_g is contained in the wedge defined by g, u, v, the line ℓ intersects P_g only in H_g. Note that $g^* \notin P_g$ implies that g^* lies strictly left of $\overline{u\,v}$ and so at least one of u or v is in $H_{i,j}$.

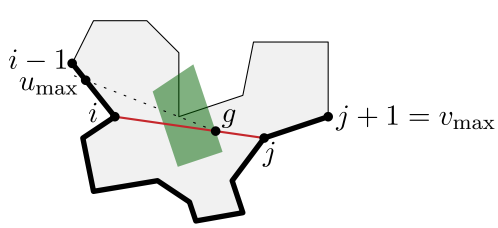

Figure 4: The visibility core $\mathcal{E}[i-1, j+1]$ in green and the segment-intersecting dominator $(g, [u_{\max}, v_{\max}])$ for (i, j).

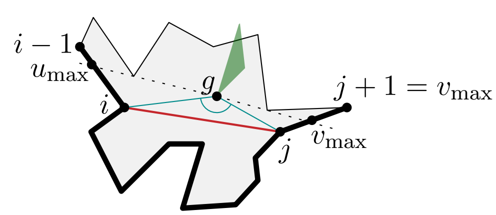

Figure 5: The visibility core $\mathcal{E}[i-1, j+1]$ in green and the angle-maximizing dominator $(g, [u_{\max}, v_{\max}])$ for (i, j).

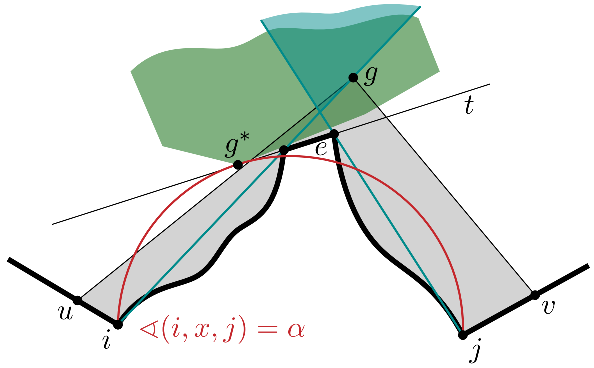

Figure 6: Case 2 of Lemma 5. The point g^* that has $\sphericalangle(i, g^*, j) = \alpha$ is in the visibility core $\mathcal{E}[u, v]$ (green), but not in P_g (gray).

Let $e \in E$ be the edge defining ℓ. By definition, e lies on ℓ and thus ℓ separates e from i and j. Consider the directed line ℓ_j from the right endpoint of e to j, and consider the chain from e's right endpoint to j. This chain must contain at least one edge e^* whose start lies left of ℓ_j and whose endpoint right right of ℓ_j (recall that we distinguish between right and *strictly* right). It follows that $\mathcal{E}[u, v]$ lies left of ℓ_j. Defining ℓ_i analogously yields a wedge bounded by ℓ_i and ℓ_j that contains $\mathcal{E}[u, v]$ (see the blue region in Figure 6). The fact that one of u or v is in $H_{i,j}$ implies that the point of intersection between ℓ_i and ℓ_j lies in H_g and not on t. Moreover, g also lies in H_g and not on t, thus no point on t is in $\mathcal{E}[u, v]$, which contradicts g^* being in $\mathcal{E}[u, v]$. □

4.2 Bad and ugly dominators

In addition to good dominators, we also define bad and ugly dominators. Bad dominators are defined for each index pair (i, j) in an analogous manner to good dominators:

Definition 5 (Bad dominators ($\star$))**.** Consider an index pair (i, j) with $i \in [2, n+1]$ and $j \in [i, 2n]$. For every vertex c of $\mathcal{E}[i-1, j+1]$, we define at most one bad dominator. Each vertex c of $\mathcal{E}[i-1, j+1]$ is defined by edges $[a-1, a]$ and $[b, b+1]$. Let $[u_{\max}, v_{\max}]$ be the maximal chain containing $[a-1, b+1]$ that is visible from c.

- If $[u_{\max}, v_{\max}]$ exists and is non-empty, then we define for c the bad dominator $(c, [u_{\max}, v_{\max}])$.

Definition 6. The *reduced bad dominators* $\mathcal{B}$ are the bad dominators that are not strictly dominated by any other guard.

Finally, the ugly dominators are defined a bit differently as they are defined off of an existing bad guard $(g, [u, v])$:

Definition 7 (Ugly dominators (✎))**.** Let $(g, [u, v])$ be a bad guard with $v = \texttt{next}(u)$ and $u \in [i-1, i)$ and $v \in (j, j+1]$. Let S be the shortest path from u to $j+1$ within P, and let ℓ be the supporting line of the first edge. If ℓ intersects $\mathcal{E}[i-1, j+1]$, let g^* be the last intersection point along ℓ, and let v^* be the farthest point on $[j, j+1]$ visible from g^*.

- If g^* and v^* exist then we define the *ugly dominator* $(g^*, [u, v^*])$.

Lemma 6. *For any bad guard $(g, [u, v])$ where $u \in [i, i+1)$ and $v = \texttt{next}(u) \in (j, j+1]$, there exists either a bad dominator ($\star$) or an ugly dominator (✎) that dominates $(g, [u, v])$.*

Proof. Let now S_v be the shortest path from u to v and denote by ℓ_u the directed supporting line of its first edge. By Lemma 3, S_v forms a left-turning chain. Let S be the shortest path from u to $j+1$. Since $(g,[u,v])$ has maximal v, Lemma 2 states that $\overline{gv}$ contains a reflex vertex of P. If $v \neq j+1$, $\overline{gv}$ contains a reflex vertex in its interior. This implies the first edge of S_v and S are the same, and thus the line ℓ_u equals ℓ. If $v = j+1$, then S_v is equal to S, so also ℓ equals ℓ_u. Denote by P' the convex polygon that is $\mathcal{E}[i-1,j+1]$ intersected with the half-plane right of ℓ_u, see Figure 7. Denote by c a point in P' that sees farthest along $[j,j+1]$. We can choose c to be a vertex of P'. We make a case distinction:

(i) If c is also a vertex of $\mathcal{E}[i-1,j+1]$ then c defines a bad dominator ($\star$) which is the guard $(c,[u_{\max},v_{\max}])$. By maximality, $[u,v]$ must be contained in $[u_{\max},v_{\max}]$.

(ii) if c is not a vertex of $\mathcal{E}[i-1,j+1]$ then c must be an intersection point of $\ell_u = \ell$ and $\mathcal{E}[i-1,j+1]$. As ℓ and $\mathcal{E}[i-1,j+1]$ intersect, the ugly dominator $(g^*,[u,v^*])$ exists, where g^* is the last point of intersection between ℓ and $\mathcal{E}[i-1,j+1]$. As the chain S_v is a left-turning chain, the intersection point of ℓ and $\mathcal{E}[i-1,j+1]$ that sees furthest along the edge $[j,j+1]$ is the last point of intersection, i.e. $g^* = c'$. As c' sees at least as far as g, because $g \in P'$, it must be that $v^* \geq v$. ☐

4.3 The `next` function and our high-level algorithm

Define by X a set of points along ∂P that contains all vertices of P and for all guards $(g,[u,v]) \in \mathcal{D}$ the point u. We show that we can compute an optimal solution to the Contiguous Art Gallery problem by starting at a guard in the set of reduced good dominators $\mathcal{D}$ or at a vertex of P, and greedily applying the 'next best' guard.

Definition 8 (The `next` function [8,38])**.** For $u \in [1,n+1]$, $\texttt{next}(u)$ returns the largest $v \in [u,2n+1]$ where there exists a guard $(g,[u,v])$.

Theorem 7. *Let G be a minimal set of contiguous guards guarding ∂P with $|G| > 1$. There exists a set of contiguous guards G' with $|G| = |G'|$ where at least one $(g,[u,v]) \in G'$ is either in $\mathcal{D}$, or has u a vertex of P.*

Proof. We show how to transform G into such a set G' (see also Figure 8). We start by replacing every guard $(g,[u,v]) \in G$ by a guard $(g,[u',v'])$ such that $[u,v] \subset [u',v']$ and the interval $[u',v']$ is maximal to form G'. Next, we show that we can adapt G' such that either there is a good guard in G' or a guard whose visibility ends at a vertex of P. Lemma 5 implies that this good guard is dominated by a guard in $\mathcal{D}$, concluding the proof.

Suppose that G' contains only bad guards and let $(g,[u,v]) \in G'$ be such a bad guard. If (u,v) does not contain a vertex of P then $[u,v] \subset [i,i+1]$ for some vertex i. We now replace $(g,[u,v])$

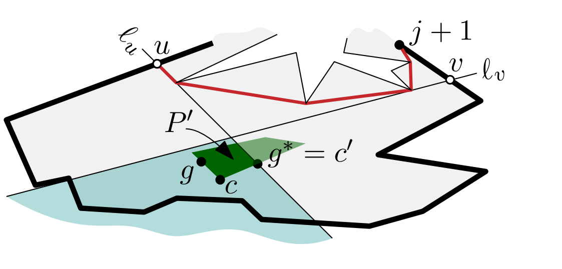

Figure 7: The shortest path from u to $j+1$ (red), and $\mathcal{E}[i-1,j+1]$ (green). Any point that sees both u and v lies in the blue wedge.

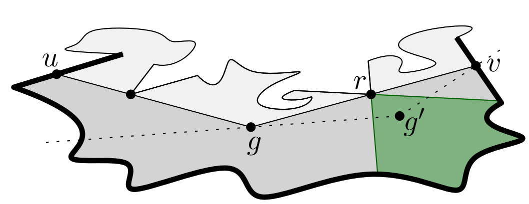

Figure 8: Illustration of Theorem 7 which shows a bad contiguous guard $(g,[u,v])$.

by $(i, [i, i+1])$ and conclude the proof. Similarly, if v is a vertex of P then, because $|G| > 1$, there has to be at least one other guard $(g', [u', v']) \in G'$ such that $v \in [u', v']$. We replace this guard by $(g', [v, v'])$ so that the start of its chain is a vertex of P which concludes the proof. Lastly, note that it is not possible for g to be u (resp. v), as in that case, g sees all edges of P containing $g = u$ (resp. $g = v$), and in particular, strictly more than $[u, v]$, contradicting the maximality of $[u, v]$.

It follows that for all bad guards $(g, [u, v]) \in G'$, the open chain (u, v) contains a vertex of P, while neither u nor v is a vertex of P. By Lemma 2, the segment $\overline{v\,g}$ contains a reflex vertex r of P with $r \notin [u, v]$. Hence, there exists a guard $(g', [u', v'])$ such that $r \in [u', v']$. If $u' = r$ or $v' = r$, we conclude the proof as above. Otherwise, if $r \in (u', v')$, then g' lies left of both supporting lines of the edges incident to r. This implies that g' lies left of $\overline{v\,g}$. Since $\sphericalangle(v, g, u) > \pi$, any point that sees r and lies left of $\overline{v\,g}$ must also lie within the polygon P_g defined by $[u, v]$, $\overline{v\,g}$, and $\overline{g\,u}$ (see Figure 8). Observe that the points in $[u', v'] \setminus (u, v)$ form a single closed chain; otherwise, $[u, v] \subset [u', v']$, contradicting the minimality of G'. Define $[a, b] = [u', v'] \setminus (u, v)$. If we replace $(g', [u', v'])$ by $(g', [a, b])$, the resulting set of guards still covers ∂P. Crucially, $[a, b]$ lies left of $\overline{g'\,v}$ and left of either the supporting line of $\overline{u\,g'}$ or $\overline{g\,g'}$. As g' lies in P_g and $\sphericalangle(v, g, u) > \pi$, this implies that, unless $g' = a$ or $g' = b$, $\sphericalangle(a, g', b) \leq \pi$, and thus G' contains a good guard. Finally, if $g' = a$ (resp. $g' = b$), then g' is on ∂P and in particular, sees the entirety of the edge containing a (resp. b), hence we can extend $[a, b]$ to start (or end) in a vertex, which concludes the proof. ☐

The following theorem gives the high-level idea of the running times achieved in Sections 5 to 7.

Theorem 8. *Let P be a simple polygon of n vertices and X be a linear-size set that includes all polygon vertices and for all $(g, [u, v]) \in \mathcal{D}$, the point u. Given P, X and an $O(T)$-time implementation of the* `next` *function, we can compute an optimal solution of size k to the* Contiguous Art Gallery *problem in $O(knT)$ time.*

Proof. We apply to each $x \in X$ the function $\texttt{next}(x)$ recursively until a set of guards covering ∂P is obtained. Each such sequence contains at most $k+1$ guards and this thus takes $O(k\,T \log n)$ time.

We claim that one of these guard sequences must have size k. By Theorem 7 there exists at least one guarding solution of size k that contains a guard $(g, [x, v])$ where the guard is either in $\mathcal{D}$ or x is a vertex of P. A classical argument now implies that recursively applying $\texttt{next}(x)$ finds a solution of size k also. For completeness, we write the argument here: Let $G = \{(g_i, [v_{i-1}, v_i])\}$ be minimum ordered sequence of guards that guards ∂P that is generated by recursively applying $\texttt{next}(x)$. Then $(g_1, [x_1, x_2]) = (g, [x, \texttt{next}(x)])$.

Suppose, for the sake of contradiction, that $|G| > k$. Let G^* be an optimal solution of size k that includes $(g, [x, v])$ and order this solution along ∂P, starting at $(g, [x, v])$. Since $|G| > |G^*|$, there must exist a minimum index i such that for the i^{th} guards $(g_i, [v_{i-1}, v_i]) \in G$ and $(g_i^*, [u_i^*, v_i^*]) \in G^*$, it holds that $v_i < v_i^*$. However, since i is minimum, it must be that $v_{i-1} \geq u_i^*$. But then $v_{i-1} \in [u_i^*, v_i^*]$ which makes $\texttt{next}(v_{i-1})$ return a value that is at least v_i^*—a contradiction. ☐

5 Intermezzo: data structures and sliding sequences

Our results rely on several classical geometric results concerning simple polygons.

Data Structure 1 ([22]). A polygon P with n vertices can be stored in $O(n)$ time and space so that, for any $s, t \in P$, the shortest path $S(s, t)$ can be reported as $O(\log n)$ balanced trees in $O(\log n)$ time. Moreover:

- The path $S(s, t)$ is represented by $O(\log n)$ balanced trees whose ordered leafs traverse $S(s, t)$.
- We can find the first edge e along $S(s, t)$ where the path makes a left turn around the first vertex of e and a right turn around the last vertex of e, in $O(\log n)$ time.

Proof. The data structure of [22] stores a family of shortest paths in P as balanced binary trees. For any query points $s, t \in P$, the shortest path $S(s,t)$ can be expressed as $O(\log n)$ subtrees, together with $O(\log n)$ newly computed edges called *bridges*.

The second property follows from a standard adaptation of this data structure (see also [20]): for each pair of consecutive edges on a precomputed shortest path, determine whether the turn is left or right. For each node in the tree, store a symbol indicating whether all of its descendant edges are left-turning, right-turning, or mixed. For a query, we can determine for each of the $O(\log n)$ newly computed bridges whether the turn at its endpoints is left or right. Combined with the pre-stored symbols, this information suffices to answer the second query. □

Data Structure 2 ([24]). A polygon P with n vertices can be stored in $O(n)$ time and space such that, for any ray r whose origin lies in P, the point on ∂P hit by r can be found in $O(\log n)$ time.

