The Parameterized Complexity of Guarding Almost Convex Polygons

Akanksha Agrawal
Ben-Gurion University, Beer-Sheva, Israel, agrawal@post.bgu.ac.il

Kristine V.K. Knudsen
University of Bergen, Bergen, Norway., kristine.knudsen@ii.uib.no

Daniel Lokshtanov
University of California, California, USA, daniello@ucsb.edu

Saket Saurabh
The Institute of Mathematical Sciences, HBNI, Chennai, India, saket@imsc.res.in

Meirav Zehavi
Ben-Gurion University, Beer-Sheva, Israel, meiravze@bgu.ac.il

Abstract

The Art Gallery problem is a fundamental visibility problem in Computational Geometry. The input consists of a simple polygon P, (possibly infinite) sets G and C of points within P, and an integer k; the task is to decide if at most k guards can be placed on points in G so that every point in C is visible to at least one guard. In the classic formulation of Art Gallery, G and C consist of all the points within P. Other well-known variants restrict G and C to consist either of all the points on the boundary of P or of all the vertices of P. Recently, three new important discoveries were made: the above mentioned variants of Art Gallery are all W[1]-hard with respect to k [Bonnet and Miltzow, ESA'16], the classic variant has an $\mathcal{O}(\log k)$-approximation algorithm [Bonnet and Miltzow, SoCG'17], and it may require irrational guards [Abrahamsen et al., SoCG'17]. Building upon the third result, the classic variant and the case where G consists only of all the points on the boundary of P were both shown to be $\exists\mathbb{R}$-complete [Abrahamsen et al., STOC'18]. Even when both G and C consist only of all the points on the boundary of P, the problem is not known to be in NP.

Given the first discovery, the following question was posed by Giannopoulos [Lorentz Center Workshop, 2016]: Is Art Gallery FPT with respect to r, the number of reflex vertices? In light of the developments above, we focus on the variant where G and C consist of all the vertices of P, called Vertex-Vertex Art Gallery. Apart from being a variant of Art Gallery, this case can also be viewed as the classic Dominating Set problem in the visibility graph of a polygon. In this article, we show that the answer to the question by Giannopoulos is *positive*: Vertex-Vertex Art Gallery is solvable in time $r^{\mathcal{O}(r^2)}n^{\mathcal{O}(1)}$. Furthermore, our approach extends to assert that Vertex-Boundary Art Gallery and Boundary-Vertex Art Gallery are both FPT as well. To this end, we utilize structural properties of "almost convex polygons" to present a two-stage reduction from Vertex-Vertex Art Gallery to a new constraint satisfaction problem (whose solution is also provided in this paper) where constraints have arity 2 and involve monotone functions.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability; Theory of computation → Computational geometry

Keywords and phrases Art Gallery, Reflex vertices, Monotone 2-CSP, Parameterized Complexity, Fixed Parameter Tractability

Acknowledgements We thank anonymous reviewers for helpful comments that improved and simplified the paper.

Lines 500

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:37
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Given a *simple* polygon P on n vertices, two points x and y within P are *visible* to each other if the line segment between x and y is contained in P. Accordingly, a set S of points within P is said to *guard* another set Q of points within P if, for every point $q \in Q$, there is some point $s \in S$ such that q and s are visible to each other. The computational problem that arises from this notion is loosely termed the Art Gallery problem. In its general formulation, the input consists of a simple polygon P, possibly infinite sets G and C of points within P, and a non-negative integer k. The task is to decide whether at most k guards can be placed on points in G so that every point in C is visible to at least one guard. The most well-known cases of Art Gallery are identified as follows: the X-Y Art Gallery problem is the Art Gallery problem where G is the set of all points within P (if X=Point), all boundary points of P (if X=Boundary), or all vertices of P (if X=Vertex), and C is defined analogously with respect to Y. The classic variant of Art Gallery is the Point-Point Art Gallery problem. Nevertheless, all variants where X=Vertex or Y=Point received attention in the literature.[1] In particular, Vertex-Vertex Art Gallery is equivalent to the classic Dominating Set problem in the visibility graph of a polygon.