Throughout the paper, we assume that we have access to Data Structure 1 and Data Structure 2 for the given polygon. We repeatedly use the concept of a *sliding sequence*. A sliding sequence can be thought of as a pair of indices (i,j) that can be updated by incrementing either i or j. Formally we define in Definition 1 a conforming sliding sequence used by almost all of our algorithms. For each (i,j), we maintain the visibility core $\mathcal{E}[i-1, j+1]$ as a convex-hull data structure supporting emptiness, containment, extreme-point, and ray-shooting queries, and an additional *angle query* (defined later). For any linear-size sliding sequence $\{(i_t, j_t)\}$, Chan, Hershberger, and Pratt [13] give a data structure supporting such updates and queries in $O(\log n)$ time:

Lemma 9 (Paraphrased Lemma 1 in [13]). *For any linear-size sliding sequence σ, the total number of distict vertices that get added or removed from the visibility core $\mathcal{E}[i-1, j+1]$ when iterating over $(i,j) \in \sigma$ is linear.*

Lemma 10. *Consider a traversal of any sliding sequence σ, each point p appears as a vertex on the visibility core in at most one contiguous time interval.*

Proof. Let v be a vertex of the visibility core $\mathcal{E}[i-1, j+1]$ that is not a vertex of $\mathcal{E}[i'-1, j'+1]$, where (i', j') follows (i,j) in σ. There are two possible reasons for v not being a vertex of $\mathcal{E}[i'-1, j'+1]$: either one of its supporting edges has been removed from the sliding window, or a new half-plane has been added to the visibility core that excludes v. In the first case, since removed edges never reappear in later windows, v cannot return. In the second case, the new half-plane remains in the sliding window at least as long as the edges defining v, again preventing v from reappearing. Hence, each vertex appears in at most one contiguous interval. □

The above two lemmas imply the following, see also the remarks made in [13].

Corollary 11. *One can maintain the intersection of half-planes, explicitly, stored as a balanced binary tree, with leaves in cyclical order, under first-in-first-out updates using linear space and an amortized update time $O(\log n)$.*

We subsequently implement a very specific data structure that we use throughout the paper:

Data Structure 3. We define Sliding-Window data structure (3) as a data structure that for a pair of indices (i,j) stores the visibility core $\mathcal{E}[i-1, j+1]$ and $\mathcal{E}[i-1, j+2]$, supporting the following operations in (amortized) $O(\log n)$ time:

- **Increment i:** Delete the corresponding half-plane.
- **Increment j:** Insert a new half-plane.
- **Emptiness query:** Determine whether $\mathcal{E}[i-1, j+1]$ is empty and return an arbitrary point inside it if not.

- **Containment query:** For a point x, determine of $x \in \mathcal{E}[i-1, j+1]$.
- **Extreme-point query:** For a given vector w return the $x \in \mathcal{E}[i-1, j+1]$ maximizing the dot product $\langle w, x \rangle$.
- **Ray-shooting query:** Return the last point of intersection between a ray ℓ (a directed half-line) and $\mathcal{E}[i-1, j+1]$.
- **Angle query:** Given u, v with $\mathcal{E}[i-1, j+1]$ left of $\overline{uv}$, find $g \in \mathcal{E}[i-1, j+1]$ that maximizes the angle $\sphericalangle(u, g, v)$.
- We also support all these queries for $\mathcal{E}[i-1, j+2]$. Whenever we perform a query on $\mathcal{E}[i-1, j+2]$ instead of $\mathcal{E}[i-1, j+1]$ we refer to such a query as a *lookahead query.*

Theorem 12. *The* Sliding-Window *data structure (3) can be implemented with $O(n)$ preprocessing and $O(n)$ space, it supports queries in $O(\log n)$ time and updates in amortized $O(\log n)$ time.*

Proof. Let (i, j) be the dynamic index pair from the Sliding-Window data structure (3). We maintain $\mathcal{E}[i-1, j+1]$ and $\mathcal{E}[i-1, j+2]$ separately, subject to increments of i and j via Corollary 11. This maintains the vertices of these convex areas as balanced binary trees where the left-to-right traversal of the leaves corresponds to a clockwise traversal. For a a convex polygon with n vertices, whose vertices are stored in a balanced tree according to their cyclic order, the containment, ray shooting, and extreme-point queries can be answered in $O(\log n)$ time [18]. We can thus immediately support all queries except for angle queries.

For angle queries, conceptually rotate and translate the plane until $\overline{uv}$ corresponds with the horizontal line through $(0, 0)$ with the area left of $\overline{uv}$ lying above this line. For any fixed angle $\gamma > 0$, the points $\vec{x} \in \mathbb{R}^2_{\geq 0}$ for which $\sphericalangle(u, \vec{x}, v) = \gamma$ forms a convex arc. For γ_1, γ_2 with $\gamma_1 > \gamma_2$ the corresponding convex arc lies strictly above the arc induced by γ_2. It follows that the function $\vec{x} \mapsto -\sphericalangle(u, \vec{x}, v)$ defined for all points $\vec{x} \in \mathbb{R}^2_{\geq 0}$ is well-defined, unimodal in $\mathcal{E}[i-1, j+1]$, and strictly monotone between adjacent edges of $\mathcal{E}[i-1, j+1]$. Since we have the edges of $\mathcal{E}[i-1, j+1]$ explicitly, in-order, in a balanced binary tree, it then follows that the point in g that maximizes $\sphericalangle(i, g, j)$ lies on the boundary of $\mathcal{E}[i-1, j+1]$ and that we can find it via performing binary search along the edges of $\mathcal{E}[i-1, j+1]$ in $O(\log n)$ time. ☐

6 Computing all reduced dominators

We now demonstrate an immediate application of Observation 1 and our data structures, which we will frequently use to compute the reduced dominators:

Lemma 13. *Let $g \in P$ be a fixed point, and let $s, m, t \in \partial P$ with $m \in [s, t]$. Suppose g lies in $\mathcal{E}[s, t]$. Then there exists a unique maximal chain $[u, v] \subset [s, t]$ that is visible to g and contains m, which can be found in $O(\log n)$ time.*

Proof. If g does not see m, we detect this in $O(\log n)$ time using Data Structure 1 and return the empty interval. Let S_s be the shortest path from g to s, and let r_s denote the ray along its first edge. Define S_t and r_t analogously. The three elements (r_s, g, r_t) form a wedge. We claim that the points u and v of ∂P that are respectively hit by r_s and r_t define the desired maximal chain. We can find these points in $O(\log n)$ time using two ray-shooting queries (Data Structure 2).

Consider the polygon P' bounded by S_s, $[s, t]$, and S_t. Since g sees m, the segment $\overline{gm}$ splits P' into two polygons, $P_s \subset P$ and $P_t \subset P$, containing s and t respectively (see Figure 9). Let u^* be the maximal point in $[s, m]$ such that $\overline{g\,u^*}$ intersects S_s in more than one point (i.e., not only in g). We first show that g sees all of $[u^*, m]$. Indeed, consider the polygon P_{u^*} bounded by $[u^*, m]$, $\overline{m\,g}$, and $\overline{g\,u^*}$. Because $g \in \mathcal{E}[s, t]$, it lies left of all edges of P_{u^*} and thus, by Observation 1, g sees all of $[u^*, m]$. By construction, P_{u^*} contains no point of S_s in its interior and hence lies within $P_s \subset P$. Therefore, g indeed sees $[u^*, m]$ in P.

If $s \neq u^*$, then g does not see any point $u' \in [s, u^*)$, since by definition of u^*, the segment $\overline{g\,u'}$ intersects an edge of S_s in its interior. It remains to show that $u^* = u$. Let r^* be the ray from g through u^*. We claim that $r^* = r_s$. Indeed, since $\overline{g\,u^*}$ intersects S_s, if this intersection occurs at a point r not lying on the last edge of S_s, then S_s could be shortened by the segment $\overline{r\,g} \subset \overline{u^*\,g} \subset P$. This contradicts the definition of S_s as a shortest path. Hence $r^* = r_s$, and we conclude that $u^* = u$. By symmetry, the same argument applies to r_t, yielding the maximal visible chain $[u, v]$. ☐

We compute two linear-size sets $\mathcal{D}'$ and $\mathcal{B}_\sigma$ that contain all reduced good and reduced bad dominators, respectively.

6.1 A linear-size set $\mathcal{D}'$ that contains all reduced good dominators in $\mathcal{D}$

We compute a linear-size set $\mathcal{D}'$ that contains all guards in the set of reduced good dominators $\mathcal{D}$ (Definition 6) via a sliding window. Formally, we maintain a pair of indices (i, j) subject to incrementing i and incrementing j. During this procedure, we maintain $\mathcal{E}[i-1, j+1]$ (and $\mathcal{E}[i-1, j+2]$) via SLIDING-WINDOW data structure (3) and we prove that this data structure can answer an advanced query:

Lemma 14. *Given vertices i and j with $i \leq j$, and $\mathcal{E}[i-1, j+1]$ in SLIDING-WINDOW data structure (3), we can compute the unique good dominator for (i, j) (see Definition 3), or return that no good dominator exists for (i, j), in $O(\log n)$ time.*

Proof. We perform an emptiness query on $\mathcal{E}[i-1, j+1]$. If the visibility core $\mathcal{E}[i-1, j+1]$ is empty then by definition no dominator exists for (i, j).

We first use extreme-point queries in the two directions orthogonal to $\overline{i\,j}$ to test for points in $\mathcal{E}[i-1, j+1]$ strictly left and right of $\overline{i\,j}$. If there is no point strictly right of $\overline{i\,j}$ then the entirety of the visiblity polygon lies left of $\overline{i\,j}$. In this case, we can, via angle queries and containment queries at i and j, compute the point g^* that is either i or j, if i or j is in $\mathcal{E}[i-1, j+1]$, and the point in $\mathcal{E}[i-1, j+1]$ that maximizes the angle $\sphericalangle(i, g^*, j)$. If instead the visibility core has both a point left of $\overline{i\,j}$, and a point strictly right of $\overline{i\,j}$, we compute a point g' that is in the interior of $\overline{i\,j}$ and in $\mathcal{E}[i-1, j+1]$, via two ray shooting queries on $\mathcal{E}[i-1, j+1]$ using the the supporting ray from i to j, and the ray from j to i. If there is no point left of $\overline{i\,j}$, or $\mathcal{E}[i-1, j+1]$ has a point strictly right of $\overline{i\,j}$, but only intersects $\overline{i\,j}$ in either i or j, then by definition there is no good dominator for (i, j). Note that at most one of g^* or g' exists.

We apply Lemma 13 to compute the maximal interval $[u_{\max}, v_{\max}]$ that is visible to g^* (or g') that includes i in logarithmic time. If $[i, j] \subset [u_{\max}, v_{\max}]$ then we return this guard as the segment-intersecting or angle-maximizing dominator for i, j. Otherwise, this procedure certified that no such dominator exists. ☐

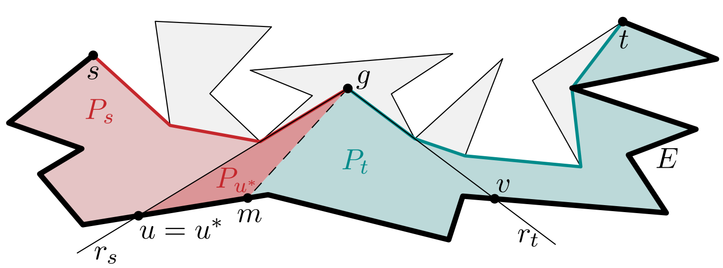

Figure 9: The inclusion-wise maximal area visible from g and containing m is bounded by the rays r_s and r_t along the first edges of the shortest paths from g to s and t, respectively. These rays hit ∂P in points u and v that define the maximal chain visible to g.

Lemma 15. *Suppose that for an index pair (i, j) with $i \in [2, n+1]$ and $j \in [i, 2n]$, there exists no good dominator. Then there exists no good dominator for (i, j') with $j' > j$.*

Proof. For any pair (i', j'), define the *potential dominator* as a point g as follows: If the interior of $\overline{i'\,j'}$ intersects $\mathcal{E}[i'-1, j'+1]$, then g is the last point of intersection between $\mathcal{E}[i'-1, j'+1]$ and $\overline{i'\,j'}$. If $\mathcal{E}[i'-1, j'+1]$ lies left of $\overline{i'\,j'}$ and intersects i or j, then g is equal to i or j, respectively. If $\mathcal{E}[i'-1, j'+1]$ lies strictly left of $\overline{i'\,j'}$, then g is the point in the visibility core that maximizes $\sphericalangle(i', g, j')$. If none of these cases apply, then g is undefined. The remainder of the proof proceeds by case distinction, depending on whether g is undefined for (i, j) or whether g does not realize a good dominator. Observe that a good dominator for (i, j) exists if and only if the potential dominator g exists and its maximal visible chain $[u_{\max}, v_{\max}]$ satisfies $u_{\max} \in [i-1, i]$ and $v_{\max} \in [j, j+1]$.

Case 1: $\mathcal{E}[i-1, j+1]$ is empty. Then $\mathcal{E}[i-1, j'+1]$ is also empty for all $j' > j$.

Case 2: $\mathcal{E}[i-1, j+1]$ is not empty but g is undefined. We claim that for all pairs (i, j') with $j' \geq j$, either this property also holds or $\mathcal{E}[i-1, j'+1] = \emptyset$. Since the vertices of P lie in general position and $\mathcal{E}[i-1, j+1]$ does not intersect the interior of $\overline{i\,j}$, the visibility core $\mathcal{E}[i-1, j+1]$ must be contained in the polygon P' defined by $\overline{i\,j}$ and $[i, j]$.

Consider any $j' > j$. We first assume that j' lies strictly right of $\overline{i\,j}$ and distinguish two subcases based on the position of j'. Figure 10: Suppose that j' is not in P'. Then no point in $\mathcal{E}[i-1, j+1]$ can see j' and so there exists no good dominator for (i, j'). Figure 11: Suppose j' is in P'. Consider the subset of $\mathcal{E}[i-1, j+1]$ that is left of $\overline{i\,j'}$. Let a be the last point of intersection of $[j, j']$ and $\overline{i\,j}$. For any point $p \in \mathcal{E}[i-1, i+1]$ that lies in the polygon B defined by $\overline{i\,j'}$, $[j', a]$, and $\overline{a\,i}$, it must be that $p \notin \mathcal{E}[i-1, j'+1,]$. So consider a point $p \in \mathcal{E}[i-1, i+1]$ that does not lie in B (not in the blue area in Figure 11). If p lies strictly left of $\overline{i\,j'}$ then p cannot see i, thus p cannot a the good dominator for (i, j'). Furthermore, the only point of $\overline{i\,j'}$ that can lie in $\mathcal{E}[i-1, j'+1]$ is the vertex j' itself. It follows that no segment-intersection dominator (**A**) exists for (i, j'). If $\mathcal{E}[i-1, j'+1]$ would be left of $\bar{i}j'$ then there would be an angle-maximizing dominator (**B**) for (i, j'). However, then either $[j'-1, j']$ is collinear with i, which contradicts our general position assumption, or j' cannot see i. We conclude that no good dominator exists for (i, j').

Suppose otherwise that j' lies left of $\overline{i\,j}$. Then because P lies in general position, j' must lie strictly left of this line. However, then the interior of $\overline{i\,j'}$ lies outside of the polygon P' defined by $\overline{i\,j}$ and $[i, j]$. We observed that $\mathcal{E}[i-1, j+1]$ is contained in P' and since $\mathcal{E}[i-1, j'+1] \subset \mathcal{E}[i-1, j+1]$ it follows that no segment-intersecting dominator (**A**) exists for (i, j'). Finally if j' lies left of $\overline{i\,j}$ and $\mathcal{E}[i-1, j'+1]$ is strictly left $\overline{i\,j'}$ then no point in $\mathcal{E}[i-1, j'+1]$ can see j' (see Figure 12), Thus, there also exists no angle-maximizing dominator (**B**) for (i, j').

Case 3: the potential dominator g exists for (i, j), but its maximal visible chain $[u_{\max}, v_{\max}]$ satisfies $u_{\max} \notin [i-1, i]$ or $v_{\max} \notin [j, j+1]$. We reduce this case to Case 2. Let $P^{\uparrow}$ denote the subpolygon of $\mathcal{E}[i-1, j+1]$ left of $\overline{i\,j}$. Consider any point $p \in P^{\uparrow}$. If p sees both i and j, then $(p, [i, j])$ forms a good guard (it satisfies Condition II). By Lemma 5, there then exists a good dominator for (i, j) that dominates $(p, [i, j])$, contradicting the assumption of the lemma. Hence, no point in $P^{\uparrow}$ sees both i and j. Since any potential dominator for (i, j') with $j' > j$ must see at least $[i, j]$, we may regard the visibility core $\mathcal{E}[i-1, j+1]$ as excluding $P^{\uparrow}$. This adjusted visibility core now lies strictly right of $\overline{i\,j}$ and thus has no potential dominator. Consequently, the situation reduces to Case 2. ☐

Theorem 16. *For a simple polygon P of n vertices, we can compute a linear-size superset $\mathcal{D}'$ of size $O(n)$ of guards that contains the reduced good dominators $\mathcal{D}$ using $O(n)$ space and $O(n \log n)$ time.*

Proof. We maintain a pair of indices (i, j) as a sliding window. We either increment i or increment j, maintaining $\mathcal{E}[i-1, j+1]$ and $\mathcal{E}[i-1, j+2]$ via Sliding-Window data structure (3) with amortized

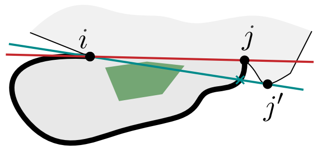

Figure 10: Any guard in the visibility core (green) will not see j'.

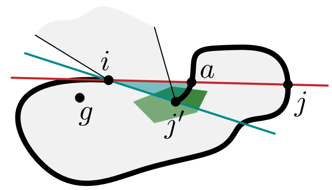

Figure 11: No point in the blue polygon can be in the visibility core $\mathcal{E}[i-1, j'+1]$.

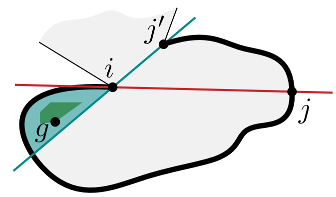

Figure 12: The potential angle-maximizing dominator for (i, j') does not see j'.

$O(\log n)$ update time. We also a set $\mathcal{D}'$ subject to the following invariant:

- For any $i' < i$, or $i' = i$, and $j' < j$, if there exists a reduced good dominator for (i', j') then it is in $\mathcal{D}'$.

Our strategy. Let (i, j) be the current index pair. We apply Lemma 14 to find out whether a good dominator $(g, [u_{\max}, v_{\max}])$ exists for (i, j). If it exists, then Lemma 14 returns it and we add it to $\mathcal{D}'$. We then increment either i or j via the following strategy:

- If no good dominator exists for (i, j) then we increase i by one.
- If we add a guard $(g, [u_{\max}, v_{\max}])$ to $\mathcal{D}'$, and $v_{\max} = j + 1$, then we increase j by one.
- Finally, if we do add a guard $(g, [u_{\max}, v_{\max}])$ to $\mathcal{D}'$ and $v_{\max} \in [j, j+1)$ then we test whether for $(i, j+1)$ we would add a guard $(g', [u'_{\max}, v'_{\max}])$ to $\mathcal{D}'$. This test happens in the exact same manner as for (i, j), using the *lookahead* queries for Sliding-Window data structure (3). If, for $(i, j+1)$, we will add a guard to $\mathcal{D}$, then we increment j. Otherwise, we increment i.

From the invariants, it follows that when starting from $(i, j) = (2, 2)$ until we reach $i = n + 1$ or $j = 2n$, we have found a set of good dominators of size $O(n)$. As both i and j are bounded by 2, the algorithm increases i or j at most $O(n)$ times. It follows that the total running time is $O(n \log n)$.

Correctness. We prove that we maintain our invariant. If we ever observe a pair (i, j) for which there exists no good dominator, then by Lemma 15 we note that there exists no good dominator for (i, j') with $j' \geq i$ and we may safely increment i in this case.

If we observe a pair (i, j) for which there exists a good dominator $(g, [u_{\max}, v_{\max}])$ and $v_{\max} = j+1$, then for all $i' \in [i+1, j]$ the good dominator of (i', j) is dominated by $(g, [u_{\max}, v_{\max}])$ and we may safely increment j in this case.

If there exists a dominator $(g, [u_{\max}, v_{\max}])$ for (i, j) where $v_{\max} < j + 1$, then we check whether there exists a good dominator for $(i, j+1)$. If so, then any good dominator for (i', j) with $i < i' < j$ will be dominated by this dominator and the invariant is thus maintained when increasing j. If not, then the previous argument implies that there is also no good dominator for $j' > j + 1$. It follows that the invariant is maintained after incrementing i. □

6.2 Computing a conforming sliding sequence σ

Recall the definition of a conforming sliding sequence:

Definition 1. We define a *sliding sequence* as an ordered set of index pairs $\{(i, j)\}$ of linear size where, for every pair of consecutive elements $((i, j), (i', j'))$, we have $(i' - i,\ j' - j) \in \{(1, 0), (0, 1)\}$. Given a polygon P, a sliding sequence σ is said to be *conforming*, if for each $u \in [1, n+1)$, with $u \in [i-1, i)$ and $\texttt{next}(u) \in (j, j+1]$, then $(i, j) \in \sigma$.

Such a conforming sliding sequence σ is used for all our underlying algorithms. We show that we can compute σ in $O(n \log n)$ time. We first show that we can check efficiently whether any guard exists for some given chain $[s,t]$ that lies inside a given visibility core.

Lemma 17. *Given vertices i and j with $i \leq j$ and $\mathcal{E}[i-1, j+1]$ in SLIDING-WINDOW data structure (3), let $s \in [i-1, i]$ and $t \in [j, j+1]$. Given s, t, we can decide whether there exists a guard $(g, [s,t])$ with $g \in \mathcal{E}[i-1, j+1]$ in $O(\log n)$ time.*

Proof. If $s = t$ then we simply output the guard $(s, [s,t])$. Otherwise, define i^* as i if $s < i$ and as $i+1$ if $s = i$. Define j^* similarly as j if $t > j$ and as $j-1$ if $t = j$. If $i^* = j^*$ then i^* can see both s and t and we output the guard $(i^*, [s,t])$. Otherwise, we know that $i^* < j^*$. We compute a constant-size set of candidate points $\mathbb{C} \subset \mathcal{E}[i-1, j+1]$.

Computing the candidate set $\mathbb{C}$. We deploy a very similar strategy to Lemma 5. We perform an emptiness query on $\mathcal{E}[i-1, j+1]$ which, if the visibility core is non-empty, returns a point γ. If the visibility core is empty we output that there exists no guard that guards $[s,t]$. We perform two ray shooting queries using the supporting ray from i^* to j^*, and the ray from j^* to i^*, in $O(\log n)$ time. If both rays do not hit $\mathcal{E}[i-1, j+1]$ then $\mathcal{E}[i-1, j+1]$ lies either strictly right or strictly left of $\overline{i^* j^*}$. We compare γ to this supporting line to test in constant time whether the visibility core $\mathcal{E}[i-1, j+1]$ is strictly left of $\overline{i^* j^*}$. If so, then we define g^* as the point in $\mathcal{E}[i-1, j+1]$ that maximizes the angle $\sphericalangle(i^*, g^*, j^*)$, which we can compute in $O(\log n)$ time using an angle maximizing query. We can also use both ray shooting queries to detect whether the interior of $\overline{i^* j^*}$ intersects $\mathcal{E}[i-1, j+1]$, and if it does, we define g' as the last point of intersection of the open directed segment $\overline{i^* j^*}$ and $\mathcal{E}[i-1, j+1]$.

Finally, we compute the shortest path $S(s,t)$ from s to t and test whether $S(s,t)$ is left-turning. If $S(s,t)$ is left-turning then we take the supporting halflines ℓ_s and ℓ_t of the first and last edge of $S(s,t)$ (these lines are directed away from s and t, respectively). We compute g_1 and g_2: the respective last points of intersection of these halflines with $\mathcal{E}[i-1, j+1]$ in $O(\log n)$ time. The set $\mathbb{C}$ is formed by γ, g^*, g', g_1 and g_2 (if the respective point exists). For each $g \in \mathbb{C}$, we observe that $g \in \mathcal{E}[i-1, j+1] \subset \mathcal{E}[s,t]$ and so we invoke Lemma 13 to compute whether g can see $[s,t]$ in $O(\log n)$ time. If none of the points in $\mathbb{C}$ has this property, we output that no guard $(g, [s,t])$ exists.

Correctness. Suppose that no good guard $(g, [s,t])$ with $g \in \mathcal{E}[i-1, j+1]$ exists. Then if any guard $(g, [s,t])$ with $g \in \mathcal{E}[i-1, j+1]$ exists it must be that $\sphericalangle(s, g, t) > \pi$. This in turn implies that $S(s,t)$ is a left-turning chain, and so a point $p' \in \mathcal{E}[i-1, j+1] \subset \mathcal{E}[s,t]$ can see $[u,v]$ if and only if p' lies right of ℓ_s and left of ℓ_t. If there exists at least one point $p' \in \mathcal{E}[i-1, j+1]$ right of ℓ_s and left of ℓ_t, then one of $\{\gamma, g_1, g_2\}$ exists and lies right of ℓ_s and left of ℓ_t. It follows that a guard $(g, [s,t])$ exists if and only if our algorithm outputs such a guard. Suppose otherwise that a good guard $(g, [s,t])$ with $g \in \mathcal{E}[i-1, j+1]$ exists. Then $\sphericalangle(i^*, g, j^*) \leq \pi$. We make a case distinction.

Case 1: g^* does not exist. Via an identical argument as in Lemma 5, we then get that $\mathcal{E}[i-1, j+1]$ must intersect the interior of $\overline{i^* j^*}$. In this case, any point of intersection between $\mathcal{E}[i-1, j-1]$ and $\overline{i^* j^*}$, and in particular our precomputed point g', can see $[s,t]$.

Case 2: g^* exists. If g^* exists, then the visibility core $\mathcal{E}[i-1, j+1]$ is strictly left of $\overline{i^* j^*}$ by definition, and thus $\sphericalangle(i^*, g^*, j^*) < \pi$. We now define the polygon P_g that is formed by $[s,t]$ and $\overline{g\, s}$ and $\overline{t\, g}$. The point g^* by definition lies in $\mathcal{E}[i-1, j+1] \subset \mathcal{E}[s,t]$. If g^* is in P_g then it lies in $P_g \cap \mathcal{E}[s,t]$ and thus sees all of $[s,t]$. We now have an identical construction to Case 2 of Lemma 5 and conclude that g^* can see $[s,t]$.

Conclusion. If a good guard $(g, [s,t])$ with $g \in \mathcal{E}[i-1, j+1]$ exists then either g' or g^* can see $[s,t]$. If no good guard with $g \in \mathcal{E}[i-1, j+1]$ exists, then any point $p \in \mathcal{E}[i-1, j+1] \subset \mathcal{E}[s,t]$ can see $[s,t]$ if and only if p lies left of ℓ_s and right of ℓ_t. So, if none of $\{\gamma, g^*, g', g_1, g_2\}$ guard the chain $[s,t]$, then no guard in $\mathcal{E}[i-1, j+1]$ can. ☐

Theorem 18. *There exists a linear-size conforming sliding sequence σ and it can be computed in $O(n \log n)$ time.*

Proof. Recall that a sliding sequence has for consecutive pairs $((i,j),(i',j'))$ that $(i'-i, j'-j) \in \{(1,0),(0,1)\}$. Formally, we construct σ by focusing on a different property. We define the *discrete maximum* function $d_{\max}$ that takes any integer $i \leq n$ and returns the maximum integer j^* such that there exists a guard $(g,[i,j^*])$ with $g \in \mathcal{E}[i-1, j^*+1]$. Note that for any $j \in [i, d_{\max}(i)]$, the visibility core $\mathcal{E}[i-1,j^*+1] \subset \mathcal{E}[i-1,j+1]$ and so for all $j \in [i, d_{\max}(i)]$ there exists a guard $(g,[i,j])$ with $g \in \mathcal{E}[i-1,j+1]$. We prove that any (i,j) for which there exists a guard $(g,[u,\texttt{next}(u)])$ with $u \in [i-1,i)$ and $\texttt{next}(u) \in (j, j+1]$ is included in any sliding sequence σ^* that includes all pairs (i,j) with $j \in [i, d_{\max}(i)]$ where 'no strictly better' pair (i',j') exists. Formally, we construct a sliding sequence σ^* such that:

- σ^* contains all (i,j) with $j \in [i, d_{\max}(i)]$ where there does not exist an $i' < i$ and $j' > j$ with $j' \in [i', d_{\max}(i')]$.

We claim that σ^* is conforming. Indeed, for fixed indices (i,j), if there exists a guard $(g,[u,\texttt{next}(u)])$ with $u \in [i-1,i)$ and $\texttt{next}(u) \in (j,j+1]$, then there also exists a guard $(g',[i,j])$ with $g' \in \mathcal{E}[i-1,j+1]$ and so $j \in [i, d_{\max}(i)]$. Thus $(i,j) \in \sigma^*$ if there exists no strictly better pair (i',j'). Suppose for the sake of contradiction that there exist indices $i' < i$ and $j' > j$ with $j' \in d_{\max}(i')$. Then by definition, there exists a guard $(g'',[i',j'])$ with $g'' \in \mathcal{E}[i'-1,j'+1] \subset \mathcal{E}[i-1,j+1]$ but this contradicts the maximality of $\texttt{next}(u)$ for our original guard g.

Computing σ^*. We maintain (i,j) subject to incrementing either i or j and maintain the following invariants:

1. We store $\mathcal{E}[i-1,j+1]$ in Sliding-Window data structure (3).
2. The sequence σ^* that has been computed thus-far has $(i,j-1)$ as its last element.
3. There exists a guard $(g,[i,j-1])$ with $g \in \mathcal{E}[i-1,j]$.
4. The sequence σ^* is complete up to (i,j). More formally, σ^* computed thus-far contains all index pairs (i_0,j_0) that have both of the following two properties:
   - $j_0 \in [i_0, d_{\max}(i_0)]$ where there does not exist an $i' < i_0$ and $j' > j_0$ with $j' \in [i', d_{\max}(i')]$,
   - $i_0 < i$, or $i_0 = i$ and $j_0 < j$.

Every time we increment i or j, we update $\mathcal{E}[i-1,j+1]$ accordingly in $O(\log n)$ time. To decide whether to increment i or j, we invoke Lemma 17 using $s = i$ and $t = j$ to check if there exists a guard $(g,[i,j])$ with $g \in \mathcal{E}[i-1,j+1]$. If yes, then we add the pair (i,j) to σ^* and increment j. Otherwise, we add the pair $(i+1,j-1)$ to σ^* and increment i.