The Art Gallery problem is a fundamental visibility problem in Discrete and Computational Geometry, which was extensively studied from both combinatorial and algorithmic viewpoints. The problem was first proposed by Victor Klee in 1973, which prompted a flurry of results [43, page 1]. The main combinatorial question posed by Klee was *how many guards are sufficient to see every point of the interior of an n-vertex simple polygon?* Chvátal [13] showed in 1975 that $\lfloor \frac{n}{3} \rfloor$ guards are always sufficient and sometimes necessary for any n-vertex simple polygon (see [24] for a simpler proof by Fisk). After this, many variants of the Art Gallery problem, based on different definitions of visibility, restricted classes of polygons, different shapes of guards, and mobility of guards, have been defined and analyzed. Three books [26, 43, 47] and several extensive surveys and book chapters were dedicated to Art Gallery and its variants (see, e.g., [17, 46]). In this article, our main proof states that the Vertex-Vertex Art Gallery problem is *fixed-parameter tractable (*FPT*)* parameterized by r, the number of reflex vertices of P. Additionally, we show that both Vertex-Boundary Art Gallery and Boundary-Vertex Art Gallery are FPT as well.

1.1. Background: Related Algorithmic Works In what follows, we focus only on algorithmic works on X-Y Art Gallery for X,Y$\in$ {Point,Boundary,Vertex}.

Hardness. In 1983, O'Rourke and Supowit [44] proved that Point-Point Art Gallery is NP-hard if the polygon can contain holes. The requirement to allow holes was lifted shortly afterwards [3]. In 1986, Lee and Lin [40] showed that Vertex-Point Art Gallery is NP-hard. This result extends to Vertex-Vertex Art Gallery and Vertex-Boundary Art Gallery. Later, numerous other restricted cases were shown to be NP-hard as well. For example, NP-hardness was established for orthogonal polygons by Katz and Roisman [34] and Schuchardt and Hecker [45]. We remark that the reductions that show that X-Y Art Gallery (for X,Y $\in$ {Point, Boundary, Vertex}) is NP-hard also imply that these cases cannot be solved in time $2^{o(n)}$ under the Exponential-Time Hypothesis (ETH).

[1] The X-Y Art Gallery problem, for any X,Y $\in$ {Point, Boundary, Vertex}, is often loosely termed the Art Gallery problem. For example, in the survey of open problems by Ghosh and Goswami [28], the term Art Gallery problem refers to the Vertex-Vertex Art Gallery problem.

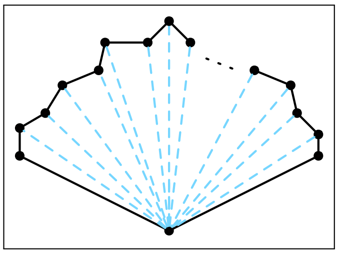

Figure 1 The solution size $k = 1$, yet the number of reflex vertices r is arbitrarily large.

While it has long been known that even very restricted cases of Art Gallery are NP-hard, the inclusion of X-Y Art Gallery, for X,Y $\in$ {Point, Boundary}, in NP remained open. (When X=Vertex, the problem is clearly in NP.) In 2017, Abrahamsen et al. [1] began to reveal the reasons behind this discrepancy for the Point-Point Art Gallery problem: they showed that *exact* solutions to this problem sometimes require placement of guards on points with *irrational* coordinates. Shortly afterwards, they extended this discovery to prove that Point-Point Art Gallery and Boundary-Point Art Gallery are $\exists\mathbb{R}$-complete [2]. Roughly speaking, this result means that *(i)* any system of polynomial equations over the real numbers can be encoded as an instance of Point/Boundary-Point Art Gallery, and *(ii)* these problems are not in the complexity class NP unless NP = $\exists\mathbb{R}$.

Approximation and Exact Algorithms. Due to lack of space, we defer the discussion of known approximation and exact algorithms to Appendix A.