Correctness. If we start our sliding window at $(i,j) = (1,2)$ then our invariants immediately imply that when we reach $i = n+1$, the sequence σ^* have been computed correctly in $O(n \log n)$ time. What remains is to prove that the invariants always hold. Clearly, the first invariant is always maintained. The second invariant is maintained by construction because before we increment j, we add (i,j) to σ^* and before we increment i, we add $(i+1,j-1)$ to σ^*.

Next, consider the third invariant. If we are about to increment j, then exists a guard $(g,[i,j])$ with $g \in \mathcal{E}[i-1,j+1] \subset \mathcal{E}[i-1,j]$, thus after incrementing j the third invariant still holds. Suppose otherwise that we are about to increment i. By the third invariant, there exists a guard $(g,[i,j-1])$ with $g \in \mathcal{E}[i-1,j]$. This guard also guards $[i+1,j]$ and $g \in \mathcal{E}[i-1,j] \subset \mathcal{E}[i,j]$. So, the guard $(g,[i,j-1])$ implies that the third invariant is maintained after incrementing i.

Finally, we consider the fourth invariant. Before we are about to increment j, (i,j) is added to σ^*. Thus, if the fourth invariant holds for up to (i,j) then it must hold after adding (i,j) to σ^* and incrementing j. If we are about to increment i then there does not exist a guard $(g,[i,j])$ with

$g \in \mathcal{E}[i-1, j+1]$. Suppose for the sake of contradiction that there exists a guard $(g, [i, j'])$ with $g \in \mathcal{E}[i-1, j'+1]$ and $j' > j$. However, $g \in \mathcal{E}[i-1, j'+1] \subset \mathcal{E}[i-1, j+1]$ and $[i, j] \subset [i, j']$ – a contradiction. We now apply the third invariant to note that there exists a guard $(g, [i, j-1])$ with $g \in \mathcal{E}[i-1, j]$. The fact that there exists no guard $(g, [i, j'])$ with $g \in \mathcal{E}[i-1, j'+1]$ and $j' > j$, and one guard $(g, [i, j-1])$ with $g \in \mathcal{E}[i-1, j]$, implies that implies that $d_{\max}(i) = j-1$. It follow that there is no pair (i, j') with $j' \geq j$ for which $j' \in [i, d_{\max}(i')]$, and we can safely increment i. ☐

6.3 A linear-size set $\mathcal{B}_\sigma$ that contains all reduced good dominators in $\mathcal{B}$

Consider a conforming sliding sequence σ. We define a set $\mathcal{B}_\sigma$ and argue that all reduced bad dominators ($\mathcal{B}$) are contained in $\mathcal{B}_\sigma$. For $(i, j) \in \sigma$ we consider $\mathcal{E}[i-1, j+1]$ and note that each vertex g of $\mathcal{E}[i-1, j+1]$ is defined by two edges in $[i-1, j+1]$.

Definition 9. We denote by V_σ the set of all vertices of visibility cores $\mathcal{E}[i-1, j+1]$ for $(i, j) \in \sigma$.

Definition 10. For each $g \in V_\sigma$, defined by edges $[a-1, a]$ and $[b, b+1]$, we define the *candidate guard* $(g, [u_{\max}, v_{\max}])$ where $[u_{\max}, v_{\max}]$ is the maximum visible chain from g that includes both $[a-1, a]$ and $[b, b+1]$.

- We define the $\mathcal{B}_\sigma$ as the set of all candidate guards $(g, [u_{\max}, v_{\max}])$ for $g \in V_\sigma$.
- We define $\mathcal{B}_\sigma(i, j)$ as $\{(g, [u_{\max}, v_{\max}]) \in \mathcal{B}_\sigma \mid g \text{ is a vertex of } \mathcal{E}[i-1, j+1]\}$.

We compute $\mathcal{B}_\sigma$ by computing for all $x \in V_\sigma$ some maximal chain $[\ell_x, r_x]$, and applying Lemma 13.

Lemma 19. *Let σ be a conforming sliding window sequence, and let x be a vertex of $\mathcal{E}[i-1, j+1]$. Define $r_x = [b, b+1]$ as the first edge with $j+1 \leq b$ where x is strictly right of r_x. We can compute r_x for all $x \in V_\sigma$ in total time $O(n \log n)$.*

Proof. We iterate over all $(i, j) \in \sigma$ in order. This requires $O(n)$ updates, where each update increments either i or j. Hence, we can maintain the vertices of $\mathcal{E}[i-1, j+1]$ explicitly in memory using the first-in-first-out data structure from Corollary 11. By Lemma 10, each point appears as a vertex of $\mathcal{E}[i-1, j+1]$ exactly once, and once removed, it never reappears. During this update sequence, we maintain a set of *active points*: points in the plane that are not necessarily current vertices of $\mathcal{E}[i-1, j+1]$ but are still under consideration. Whenever we advance in σ, we add all new vertices of $\mathcal{E}[i-1, j+1]$ to the active set. By construction, no point is ever added twice.

Consider the moment after we increment the second index, i.e., the current pair is (i', j') after increasing j'. Denote by H the half-plane left of $[j'+1, j'+2]$. We remove from the active set each point x that does not lie in H. Since x was a vertex of some earlier $\mathcal{E}[i-1, j+1]$, where (i, j) precedes (i', j') in σ, it follows that x remains active exactly until $r_x = [j'+1, j'+2]$. It remains to show that these operations can be performed in $O(n \log n)$ total time.

We store the active points in the data structure of Brodal and Jacob [11], which supports insertions and deletions in $O(\log n)$ time and and extreme-point queries in $O(\log n)$ time. Each time we add a vertex to the active set, we perform an insertion. When j' is incremented, we let H be the half-plane left of $[j'+1, j'+2]$ and iteratively perform extreme-point queries in the direction of the outward normal of H. If a query returns a vertex x outside H, then we set $r_x = [j'+1, j'+2]$ and remove x from the active set, charging $O(\log n)$ time for both the query and the update. The first time a query returns a vertex inside H, all remaining active points lie in H, and the process terminates.

By Lemma 9 and Lemma 10, at most $O(n)$ points are ever added to the active set, and each point is removed at most once. As each insertion, deletion, and query takes $O(\log n)$ time, the total running time is $O(n \log n)$. ☐

Corollary 20. *For every vertex $x \in V_\sigma$, we can compute the maximal chain $[\ell_x, r_x]$ of polygon edges such that the two defining edges of x are in $[\ell_x, r_x]$, and x is left of every supporting line of every edge in $[\ell_x, r_x]$, in total time $O(n \log n)$.*

Proof. This is an immediate consequence of Lemma 19 as we can iterate over σ once forward, and once backward. ☐

Theorem 21. *For a simple polygon P of n vertices, and conforming sliding sequence σ, we can compute a linear-size superset $\mathcal{B}_\sigma$ of size $O(n)$ of guards that contains the set of reduced bad dominators $\mathcal{B}$ using $O(n)$ space and $O(n \log n)$ time. Furthermore, we can construct a data structure in $O(n \log n)$ time that can answer the following query in $O(\log n)$ time:*

- *Given $x \in \partial P$ and $(i, j) \in \sigma$ such that $x \in [i-1, i)$ return the $(g, [u, v]) \in \mathcal{B}_\sigma(i, j)$ that maximizes $v \in [j, j+1]$.*

Proof. Lemma 9 implies that $|V_\sigma| \in O(n)$, so $|\mathcal{B}_\sigma| \in O(n)$. We apply Corollary 20 and compute for each $x \in V_\sigma$ the chain $[\ell_x, r_x]$. We then apply for each $x \in V_\sigma$ Lemma 13, which yields $\mathcal{B}_\sigma$ in $O(n \log n)$ time. We next prove that $\mathcal{B}_\sigma$ contains the set of reduced bad dominators. Suppose for contradiction that there is a reduced bad dominator $(c, [u_{\max}, v_{\max}])$ that is not in $\mathcal{B}_\sigma$. Then c is a vertex of some $\mathcal{E}[i-1, j+1]$, defined by two edges $[a-1, a]$ and $[b, b+1]$, and $u_{\max} \in [i-1, i)$ and $v_{\max} \in (j, j+1]$. If $(i, j) \in \sigma$, then $(c, [u_{\max}, v_{\max}])$ would be a candidate guard and thus in $\mathcal{B}_\sigma$. So, $(i, j) \notin \sigma$. By definition of σ, it must be that $\texttt{next}(u_{\max}) > j+1$. However, by definition of the reduced bad dominators (Definition 6) there is no guard dominating $(c, [u_{\max}, v_{\max}])$, contradicting that $\texttt{next}(u_{\max}) > j+1$. We conclude all reduced dominators are in $\mathcal{B}_\sigma$.

To construct the corresponding querying data structure, we loop over all $(i, j) \in \sigma$ in order, which discretizes time $t \in [0, |\sigma|]$ where at time t our loop is at $(i_t, j_t) \in \sigma$. For each $g \in V_\sigma$ there is a unique time interval $[t_1, t_2]$ where for all $t \in [t_1, t_2]$, g is a vertex of $\mathcal{E}[i_t - 1, j_t + 1]$. We loop over all t and use the first-in-first-out convex hull data structure from Corollary 11 to maintain $\mathcal{E}[i_t - 1, j_t + 1]$ explicitly in $O(n \log n)$ time and $O(n)$ space. This computes for all $g \in V_\sigma$ their time interval $[t_1, t_2]$. Let $g \in V_\sigma$, $(g, [u_{\max}, v_{\max}]) \in \mathcal{B}_\sigma$ and $[t_1, t_2]$ be the corresponding interval. We create a weighted rectangle $R_g := [t_1, t_2] \times [u_{\max}, v_{\max}]$ in $\mathbb{R}^2$ where the weight is $v_{\max}$. This creates a set R of $O(n)$ weighted rectangles in the plane, which we store in a stabbing-query data structure that for any query point q, returns the maximum-weighted rectangle in R that intersects q in logarithmic time. Such a stabbing query data structure of size $O(n)$ can be implemented in various ways through standard techniques (for details, we refer to the stabbing query implementations in [3]). Given a query point $x \in \partial P$ with $(i, j) \in \sigma$, we compute in $O(\log n)$ the corresponding time t such that $(i, j) = (i_t, j_t)$. We then perform a stabbing query with the point (t, x) and the maximum-weight rectangle corresponds to the desired query output. ☐

7 An $O(kn \log n)$-time algorithm

We will now present an algorithm, which for a set $X \subset \partial P$ will compute $\texttt{next}(x)$ for every $x \in X$ in total time $O((n + |X|) \log n)$. This then yields a $O(kn \log n)$ algorithm by applying this subroutine k times, together with Theorem 7. Figures 13 and 14 show how the $\texttt{next}$ function is recursively applied to different starting points in the same polygon.

Lemma 22. *Let P be a polygon with n vertices. Let $X \subset \partial P$ be given with $|X| \in O(n)$. There is an algorithm which computes a guard $(g, [x, \texttt{next}(x)])$ for every $x \in X$ in total time $O((n + |X|) \log n)$ using $O(n)$ space.*

Proof. As a preprocessing step, we compute in $O(n \log n)$ time via Theorems 16, 18 and 21:

- a conforming sliding sequence σ,
- a linear-sized set $\mathcal{D}'$ of $O(n)$ guards that contains the set $\mathcal{D}$ of reduced good dominators,
- a linear-sized set $\mathcal{B}_\sigma$ of $O(n)$ guards that contains the set $\mathcal{B}$ of reduced bad dominators,
- a segment tree T that stores, for all $(g, [u, v]) \in \mathcal{D}' \cup \mathcal{B}_\sigma$, the interval $[u, v]$ (together with g).

We first sort the points of X along ∂P in $O(|X| \log |X|)$ time and insert them into a queue. We then traverse all pairs $(i, j) \in \sigma$ by iteratively incrementing either i or j. We maintain the visibility core $\mathcal{E}[i-1, j+1]$ using the Sliding-Window data structure (3).

By definition of a conforming sliding sequence σ, any $x \in X$ with $x \in [i-1, i)$ and $\texttt{next}(x) \in (j, j+1]$ satisfies $(i, j) \in \sigma$. We maintain the invariant that for the current pair $(i, j) \in \sigma$, the head of the queue x satisfies $i \leq x$ and $j \leq \texttt{next}(x)$. Since next is monotone, this invariant for the head of the queue implies the same property for all remaining elements as well. Whenever $\texttt{next}(x)$ has been computed, we dequeue x.

Let x denote the current head of the queue. For any pair $(i', j') \in \sigma$ for which $x \in [i', i'+1)$, there exists a guard $(g, [x, j'])$ by definition of σ. If $x \notin [i-1, i)$, we continue traversing σ (incrementing either i or j). We are guaranteed that during this traversal, $j \leq \texttt{next}(x)$. Once (i, j) is such that $x \in [i-1, i)$, we have $\texttt{next}(x) \in (j, j+1]$ if and only if there exists no contiguous guard $(g, [x, j+1])$ with $g \in \mathcal{E}[i-1, j+2]$. We therefore apply Lemma 17 using the lookahead queries of the Sliding-Window data structure (3) to test in $O(\log n)$ time whether a guard $(g, [x, j+1])$ with $g \in \mathcal{E}[i-1, j+2]$ exists, and, if so, we increment j. This way, we find the pair (i, j) that corresponds to $x, \texttt{next}(x)$.

Given a pair (i, j) such that $x \in [i-1, i)$ and $\texttt{next}(x) \in (j, j+1]$, we compute the corresponding guard $(g, [x, \texttt{next}(x)])$ as follows. We first perform a stabbing query on T to find a maximum value v for which there exists a guard $(g, [u, v]) \in \mathcal{D}' \cup \mathcal{B}_\sigma$ such that $x \in [u, v]$. This stabbing query also returns $(g, [u, v])$. Every good guard is dominated by a reduced good dominator, and every bad guard is dominated by either a reduced bad dominator or an ugly dominator. Hence, $v \neq \texttt{next}(x)$ if and only if the only guards that can see $[x, \texttt{next}(x)]$ are ugly dominators.

We compute the shortest path from x to $j+1$ in $O(\log n)$ time and obtain the supporting line ℓ of its first edge. A ray-shooting query in Sliding-Window data structure (3) identifies the last point of intersection g^* between ℓ and $\mathcal{E}[i-1, j+1]$. By definition, the only ugly dominator that sees $[x, \texttt{next}(x)]$ has g^* as its guard. We apply Lemma 13 to compute the maximal chain $[u_{\max}, v_{\max}]$ that contains j and is visible from g^* in $O(\log n)$ time.

It follows that $\texttt{next}(x)$ is realized either by $(g, [u, v])$ or by $(g^*, [u_{\max}, v_{\max}])$, and we can distinguish between these cases in constant time. We dequeue x and proceed to the next element in the queue. Each $O(\log n)$ step is charged either to an increment of j or to the removal of one element from the queue. Hence, the total running time is $O((n + |X|) \log n)$. ◻

Theorem 23. *Let P be an instance of the Contiguous Art Gallery problem where k denotes the size of the optimal solution. We can compute a set of k guards that guard ∂P using linear space and $O(kn \log n)$ time.*

Proof. Let X be the set that contains all vertices of P and includes for all guards $(g, [u, v]) \in \mathcal{D}'$ the point u. Theorem 16 implies X can be computed in $O(n \log n)$ time. By Theorem 7 and the proof of Theorem 8, it follows that if we apply $\texttt{next}(x)$ to each $x \in X$, recursively, k times then for at least one $x \in X$ the result is a point $v > x + n$. In other words, for this $x \in X$ the recursive application of $\texttt{next}(x)$ results in a solution of size k that guards ∂P.

We first compute k and such a corresponding $x \in X$, and afterwards find the corresponding optimal guarding solution. Via Lemma 22, we can compute $\texttt{next}(x)$ for every x in X. We do this repeatedly, computing, and maintaining, $X_i = \{\texttt{next}(x) | x \in X_{i-1}\}$, with $X_0 = X$, until there is a point x such that $\texttt{next}^k(x) \geq x + n$. This takes k rounds, and thus a total of $O(kn \log n)$ time, and returns the size k of the optimal solution and some x. To also output the corresponding set of guards, we apply Lemma 22 once more with an input $X = \{x\}$ where each time we find a guard $(g, [x, \texttt{next}(x)])$, we quickly add $\texttt{next}(x)$ to X and the queue before we dequeue x. Since Lemma 22 computes the explicit guard $(g, [x, \texttt{next}(x)])$, the result are k guards that together see all of ∂P. ☐

For the remainder of this paper we will assume that we know that there is no solution of size 3 or less. This allows us to impose stronger structure, which in turn enables us to give a $O(n \log n)$ algorithm for a polygon where only solutions of size at least 4 exist. To this end, we show that Theorem 23 implies an $O(n \log n)$-time test for this property:

Corollary 24. *Let P be a polygon consisting of n vertices and K be some integer. We can test in $O(Kn \log n)$ time whether the* Contiguous Art Gallery *problem with P as its input has a solution of size K.*

Proof. We simply stop the algorithm described in Theorem 23 after K rounds. ☐

8 Computing the functions

In this section, we concern ourselves with computing a representation of $\texttt{next} : \partial P \to \partial P$, which we think of as a function from $[1, n+1)$ to $[1, 2n+1]$. In particular, this representation partitions $[1, n+1)$ into $O(n)$ contiguous pieces $I_1, \ldots$, where each I_i is endowed with four values A_i, B_i, C_i and D_i such that

$$\texttt{next}(u) = \begin{cases} \frac{A_1+B_1u}{C_1+D_1u} & \text{if } u \in I_1 \\ \frac{A_2+B_2u}{C_2+D_2u} & \text{if } u \in I_2 \\ & \vdots \end{cases}$$

For this, we use the sequence σ of Definition 1 to traverse ∂P, maintaining the visibility core in a Sliding-Window data structure (3). From here on, we fix the sequence σ as the conforming sliding sequence computed by Theorem 18. Notably, we only need to concern ourselves with u, where $\texttt{next}(u)$ is realized only by ugly dominators (). Central to our analysis of the construction of this function are multiple interwoven charging arguments. In particular, we will charge almost all queries to the data structure to vertices of the shortest path from u to $\texttt{next}(u)$, and vertices of the visibility core itself.

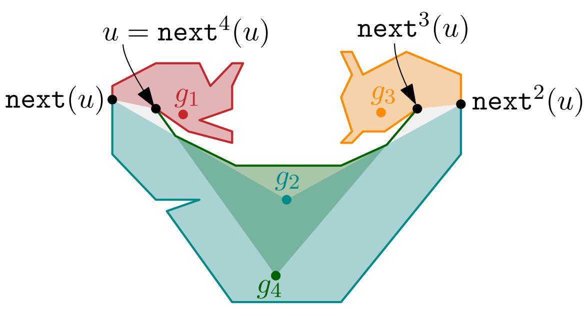

Figure 13: Illustration of optimal solution of size 4, induced by $u \in \partial P$ that is a vertex of P. The guard g_i is the realizing guard/dominator for $[\texttt{next}^{i-1}(u), \texttt{next}^i(u)]$.

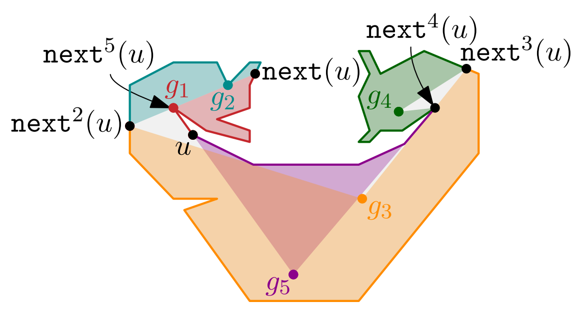

Figure 14: Illustration of non-optimal solution of size 5, induced by $u \in \partial P$ that is a vertex of P. The guard g_i is the realizing guard/dominator for $[\texttt{next}^{i-1}(u), \texttt{next}^i(u)]$.

8.1 Partitioning the function domain

Lemma 25. *Suppose $u \in (i-1, i)$ and $\texttt{next}(u) \in (j, j+1)$. Let $(h, [u_h, v_h])$ be the guard in $\mathcal{B}_\sigma(i,j)$ such that $u \in [u_h, v_h]$, and v_h is maximal. Let e be the edge of $\mathcal{E}[i-1, j+1]$ after h in counter-clockwise order. Then $\texttt{next}(u)$ is realized by an ugly dominator ($\searrow$), if and only if*

(i) $u > u_h$, and
(ii) $\sphericalangle(u_h, h, v_h) > \pi$, and
(iii) the edge e lies right of the first edge of the shortest path $S(h, j+1)$.

In this case, the realizing guard g of $\texttt{next}(u)$ is on edge e.

Proof. Suppose $\texttt{next}(u)$ is realized by an ugly dominator ($\searrow$) $(g, [u, \texttt{next}(u)])$. First, if $u = u_h$, then $u_h \in (i-1, i)$ and then by Lemma 3, $\overline{h\,u_h}$ has a reflex-vertex in its interior. But then h is the last intersection point of the supporting line of the first edge of the shortest path $S(u, j+1)$, and thus $g = h$, and in particular $v_h = \texttt{next}(u)$. Thus $\texttt{next}(u)$ is realized by a type ($\star$) dominator instead. Hence (i) holds.

Next, by definition, g is the second intersection point of the supporting line ℓ_u of the first edge of the shortest path $S(u, j+1)$ with $\mathcal{E}[i-1, j+1]$. As $\texttt{next}(u)$ is realized by an ugly dominator ($\searrow$) g is not a vertex of $\mathcal{E}[i-1, j+1]$. Let C be the counter-clockwise chain of the boundary of $\mathcal{E}[i-1, j+1]$ that is right of ℓ_u. Let h' be the last vertex in C. Let $[u', v']$ be the maximal visibility of h' in $[i-1, j+1]$. This is unique, as $h' \in \mathcal{E}[i-1, j+1]$ and $S(u, \texttt{next}(u))$ is left turning (by Lemma 3). As h' is right of ℓ_u, it can see u, so $u' \leq u$. Furthermore, by maximality of $\texttt{next}(u)$, and the fact that h' induces a bad dominator ($\star$) whose visibility contains $[u', v']$, it must be that g sees further than v'. It follows that the point g lies right of the first edge of the shortest path $S(h', j+1)$, which coincides with the supporting line of $\overline{h'\,v'}$. Let x' be the vertex of $S(h', j+1)$ defining this first edge. Any other vertex of C, and any other vertex of $\mathcal{E}[i-1, j+1]$ that sees u, is left of $\overline{h'\,x'}$ and thus has x' on its shortest path to $j+1$. Thus it cannot see v'. Further, the guard g must be on the edge after h' in counter-clockwise order. As g is below the supporting line of $\overline{h'\,v'}$, so is the entirety of the edge e. Thus h' sees its two defining edges and hence is in $\mathcal{B}_\sigma(i,j)$, and in particular, $h' = h$, and $[u', v'] = [u_h, v_h]$, and g is on e, and (iii) holds. By general position, x' is not colinear with e, and thus we have that both $u' = u_h$ and $v' = v_h$ lie strictly left of e, and thus (ii) holds.

Now, conversely, assume (i), (ii), and (iii) hold. Then (i) implies that u can see a point g' on the edge e. By (iii), this guard g' sees further than v_h. But then it must be dominated by an ugly dominator ($\searrow$). In particular, the next vertex (in counter-clockwise order) after h of $\mathcal{E}[i-1, j+1]$ can see the furthest among all guards on the edge e, and in particular further than h. Thus, u does not see this next vertex, and the supporting line of the first edge of the shortest path $S(u, j+1)$ intersects the edge e, defining the the ugly dominator ($\searrow$) g of u, concluding the proof. $\square$

The consequence of the above lemma is formalized in the following lemma, stating that we can compute $O(n)$ interior-disjoint intervals such that $u \in \partial P$ is in one of these intervals if and only if $\texttt{next}(u)$ is realized by an ugly dominator ($\searrow$). Each such interval is defined by an index i, such that $u \in (i-1, i)$, an index j, such that $\texttt{next}(u) \in (j, j+1)$, and an edge e of $\mathcal{E}[i-1, j+1]$, such that the realizing guard associated to it will be on edge e. We call the set that has these three edges associated to it $I_{i,j,e}$. Let $\mathcal{T}$ be the set of all such triples (i, j, e) such that $I_{i,j,e} \neq \emptyset$.

Lemma 26. *The sets $\{I_{i,j,e} | (i,j,e) \in \mathcal{T}\}$ can be computed in $O(n \log n)$ time given the set $\mathcal{B}_\sigma$.*

Proof. Let us first consider the partition of ∂P into sets $C_{i,j} = \{u \in \partial P | u \in [i-1, i), \texttt{next}(u) \in (j, j+1]\}$. Each $C_{i,j}$ is contiguous, and there are at most $O(n)$ non-empty such $C_{i,j}$. These can also be computed by Lemma 9 and Lemma 10 in $O(n \log n)$ time. Next, consider the refinement of the set $C_{i,j}$ via the arrangement of $\{[u_h, v_h] | (h, [u_h, v_h]) \in \mathcal{B}_\sigma(i,j)\}$ into cells $C_{i,j,h}$. By Lemma 10, each guard in $\mathcal{B}_\sigma$ is in a contiguous subsequence of $\{\mathcal{B}_\sigma(i,j) | (i,j) \in \sigma\}$, there are again only $O(n)$ non-empty sets among all such $C_{i,j,h}$, and they can be computed in $O(n \log n)$ time via $\mathcal{B}_\sigma$ and $C_{i,j}$. Finally, associate every $C_{i,j,h}$ with the guard $(h^*, [u_h, v_h]) \in \mathcal{B}_\sigma(i,j)$ (or more precisely, the edge e of $\mathcal{E}[i-1, j+1]$ that is after h^* in counter-clockwise order) that maximizes v_h subject to $C_{i,j,h} \subset [u_h, v_h]$. This can be done via Theorem 21. Now consider only those $C_{i,j,h}$ such that its associated guard and edge fulfill the conditions of Lemma 25. The set of cells among them associated to the same guard $(h^*, [u_h, v_h])$ is contiguous, as $i \in [u_h, v_h]$, and hence any later $C_{i,j',h'}$ is also contained in $[u_h, v_h]$. The union of these sets associated to h^* and with it, associated to e define the set $I_{i,j,e}$. By construction, there are at most $O(n)$ many of these which are non-empty, and these can be computed in $O(n \log n)$ time from $C_{i,j,h}$ via a simple linear scan. ☐

Definition 11. For $u \in I_{i,j,e}$ we define x_u to be the vertex of $S(i-1, j+1)$ that defines the first edge of the shortest path $S(u, j+1)$. We call x_u the left pivot of u. We similarly define the right pivot y_u to be the vertex of $S(i-1, j+1)$ defining the first edge of the shortest path $S(g, j+1)$, where g is the guard realizing $\texttt{next}(u)$. With this, we define the sets

- $L_{i,j,e,x} = \{u \in I_{i,j,e} | x = x_u\}$, and
- $R_{i,j,e,y} = \{u \in I_{i,j,e} | y = y_u\}$.

The collection of sets $L_{i,j,e,x}$ (and $R_{i,j,e,y}$) for all x (and y) partition $I_{i,j,e}$. Their intersections $C_{i,j,e,x,y} = L_{i,j,e,x} \cap R_{i,j,e,y}$ together define another partition of $I_{i,j,e}$.

Observation 27. $L_{i,j,e,x}$ and $R_{i,j,e,y}$ are contiguous subsets of $I_{i,j,e}$, and by definition interior-disjoint.

Lemma 28. *For every $I_{i,j,e}$ we have*

$$|\{x, y | C_{i,j,e,x,y} \neq \emptyset\}| = O(|\{x | L_{i,j,e,x} \neq \emptyset\}| + |\{y | R_{i,j,e,y \neq \emptyset}\}|).$$

Further, the set of all non-empty $C_{i,j,e,x,y}$ on $I_{i,j,e}$ can be computed in $O(|\{x | L_{i,j,e,x} \neq \emptyset\}| + |\{y | R_{i,j,e,y \neq \emptyset}\}| + \log n)$ time.

Proof. Let $I_{i,j,e} = [s,t] \subset [i-1, i]$. Let g_s be the guard on edge e realizing $\texttt{next}(s)$, and let g_t be the guard on edge e realizing $\texttt{next}(t)$. For every $u \in [s,t]$, the realizing guard g_u is in $\overline{g_s\, g_t}$. The left pivot of u is the first vertex of the shortest path $S(g_u, u)$ and in particular, by the convexity of $S(i-1, j+1)$ on the shortest path $S(g_t, s)$. Conversely, for every vertex x in the shortest path $S(g_t, s)$ there is a u such that $\ell_u = x$. Similarly, the set of right pivots for $u \in [s,t]$ is precisely the set of vertices in the shortest path $S(g_s, \texttt{next}(t))$. These can be computed in time $O(\log n + |\{x | L_{i,j,e,x} \neq \emptyset\}| + |\{y | R_{i,j,e,y \neq \emptyset}\}|)$ using Data Structure 1. The non-empty sets $L_{i,j,e,x}$ and $R_{i,j,e,y}$ are contiguous, and hence the non-empty sets $C_{i,j,e,x,y}$ can be computed by a simple linear scan in time $O(|\{x | L_{i,j,e,x} \neq \emptyset\}| + |\{y | R_{i,j,e,y \neq \emptyset}\}|)$ concluding the proof. ☐

Lemma 29. *Let $C_{i,j,e,x,y}$ be given. Then in $O(1)$ time one can compute values A, B, C, and D such that*

$$\forall u \in C_{i,j,e,x,y} : \texttt{next}(u) = \frac{A + Bu}{C + Du}.$$

Proof. For $u \in C_{i,j,e,x,y}$, $\texttt{next}(u)$ is realized by an ugly dominator (). That is, its realizing guard g is the intersection of the supporting line of $\overline{u\, x}$ with (the supporting line of) e. Similarly, given g, $\texttt{next}(u)$ is the intersection of the supporting line of $\overline{g\, y}$ with (the supporting line of) $[j, j+1]$. Both of these functions are Möbius transformations, i.e., functions of the form $\frac{A+Bu}{C+Du}$, where A, B, C, and D depend on the coordinates of the start- and end points of $[i, i+1]$, e, $[j, j+1]$, x and y. As Möbius transformations are closed under concatenation, their concatenation is of the form $\frac{A+Bu}{C+Du}$ as well, concluding the proof. ☐

Thus, if in total only $O(n)$ of the sets $L_{i,j,e,x}$ and $R_{i,j,e,y}$ are non-empty, then we can compute a representation of the $\texttt{next}$ function of $O(n)$ pieces. To this end, consider the set $\hat{\mathcal{U}}$ of points $u \in \partial P$ such that $\texttt{next}(u)$ is realized by only bad guards. The realizing guard $(g, [u, \texttt{next}(u)])$ has that $\sphericalangle(u, g, \texttt{next}(u)) > \pi$, and that $S(u, \texttt{next}(u))$ is left-turning. Consider the subset $\mathcal{U} \subset \hat{\mathcal{U}}$ for which the shortest path $S(u, \texttt{next}(u))$ has at least two inner vertices. By Lemma 25 and Lemma 3, $u \in \partial P$ such that neither u nor $\texttt{next}(u)$ is a vertex of P and $\texttt{next}(u)$ is realized by an ugly dominator (➴) in $\mathcal{U}$. For a vertex x of P, let $L_x \subset \mathcal{U}$ be the set of points $u \in U$ such that x is the first inner vertex of the shortest path $\pi_u = S(u, \texttt{next}(u))$. Let $\#L_x$ be the number of connected components of L_x as a subset of $\mathcal{U}$. In the next section, we show that

$$\sum_{x \text{ vertex of } P} \#L_x \in O(n).$$

This in turn implies that

$$\sum_{(i,j,e) \in \mathcal{T}} |\{x | L_{i,j,e,x} \neq \emptyset\}| \leq \left(\sum_{(i,j,e) \in \mathcal{T}} 2 \right) + \left(\sum_{x \text{ vertex of } P} \#L_x \right) \in O(n).$$

A symmetric argument shows $\sum_{(i,j,e) \in \mathcal{T}} |\{y | R_{i,j,e,y} \neq \emptyset\}| \in O(n)$.

8.2 Bounding the number of pivot events

We begin by showing a stronger version of Theorem 7. For this, we first define a relation on the vertices of the polygon, where $x \prec y$, if x appears before y in $\pi_u = S(u, \texttt{next}(u))$ for some $u \in \mathcal{U}$, which will induce a strict order.

Lemma 30. *Let P be given. If no three contiguous guards can guard all of ∂P, then the transitive closure of the relation $\prec$ is a partial order.*

Proof. It suffices to show that if there is a cycle in $\prec$, i.e., there are $u_1, \ldots, u_M \in \mathcal{U}$ such that for $i < M$ the path $\pi_i = S(u_i, \texttt{next}(u_i))$ has the vertex x_i before x_{i+1} and π_M has the vertex x_M before x_1, then a guarding set of size 3 exists. Hence, let us assume such a cycle exists. Consider the prefix $x_1, \ldots, x_m, x_{m+1}$ where x_{m+1} is the first x_i that is more than one full revolution ahead of x_1, that is, $x_1 \in [x_m, x_{m+1}]$. Then $x_1, \ldots, x_m$ are sorted clockwise along the boundary of P, as all shortest paths π_i are left-turning. We may assume that $x_{m+1} \in [x_1, x_2]$, by removing a prefix from $x_1, \ldots, x_{m+1}$.

Let $\overline{\pi_i}$ the subpath of π_i from x_i to x_{i+1}. Observe that each $\overline{\pi_i}$ is left-turning by definition of $\mathcal{U}$. Let P_i be the portion of ∂P between x_i and x_{i+1}, i.e. $P_i = [x_i, x_{i+1}]$. The concatenation of all $\overline{\pi_1}, \ldots, \overline{\pi_m}$ forms a closed loop L in P, after short-cutting $\overline{\pi_1}$ and $\overline{\pi_m}$ via their intersection point (they must intersect, as $x_m \leq x_1 \leq x_{m+1} < x_2$).

The loop L is simple and clockwise. First, observe that $\overline{\pi_i}$ and $\overline{\pi_{i+1}}$ intersect in x_{i+1} (or the intersection point between x_1 and x_m), and as $\overline{\pi_i}$ and $\overline{\pi_{i+1}}$ are both left-turning shortest paths they cannot intersect in any other point. It remains to show that $\overline{\pi_i}$ and $\overline{\pi_j}$ do not intersect if $j \notin \{i-1, i, i+1\}$. Assume otherwise. Then either x_j or x_{j+1} lies in P_i contradicting the clockwise order of x_i, x_{i+1}, x_j and x_{j+1}. Thus, L is a simple loop. Furthermore, as each π_i is left-turning and contained in P, the loop L must be clockwise.

At most three guards guard all of ∂P. Let $x \in L$ be a point that is in the interior of some $\overline{\pi_i}$. A half-plane h defined by a line ℓ is said to be a tangent at x, if $x \in \ell$, and h does not contain $\overline{\pi_i}$ in its interior. At the intersection point x' between $\overline{\pi_i}$ and $\overline{\pi_{i+1}}$ we may choose (and need to specify) the membership of x' to be either $\overline{\pi_i}$ or $\overline{\pi_{i+1}}$. In other words, a half-plane h defined by a line ℓ is a tangent at x', if $x' \in \ell$, and h does not contain one of $\overline{\pi_i}$ and $\overline{\pi_{i+1}}$.

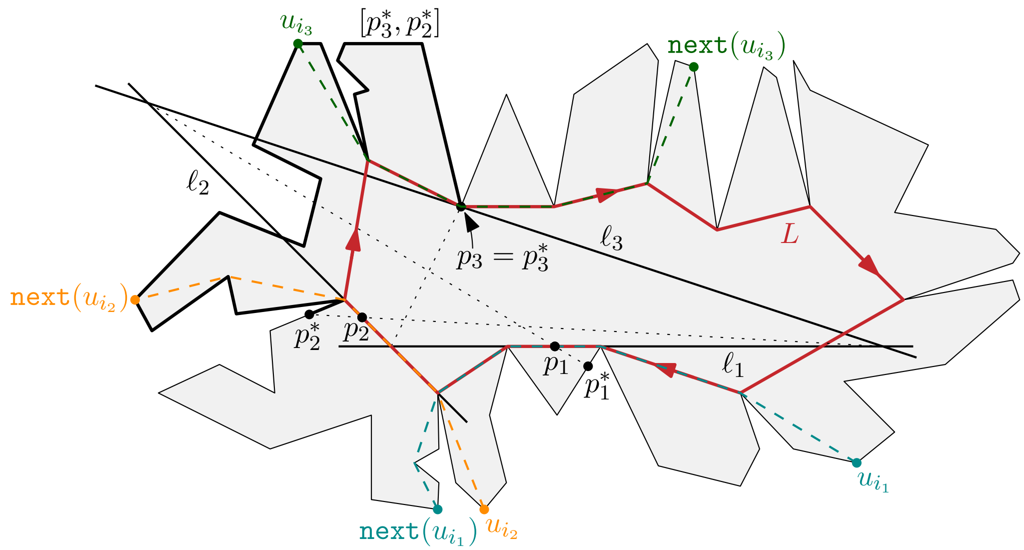

Figure 15: Construction of three guards guarding all of ∂P from the proof of Lemma 30. Illustrated is the fact that $[u_{i_1}, \texttt{next}(u_{i_1})]$ contains $[p_3^*, p_2^*]$, which implies that g_{i_1} sees $[p_3^*, p_2^*]$.

Let us for now suppose, there are three points p_1, p_2, and p_3 on L, with tangents h_1, h_2 and h_3 defined by ℓ_1, ℓ_2, and ℓ_3, such that (i) each h_i contains all three points p_1, p_2, and p_3, (ii) $h_1 \cap h_2 \cap h_3$ is bounded, and (iii) for the three π_{i_1}, π_{i_2} and π_{i_3} containing p_1, p_2 and p_3, no two are equal. Then there are three guards guarding all of ∂P (refer to Figure 15): As $h_1 \cap h_2 \cap h_3$ is bounded, we can pair p_1 with a point $p_1^* \in \partial P$ that is in $h_2 \cap h_3$, if p_1 is not already on ∂P, by shooting a ray in P starting in p_1 away from the intersection point $\ell_2 \cap \ell_3$. As h_1 contains the point $\ell_2 \cap \ell_3$, and the ray cannot intersect π_{i_1} a second time before hitting the polygon boundary, we have that $p_1^* \in P_{i_1}$, unless π_1 was already in P_{i_1}, in which case we define $p_1^* = p_1$. Similarly, we find for p_2 and p_3 the points p_2^* and p_3^*. The points p_1^*, p_2^*, and p_3^* appear along ∂P in clockwise order. Finally, note that p_2^* and p_3^* are in h_1, and thus do not lie in $[u_{i_1}, \texttt{next}(u_{i_1})]$, which implies that $[p_3^*, p_2^*] \subset [u_{i_1}, \texttt{next}(u_{i_1})]$.[2] In particular, there is a contiguous guard $(g_1, [p_3^*, p_2^*])$. Similarly, there are guards $(g_2, [p_1^*, p_3^*])$ and $(g_3, [p_2^*, p_1^*])$, which concludes the proof.

The points p_1, p_2, and p_3 such that (i), (ii), and (iii) hold, exist. Observe that indeed any number of such points already induce a set of three such points. The only condition that might fail, by removing such points is (ii), but as the intersection of tangents together form a convex, bounded set, there must also be a subset of just three such tangents whose intersection is bounded.

We interpret L as a simple polygon oriented clockwise. We begin, by picking p_1 and h_1 arbitrarily such that p_1 is not the intersection point of two paths π_i and π_{i+1}. Let p_2 result from p_1 by shooting a ray from p_1 orthogonal to ℓ_1. Pick h_2 arbitrarily, such that p_1 is not on h_2. Without loss of generality, ℓ_1 and ℓ_2 are not parallel, as otherwise $[\texttt{next}(u_{i_1}), u_{i_1}]$ and $[\texttt{next}(u_{i_2}), u_{i_2}]$ are disjoint, and there is even a set of guards guarding all of ∂P of size 2. Assume without loss of generality that ℓ_1 and ℓ_2 intersect right of the line segment $\overline{p_1\, p_2}$. As L is a simple loop, there is a chain $[s, t] \subset L$ such that s is on ℓ_1, t is on ℓ_2, and all of $[s, t]$ is in $h_1 \cap h_2$, see Figure 16. If there is a point in $[s, t]$ with a tangent parallel to $\overline{p_1\, p_2}$, then we can choose p_3 as this point, as all of $[s, t]$ is left of $\overline{p_1\, p_2}$. Now assume that there is no point in $[s, t]$ with a tangent parallel to $\overline{p_1\, p_2}$. Consider the function f which maps $x \in [s, t]$ to the distance from its orthogonal projection on ℓ_1 to p_1. The function f can only have one local minimum in $[s, t]$, as otherwise there is a point between two such local

[2]Recall that intervals between points on ∂P are defined in counter-clockwise order.

minima with tangent parallel to $\overline{p_1 p_2}$. If f has its local minimum at either s or t, the chain $[s,t]$ is left-turning and there must be a point p_3 in $[s,t]$ whose tangent includes p_1 and p_2, as $[s,t]$ does not intersect $\overline{p_1 p_2}$. Let instead $m \in (s,t)$ be the point where this minimum is attained.

Now root a halfspace h_s with supporting line ℓ_s at s that is parallel to ℓ_2 and contains t, and similarly root a halfspace h_t with supporting ℓ_t at t that is parallel to ℓ_1 and contains s. If $[s,m]$ enters the interior of h_s, then the first such point x has a tangent which intersects both ℓ_1 and ℓ_2 left of $\overline{p_1 p_2}$, defining the desired point p_3. Symmetrically, we find a point p_3 if $[m,t]$ intersects the interior of h_t. So, assume that no point of $[s,t]$ enters the interior of $h_s \cap h_t$.The point m is an intersection point of two π_{i_m} and π_{i_m+1}, as L is right-turning at m.

Let z be the intersection point of ℓ_s and ℓ_t. Again refer to Figure 16 for the following construction. The triangle defined by s, t and z does not contain any point of $[s,t]$ in its interior. We transform this triangle by moving z towards m until it hits either $[s,m]$ or $[m,t]$. Without loss of generality, let it hit $[m,t]$. We then continue moving z, but away from t, instead of towards m. The result is a triangle that touches both $(s,m]$ and $[m,t)$, and still does not contain any point of $[s,t]$ in its interior. Let p_s and p_t be the two points where the triangle touches $(s,m]$ and $[m,t)$. If p_s is m, then we choose the tangent h'_s defined by ℓ'_s at p_s according to π_{i_m}. If p_t is m, then we choose the tangent h'_t defined by ℓ'_t at p_t according to π_{i_m+1}. Otherwise, we chose the tangent such that it is colinear with the face of the triangle it is touching. This way, we obtain two tangents that intersect, and their intersection contains the triangle and, in particular, s, t, p_1, and p_2. Further, the line ℓ'_s must intersect ℓ_1 left of p_1, as otherwise there is a point in (s,p_s) that is another local minimum of f, contradicting the fact that m is the only local minimum of f. It follows that, p_1, p_2, p_3, and p_4 fulfill properties (i), (ii), and (iii), and in particular, a subset of at most three of them as well, concluding the proof. ☐

A consequence of the proof of Lemma 30 is, that *any* minimal solution of size at least 4 contains at least one good guard, strengthening Theorem 7. We do not make use of this fact. Instead, the consequence we are after is the following.

Corollary 31. *Let P be a simple polygon where no three contiguous guards can guard all of ∂P. Then there is strict order $\prec^*$ of the vertices of the polygon such that for any $u \in \mathcal{U}$, the first inner vertex of the shortest path $S(u, \mathtt{next}(u))$ is the minimum w.r.t. the order $\prec^*$, of all inner vertices in $S(u, \mathtt{next}(u))$.*

Proof. This is an immediate consequence of Lemma 30. ☐

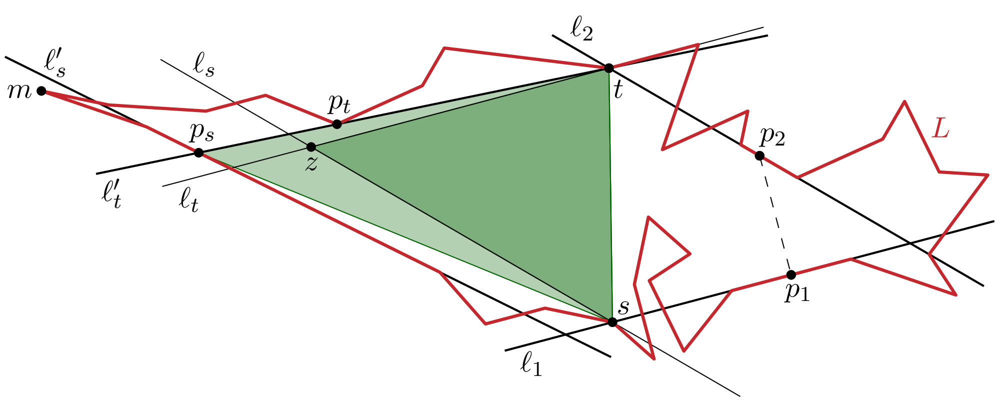

Figure 16: Construction of the points p_s and p_t by transforming the green triangle defined by s, t, and z into the light green triangle.

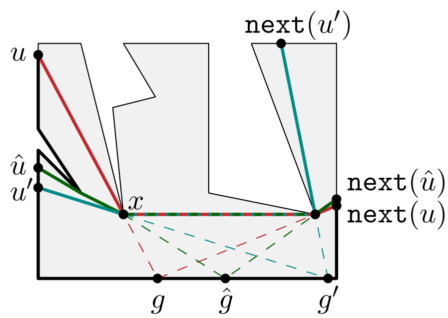

Figure 17: The set L_x is not necessarily connected, but for any two points u and u' in L_x, the vertex x stays on the shortest path for any $\hat{u}$ in between u and u', i.e., $\hat{u} \in I_x$.

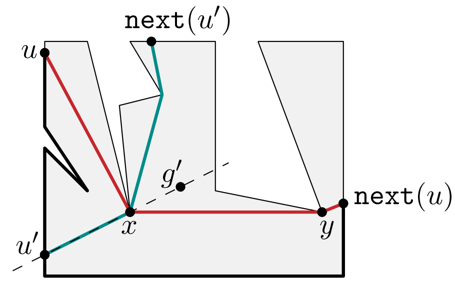

Figure 18: For any point $u' \in L_x$ strictly right of $\overline{x\,y}$, the potential ugly dominator () g' cannot see $\texttt{next}(u)$.

Let us now consider for any vertex x of P the subset $I_x \subset \mathcal{U}$ of points $u \in \mathcal{U}$ such that x is on the shortest path $S(u, \texttt{next}(u))$. By Corollary 31, $u \in L_x$ if and only if $x = \min_{\prec^*}\{y | y \in I_u\}$. Figure 17 illustrates that the set L_x is not necessarily connected, but there is some contiguity for the set I_x, which we prove in the next lemma.

Lemma 32. *Let P be a simple polygon where no three contiguous guards can guard all of ∂P. Let $u, u' \in L_x$ for a vertex x of P, with $u < u'$. Then at least one of $[u, u']$ and $[u', u+n]$ is contained in I_x.*

Proof. Recall that $u \in L_x$ means that $u \in \mathcal{U}$, so the shortest path $\pi_u = S(u, \texttt{next}(u))$ is left-turning and has at least two inner vertices, and x is the first inner vertex of π_u. Observe that $[u, \texttt{next}(u)]$ and $[u', \texttt{next}(u')]$ intersect, as otherwise the two left-turning shortest paths $\pi_u = S(u, \texttt{next}(u))$ and $\pi_{u'} = S(u', \texttt{next}(u'))$ cannot share a vertex. Assume that $u' \in [u, \texttt{next}(u)]$, and thus $\texttt{next}(u) \in [u', \texttt{next}(u')]$, Otherwise, we can reverse the roles of u and u' via $u \leftarrow u + n$. Refer to Figure 18 for an illustration of the proof. We first show that u' must be left of the second edge of π_u, i.e. the edge after x. Suppose for contradiction that u' is strictly right of the second edge of π_u. Let y be the vertex after x of π_u. Note that y exists as π_u has at least two inner vertices. Then $y \notin \pi_{u'}$, as $y \in \pi_{u'}$ would contradict that $\pi_{u'}$ is left-turning and contains x. It follows that $\texttt{next}(u') \in [y, x]$. Because the guard $(g', [u', \texttt{next}(u')])$ is an ugly dominator (), g' is the intersection point of the supporting lines of the first and last edges of $\pi_{u'}$, and thus on the supporting line of the edge $\overline{u'\,x}$, and strictly after x. Hence, g' is in the polygon bounded by $\overline{x\,y}$ and $[y, x]$. However, then g' does not see $\texttt{next}(u)$, contradicting that it sees all of $[u', \texttt{next}(u')] \ni \texttt{next}(u)$.

We conclude that u' is left of the second edge of π_u. This implies that both shortest paths $S(u, \texttt{next}(u'))$ and $S(u', \texttt{next}(u))$ are also left-turning shortest paths that contain x. As for any point $\hat{u} \in [u, u']$ we have that $\texttt{next}(\hat{u}) \in [\texttt{next}(u), \texttt{next}(u')]$, the shortest path $S(\hat{u}, \texttt{next}(\hat{u}))$ cannot properly intersect $S(u, \texttt{next}(u'))$ and $S(u', \texttt{next}(u))$. Thus, $S(\hat{u}, \texttt{next}(\hat{u}))$ must be left-turning with at least two inner vertices and also have $x \in S(\hat{u}, \texttt{next}(\hat{u}))$. ☐

Corollary 33. *Let P be a simple polygon where no three contiguous guards can guard all of ∂P. Then for every vertex x of P there is a contiguous subset $I_x^* \subset I_x \subset \mathcal{U}$ such that $L_x \subset I_x^*$. In particular,*

$$u \in L_x \iff x = \min_{\prec^*}\{y | u \in I_y^*\}.$$

Proof. This is an immediate consequence of Corollary 31 and Lemma 32. ☐

Lemma 34. *Let P be a simple polygon where no three contiguous guards can guard all of ∂P. Then*

$$\sum_{x \text{ vertex of } P} \#L_x \in O(n).$$

Proof. Consider for every vertex x of P the contiguous subset $I_x^* \subset I_x \subset \mathcal{U}$ as defined in Corollary 33. These sets define $O(n)$ interior disjoint contiguous intervals in $\mathcal{U}$, where in every interval all points in the interval are contained in the same sets I_x^*. In particular, $u \mapsto \min_{\prec^*}\{y|u \in I_y^*\}$ is constant for each interval. Hence for $\#L_x$, which is the number of connected components of L_x as a subset of $\mathcal{U}$, we have:

$$\sum_{x \text{ vertex of } P} \#L_x \in O(n). \qquad \square$$

Lemma 35. *Let P be a simple polygon where no three contiguous guards can guard all of ∂P. Then*

$$\sum_{(i,j,e)\in\mathcal{T}} (|\{x|L_{i,j,e,x} \neq \emptyset\}| + |\{x|R_{i,j,e,x} \neq \emptyset\}|) \in O(n).$$

Proof. This is an immediate consequence of Lemma 34, together with the dual nature of L_x and R_x, and the fact that

$$\sum_{(i,j,e)\in\mathcal{T}} |\{x|L_{i,j,e,x} \neq \emptyset\}| \leq \left(\sum_{(i,j,e)\in\mathcal{T}} 2\right) + \left(\sum_{x \text{ vertex of } P} \#L_x\right) \in O(n). \qquad \square$$

Theorem 36. *Let P be a simple polygon consisting of n vertices. Suppose, P cannot be guarded with three or less guards. Then a representation of* $\texttt{next} : [1,n] \to [1,n]$ *consisting of $O(n)$ disjoint intervals, where on each disjoint interval I, we have that* $\texttt{next}(u) = \frac{A_I + B_I u}{C_I + D_I u}$, *can be computed in total time $O(n \log n)$.*

Proof. This is an immediate consequence of the computation of all $O(n)$ guards in $\mathcal{D}'$, $\mathcal{B}_\sigma$ in $O(n \log n)$ time together with Lemma 25, Lemma 28, Lemma 29, and Lemma 35. $\square$

9 An $O(n \log n)$-time algorithm

Theorem 37. *The* Contiguous Art Gallery *problem can be solved in $O(n \log n)$ time.*

Proof. We first check in $O(n \log n)$ time via Corollary 24, if there is a solution of size at most 3. Otherwise, via Theorem 36, we compute the representation of `next` as a piecewise Möbius transform, consisting of $O(n)$ pieces. Given this representation, there is a $O(n \log k)$ algorithm [4], also discussed in [38, Remark 10], which can compute the minimum k, together with a point $x \in \partial P$, such that $\texttt{next}^k(x)$ overtakes x, i.e., $[x, \texttt{next}^k(x)] = \partial P$, or conversely, $\texttt{next}^k(x) \geq x + n$. From this x, via the representation of `next` we can compute a solution of size k in time $O(k \log n)$, concluding the proof. $\square$

10 Lower bound

We complement our results in the `realRAM`-model with a tight lower bound. We reduce from SetDisjointness.

Problem Statement 2 (SetDisjointness). Given two lists $A, B \subset [1, n^3]$ of integers, each of size n, is $A \cap B = \emptyset$?

Theorem 38. *In any comparison-based model of computation, the* SetDisjointness *problem takes* $\Omega(n \log n)$ *comparisons, even if* A *and* B *are integer sets where one of them is sorted.*

Proof. Although this follows almost-immediately from the entropy argument in [45], we present an argument for completeness. Consider the special case where for the input instance all elements in the list $A \cup B$ are distinct. Any deterministic *correct* algorithm must produce a certificate that verifies that for all index pairs (i, j): $A[i] \neq B[j]$. Indeed, consider an algorithm $\mathcal{A}$ where for some input instance I, for some index pair (i, j), the algorithm does not verify whether $A[i] = B[j]$. Then an adversary may simply consider two instances, I and I' where I' replaces $B[j]$ by $A[i]$ – adding a duplicate. The algorithm $\mathcal{A}$ cannot distinguish between these two inputs, since any such distinction would produce a certificate that $A[i] \neq B[j]$ for the input I. However, this implies that the algorithm $\mathcal{A}$ must provide the wrong answer for one of these two inputs.

For ease of arithmetic, we assume that A has $n + 1$ elements starting at $A[0]$. For any permutation π of $\{1, \ldots, n\}$ we can construct an input I where for all indices $i \in [1, n-1]$, $A[\pi(i) - 1] < B[i] < A[\pi(i)]$. For any two permutations π_1, π_2, the resulting certificates must be distinct because for all $i \in [1, n-1]$, the fact that $B[i] \neq A[\pi(i)]$ and $B[i] \neq A[\pi(i) - 1]$ cannot be implied by transitivity over comparisons between any $B[j]$ and elements of A. It follows that there are $\Theta(n!)$ inputs that all yield different certificates which implies an $\Omega(n \log n)$ comparison-based lower bound. □

We now give a construction reducing SetDisjointness to computing the minimum number of guards for a given polygon. For this we construct three gadgets, that we place on the boundary of a big polygon. The big polygon is a right isosceles triangle, whose base has length n^3. For the following construction refer to Figure 19.

The first gadget, called the *blocker gadget*, is used to encode the sorted list A. We remove for each $a \in A$ a unit square centered at the boundary of the triangle at $(a, \min(a, n^3 - a))$. The primary effect these gadgets have, is that there can exist no guard $(g, [u, v])$ with g at the boundary of the big polygon and $g \in (a, \min(a, n^3 - a))$.

The second gadget, called the *laser gadget*, is used to encode the unsorted list B. At the midpoint of the base, in an interval of length 1, we will place laser gadgets evenly spread over the interval. For each $b \in B$, the laser gadget consists of two thin rectangular extensions to the polygon. They are angled in such a way that the extension of the rectangles intersects only near $(b, \min(b, n^3 - b))$. In particular, any guard g that wants to guard both these rectangular extensions has to be at the boundary of the big polygon and in particular in $(b, \min(b, n^3 - b))$.

The last gadget, called the *disjoiner gadget*, is used to ensure that no guard can guard more than one laser gadget. In fact, any guard that guards more than one edge from the disjoiner gadget must be below the base of the triangle. The disjoiner gadget is formed by placing a small rectangle underneath the base of the polygon, and connect it with a small channel to the base. We place a disconnector gadget between every laser gadget, and before the first and after the last laser gadget.

Since A is sorted, we can perform the first part of the construction in linear time, describing the polygon in a clock-wise order. The remaining part of the construction can also be completed in a clock-wise order: the disjoiners do not rely on the input, and the laser gadgets can be constructed by taking the line from $(b, \min(b, n^3 - b))$ through four points on the x-axis, and intersecting with some well-chosen rectangle below the x-axis.

Lemma 39. *No contiguous guard can guard two points* u *and* v *between which a disjoiner gadget lies. I.e., no contiguous guard can guard a chain* $[u, v]$ *where the open chain* (u, v) *that contains a disjoiner gadget. Also, no contiguous guard can guard two disjoiner gadgets at once.*

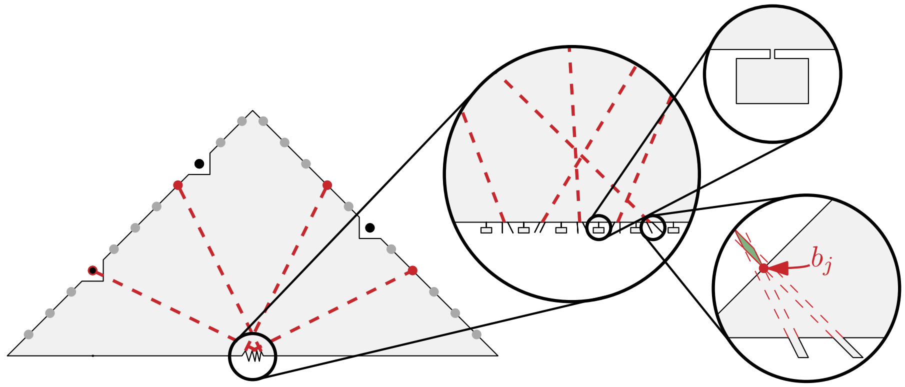

Figure 19: The lower bound construction. Black points are in the set A, red points in the set B. Shown is an instance where $A \cap B \neq \emptyset$.

Proof. Any contiguous guard that guards from u to v has to guard the entirety of the disjoiner gadget. In particular, it must guard the upper right and left corner of the rectangle. Hence, the guard must be placed below the x-axis. Then, this guard cannot guard the points on the base of the triangle that are a small distance $\varepsilon > 0$ right and left of entrance to the disjoiner gadget. Thus this guard cannot see the entire chain $[u, v]$.

As a contiguous guard that sees an interior disjoiner gadget has to be placed inside the rectangle defining the gadget, and no guard can be placed in the rectangle of two different disjoiner gadget, no two disjoiner gadgets can be guard by the same contiguous guard. □

Lemma 40. *The constructed polygon can be guarded by* $2n + 2$ *contiguous guards if and only if* $A \cap B = \emptyset$*.*

Proof. Suppose $A \cap B = \emptyset$. Place one guard in all $n + 1$ disjoiner gadgets, one guard at every $(b, \min(b, n^3 - b))$, and finally one guard at $(n^3/2, 0)$. The first $n + 1$ guards guard the disjoiner gadgets. The fact that $A \cap B = \emptyset$ implies that each point $(b, \min(b, n^3 - b))$ is contained in the polygon. Furthermore, the placement of the laser gadgets in the length 1 interval in the middle of the base implies that the visibility from $(b, \min(b, n^3 - b))$ to the corresponding laser gadget (including the edge of the base to the preceding disjoiner gadget and the edge to the following disjoiner gadget) is not blocked by any other blocker gadget. So, the second n guards guard each of the laser gadgets and the adjacent edges. Finally, the guard at $(n^3/2, 0)$ guards the entire rest of the polygon boundary, that is, from the last disjointer gadget up to the first disjoiner gadget in counter-clockwise direction.

Now suppose, a solution of size $2n + 2$ exists. By Lemma 39, at least $n + 1$ of those must guard the $n + 1$ areas of the polygon that are separated by disjoiner gadgets. That leaves $n + 1$ unguarded disjoiner gadgets, which each require their own guard. Hence each area that is separated by a disjoiner gadget has to be guarded by a single guard. In particular, every laser gadget has to be guarded by a single guard. But this is only possible, if $(b, \min(b, n^3 - b))$ is in the polygon for every $b \in B$. This in turn is only possible, if $B \cap A = \emptyset$, concluding the proof. □

Theorem 41. *Given a polygon* P*, and a integer* k*, deciding whether* ∂P *can be guarded by* k *guards, requires* $\Omega(n \log n)$ *time in the worst case.*

Proof. This is an immediate consequence of Theorem 38 and Lemma 40, together with the fact that the construction from Lemma 40 takes $O(n)$ time, and the size of the coordinates of the vertices is polynomial in the input size. □

Remark 42. The hardness does not depend on the fact that the construction is in general position. In fact, the instance remains hard, even after perturbing every coordinate slightly.

11 Conclusion

We presented matching upper and lower bounds for the Contiguous Art Gallery problem.

A natural direction for further research is to investigate whether our algorithmic ideas extend to polygons with holes. For polygons with holes, there are two natural variants: either one must guard the boundaries of the holes in addition to the outer boundary, or only the outer boundary is required to be guarded. In both cases, at least the outer boundary must be covered. For our approach, however, several crucial components break down when holes are allowed. Not only do key data structures, such as the shortest-path structure, no longer behave as required, but certain structural properties also fail to generalise. In particular, Theorem 7 shows that for simple polygons any solution can be modified so that it contains a good guard. Given the $O(n)$ reduced good dominators, implementing `next` yields the $O(kn \log n)$ algorithm. In polygons with holes, by contrast, there exist configurations in which no such modification is possible (see Figure 20). The essential insight behind the proof of Theorem 7 is that every bad guard is bounded by reflex vertices of the outer boundary, which themselves must be guarded. If instead the bounding vertices occur on holes, we can no longer force the presence of a good guard. It is therefore likely that new structural insights are required to obtain near-linear-time algorithms for polygons with holes, if near-linear time is achievable at all.

Another direction concerns the model of computation. Throughout this work we have ultilised the `realRAM` model, which is standard in computational geometry. One may ask how our results behave under different models of computation. For the classical Art Gallery problem, Abrahamsen, Adamaszek, and Miltzow [1] showed that even when polygon vertices are integers, an optimal solution may require guard positions with irrational coordinates and the `realRAM` is therefore arguably a necessary model of computation. The Contiguous Art Gallery problem does not exhibit this phenomenon. In fact, the existing solutions [8, 38] imply a mild upper bound on bit complexity: if P is described using $\log n$ bits per coordinate, then any optimal solution requires at most $\tilde{O}(kn)$ bits. A formal analysis is given in [38].

Switching computational models affects both the lower and upper bounds. In any comparison-based model, the $\Omega(n \log n)$ lower bound remains valid, but in models such as the `wordRAM`, this lower bound no longer applies. On the upper-bound side, in models where the cost of function evaluation depends on input bit length, our algorithm incurs additional overhead. Our $O(n \log n)$ algorithm computes the k-fold composition of Möbius transforms. After $O(k)$ preprocessing, the resulting composition can be evaluated in $O(1)$ time in the `realRAM` model. In the `wordRAM` model, however, the bit complexity of the composed transform is linear in k, since the parameters become products of the coordinates of k input points. Thus, if the input is representable with n bits, the running time increases to roughly $O(kn \operatorname{polylog} n)$. This raises the question of whether matching upper and lower bounds for Contiguous Art Gallery can also be obtained in the `wordRAM`.

While we do not provide a formal argument, it appears unlikely that any algorithm in the `wordRAM` model can achieve running time $O(k^{2-\varepsilon})$. The k-fold composition of $\texttt{next}(\cdot)$ may contain intervals on which $\texttt{next}^k(u)$ is expressed by a single Möbius transform whose bit complexity is $\Theta(k)$ (see Figure 21). This occurs when a subsequence $[u, \texttt{next}(u)], [\texttt{next}(u), \texttt{next}^2(u)], \ldots, [\texttt{next}^{k-1}(u), \texttt{next}^k(u)]$ of length $\Theta(k)$ is realised by ugly dominators. Consequently, a solution may require $\Theta(k^2)$ bits. We believe that, by combining ideas from Figures 13 and 14 with Figure 21, one can construct instances where every optimal solution has bit complexity $\Omega(k^2)$. This suggests that in the `wordRAM` model, the Contiguous Art Gallery problem may inherently require $\Omega(kn)$ time.

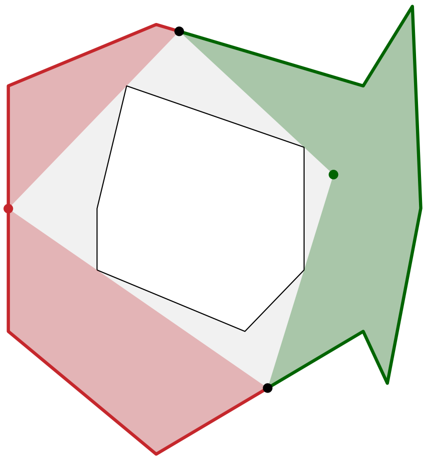

Figure 20: Illustration of a set of guards guarding the outer boundary of a polygon P with a hole. The solution cannot readily be transformed via Theorem 7 to have at least one good guard.

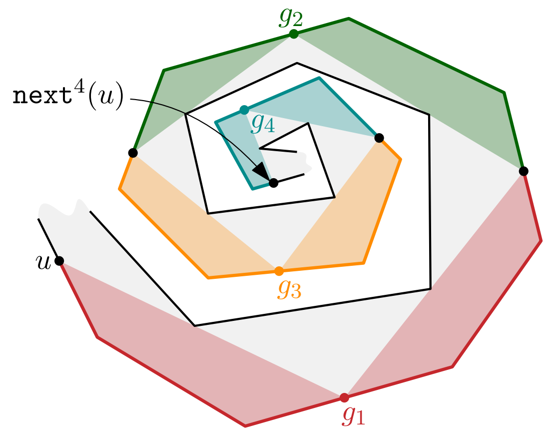

Figure 21: Illustration of compounding bit complexity in the `wordRAM`-model. The bit complexity of $\mathtt{next}^k(u)$ may be linear in k. The bit complexity of a minimal set of guards may be quadratic in k.

References

[1] Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. Irrational Guards are Sometimes Needed. In *Symposium on Computational Geometry (SoCG)*, volume 77 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 3:1–3:15, Dagstuhl, Germany, 2017. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SoCG.2017.3.

[2] Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. In *ACM SIGACT Symposium on Theory of Computing (STOC)*, STOC 2018, page 65–73, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188868.

[3] Pankaj K. Agarwal, Lars Arge, and Ke Yi. An optimal dynamic interval stabbing-max data structure? In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, SODA '05, page 803–812, USA, 2005. Society for Industrial and Applied Mathematics. URL: https://dl.acm.org/doi/10.5555/1070432.1070546.

[4] Alok Aggarwal, Heather Booth, Joseph O'Rourke, Subhash Suri, and Chee K Yap. Finding minimal convex nested polygons. *Information and Computation*, 83(1):98–110, 1989. doi:10.1016/0890-5401(89)90049-7.

[5] Pritam Bhattacharya, Subir Kumar Ghosh, and Bodhayan Roy. Approximability of guarding weak visibility polygons. *Discrete Applied Mathematics*, 228:109–129, 2017. doi:10.1016/j.dam.2016.12.015.

[6] Therese Biedl and Saeed Mehrabi. On r-Guarding Thin Orthogonal Polygons. In *International Symposium on Algorithms and Computation (ISAAC)*, volume 64 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 17:1–17:13, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ISAAC.2016.17, doi:10.4230/LIPIcs.ISAAC.2016.17.

[7] Therese Biedl and Saeed Mehrabi. Grid-obstacle representations with connections to staircase guarding. In *Graph Drawing and Network Visualization (GD)*, volume 10692 of *Lecture Notes in Computer Science*, pages 81–87. Springer, 2017. doi:10.1007/978-3-319-73915-1_7.

[8] Ahmad Biniaz, Anil Maheshwari, Magnus Christian Ring Merrild, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, Eliot W. Robson, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, Thomas Shermer, Jack Spalding-Jamieson, Rolf Svenning, and Da Wei Zheng. Polynomial-Time Algorithms for Contiguous Art Gallery and Related Problems. In *Symposium on Computational Geometry (SoCG)*, volume 332 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:21, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SoCG.2025.20, doi:10.4230/LIPIcs.SoCG.2025.20.

[9] Ahmad Biniaz, Anil Maheshwari, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, and Thomas C. Shermer. Contiguous boundary guarding, 2024. arXiv:2412.15053.

[10] Édouard Bonnet and Tillmann Miltzow. Parameterized hardness of art gallery problems. *ACM Transactions on Algorithms*, 16(4), 2020. doi:10.1145/3398684.

[11] Gerth Stølting Brodal and Riko Jacob. Dynamic planar convex hull. In *Symposium on Foundations of Computer Science (FOCS)*, pages 617–626. IEEE, IEEE Computer Society, 2002. doi:10.1109/SFCS.2002.1181985.

[12] Svante Carlsson, Håkan Jonsson, and Bengt J. Nilsson. Finding the shortest watchman route in a simple polygon. *Discrete & Computational Geometry*, 22(3):377–402, 1999. doi:10.1007/PL00009467.

[13] Timothy M. Chan, John Hershberger, and Simon Pratt. Two approaches to building time-windowed geometric data structures. *Algorithmica*, 81(9):3519–3533, 2019. doi:10.1007/s00453-019-00588-3.

[14] D. Z. Chen, V. Estivill-Castro, and J. Urrutia. Optimal guarding of polygons and monotone chains. In *Canadian Conference on Computational Geometry (CCCG)*, pages 133–138. Carleton University, Ottawa, Canada, 1995.

[15] Wei-Pang Chin and Simeon Ntafos. Shortest watchman routes in simple polygons. *Discrete & Computational Geometry*, 6(1):9–31, 1991. doi:10.1007/BF02574671.

[16] Wen-Chang Chin and Simeon Ntafos. Optimum watchman routes. *Information Processing Letters*, 28(1):39–44, 1988. doi:10.1145/10515.10518.

[17] Václav Chvátal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*, 18(1):39–41, 1975. doi:10.1016/0095-8956(75)90061-1.

[18] David P. Dobkin and David G. Kirkpatrick. Fast detection of polyhedral intersection. *Theoretical Computer Science (TSC)*, 27:241–253, 1983. doi:10.1016/0304-3975(82)90120-7.

[19] Adrian Dumitrescu and Csaba D. Tóth. Watchman tours for polygons with holes. *Computational Geometry: Theory and Applications*, 45(7):326–333, 2012. doi:10.1016/j.comgeo.2011.11.005.

[20] Patrick Eades, Ivor van der Hoog, Maarten Löffler, and Frank Staals. Trajectory Visibility. In *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, volume 162 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 23:1–23:22, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.SWAT.2020.23.

[21] Stephan J. Eidenbenz, Christoph Stamm, and Peter Widmayer. Inapproximability results for guarding polygons and terrains. *Algorithmica*, 31(1):79–113, 2001. doi:10.1007/s00453-001-0040-8.

[22] L. Guibas and J. Hershberger. Optimal shortest path queries in a simple polygon. In *Symposium on Computational Geometry (SoCG)*, page 50–63, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/41958.41964.

[23] Yijie Han. Deterministic sorting in o(n log log n) time and linear space. In *ACM Symposium on Theory of Computing (STOC)*, STOC '02, page 602–608, USA, 2002. Association for Computing Machinery. doi:10.1145/509907.509993.

[24] John Hershberger and Subhash Suri. A pedestrian approach to ray shooting: shoot a ray, take a walk. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, SODA '93, page 54–63, USA, 1993. Society for Industrial and Applied Mathematics. URL: https://dl.acm.org/doi/10.5555/313559.313611.

[25] Hannah Miller Hillberg, Erik Krohn, and Alex Pahlow. On the complexity of half-guarding monotone polygons. In *Latin American Symposium on Theoretical Informatics (LATIN)*, page 761–777, Berlin, Heidelberg, 2022. Springer-Verlag. doi:10.1007/978-3-031-20624-5_46.

[26] Hamid Hoorfar and Alireza Bagheri. Guarding path polygons with orthogonal visibility, 2017. arXiv:1709.01569.

[27] Hamid Hoorfar and Alireza Bagheri. Minimum hidden guarding of histogram polygons, 2017. arXiv:1708.05815.

[28] Erik Krohn and Bengt J. Nilsson. The complexity of guarding monotone polygons. In *Canadian Conference on Computational Geometry (CCCG)*, pages 167–172, Charlottetown, Prince Edward Island, Canada, 2012. CCCG.

[29] Aldo Laurentini. Guarding the walls of an art gallery. *The Visual Computer*, 15(6):265–278, 1999. doi:10.1007/s003710050177.

[30] D. Lee and A. Lin. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory*, 32(2):276–282, 1986. doi:10.1109/TIT.1986.1057165.

[31] D. Lee and F. Preparata. An optimal algorithm for finding the kernel of a polygon. *Journal of the ACM (JACM)*, 26:415–421, 07 1979. doi:10.1145/322139.322142.

[32] Salma Sadat Mahdavi, Saeed Seddighin, and Mohammad Ghodsi. Covering orthogonal polygons with sliding k-transmitters. *Theoretical Computer Science*, 815:163–181, 2020. URL: https://www.sciencedirect.com/science/article/pii/S030439752030092X, doi:10.1016/j.tcs.2020.02.008.

[33] Magnus Christian Ring Merrild, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, and Rolf Svenning. The contiguous art gallery problem is solvable in polynomial time, 2024. arXiv:2412.13938.

[34] Joseph S. B. Mitchell. Approximating watchman routes. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 844–855. SIAM, 2013. doi:10.1137/1.9781611973105.60.

[35] J. O'Rourke and K. Supowit. Some np-hard polygon decomposition problems. *IEEE Transactions on Information Theory*, 29(2):181–190, 1983. doi:10.1109/TIT.1983.1056648.

[36] Leonidas Palios and Petros Tzimas. Minimum r-star cover of class-3 orthogonal polygons. In *International Workshop on Combinatorial Algorithms (IWOCA)*, page 286–297, Berlin, Heidelberg, 2014. Springer-Verlag. doi:10.1007/978-3-319-19315-1_25.

[37] Christian Rieck and Christian Scheffer. The dispersive art gallery problem. *Computational Geometry: Theory and Applications*, 117(C), February 2024. doi:10.1016/j.comgeo.2023.102054.

[38] Eliot W. Robson, Jack Spalding-Jamieson, and Da Wei Zheng. The analytic arc cover problem and its applications to contiguous art gallery, polygon separation, and shape carving, 2024. arXiv:2412.15567.

[39] Dietmar Schuchardt and Hans-Dietrich Hecker. Two np-hard art-gallery problems for ortho-polygons. *Mathematical Logic Quarterly*, 41(2):261–267, 1995. doi:10.1002/malq.19950410212.

[40] Jack Stade. The Point-Boundary Art Gallery Problem Is $\exists\mathbb{R}$-Hard. In *Symposium on Computational Geometry (SoCG)*, volume 332 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 74:1–74:23, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SoCG.2025.74, doi:10.4230/LIPIcs.SoCG.2025.74.

[41] Xuehou Tan. Fast computation of shortest watchman routes in simple polygons. *Information Processing Letters*, 77(1):27–33, 2001. doi:10.1016/S0020-0190(00)00146-0.

[42] Ana Paula Tomás. Guarding thin orthogonal polygons is hard. In *Fundamentals of Computation Theory (FCT)*, FCT'13, page 305–316, Berlin, Heidelberg, 2013. Springer-Verlag. doi:10.1007/978-3-642-40164-0_29.

[43] Arash Vaezi, Bodhayan Roy, and Mohammad Ghodsi. Reflective guarding a gallery. In *WALCOM: Algorithms and Computation*, page 78–89, Berlin, Heidelberg, 2023. Springer-Verlag. doi:10.1007/978-3-031-27051-2_8.

[44] Chris Worman and Mark Keil. Polygon decomposition and the orthogonal art gallery problem. *International Journal of Computational Geometry & Applications*, 17(02):105–138, 2007. doi:10.1142/S0218195907002264.

[45] Andrew Chi-Chih Yao. Lower bounds for algebraic computation trees with integer inputs. *SIAM Journal on Computing*, 20(4):655–668, 1991. doi:10.1137/0220041.