Parameterized Complexity. Two years ago, Bonnet and Miltzow [9] showed that Vertex-Point Art Gallery and Point-Point Art Gallery are W[1]-hard with respect to the *solution size*, k. With straightforward adaptations, their results extend to most of the known variants of the problem, including Vertex-Vertex Art Gallery. Thus, *the classic parameterization by solution size leads to a dead-end.* However, this does not rule out the existence of FPT algorithms for non-trivial structural parametrizations. We refer to the nice surveys by Niedermeier on the art of parameterizations [41, 42].

1.2. Giannopoulos's Parameterization and Our Contribution.

In light of the W[1]-hardness result by Bonnet and Miltzow [9], Giannopoulos [29] proposed to parameterize the Art Gallery problem by the number r of reflex vertices of the input polygon P. Specifically, Giannopoulos [29] posed the following open problem: *"Guarding simple polygons has been recently shown to be* W[1]*-hard w.r.t. the number of (vertex or edge) guards. Is the problem* FPT *w.r.t. the number of reflex vertices of the polygon?"* The motivation behind this proposal is encapsulated by the following well-known proposition, see [43, Sections 2.5-2.6].

▶ **Proposition 1.1** (Folklore)**.** *For any polygon P, the set of reflex vertices of P guards the set of all points within P.*

That is, the minimum number k of guards needed (for any of the cases of Art Gallery) is upper bounded by the number of reflex vertices r. Clearly, k can be arbitrarily smaller than r (see Fig. 1). Our main result is that the Vertex-Vertex Art Gallery problem is FPT parameterized by r. This implies that guarding the vertex set of "almost convex polygons" is easy. In particular, whenever $r^2 \log r = \mathcal{O}(\log n)$, the problem is solvable in polynomial time.

▶ **Theorem 1.** Vertex-Vertex Art Gallery *is* FPT *parameterized by r, the number of reflex vertices. In particular, it admits an algorithm with running time $r^{\mathcal{O}(r^2)} n^{\mathcal{O}(1)}$.*

A few remarks are in place. First, our result extends (with straightforward adaptation) to the most general discrete annotated case of Art Gallery where G and C are each a subset of the vertex set of the polygon, which can include points where the interior angle is of 180 degrees. Consequently, a simple discretization procedure shows that Vertex-Boundary Art Gallery and Boundary-Vertex Art Gallery are both FPT parameterized by r as well. However, we do not know how to handle Vertex-Point Art Gallery and Point-Vertex Art Gallery; determining whether these variants are FPT with respect to r remains open. Second, for variants where both X $\neq$ Vertex and Y $\neq$ Vertex, the design of *exact* algorithms poses extremely difficult challenges. As discussed earlier, these cases are not even known to be in NP; in particular, Point-Point Art gallery is $\exists\mathbb{R}$-hard [2]. Moreover, there is only one known exact algorithm that resolves these cases and it employs extremely powerful machinery (as a black box), not known to be avoidable (see Appendix A). Third, note that our result is among very few *positive* results that concern *optimal* solutions to (any case of) Art Gallery.

Along the way to establish our main result, we prove that a constraint satisfaction problem called Monotone 2-CSP is solvable in polynomial time. This result might be of independent interest. Informally, in Monotone 2-CSP, we are given k variables and m constraints. Each constraint is of the form $[x \,\mathsf{sign}\, f(x')]$ where x and x' are variables, $\mathsf{sign} \in \{\leq, \geq\}$, and f is a *monotone* function. The objective is to assign an integer from $\{0, 1, \ldots, N\}$ to each variable so that all of the constraints will be satisfied. For this problem, we develop a surprisingly simple algorithm based on a reduction to 2-CNF-SAT.

▶ **Theorem 2.** Monotone 2-CSP *is solvable in polynomial time.*

Essentially, the main technical component of our work is an exponential-time reduction that creates an exponential (in r) number of instances of Monotone 2-CSP so that the original instance is a Yes-instance if and only if at least one of the instances of Monotone 2-CSP is a Yes-instance. Our reduction is done in two stages due to its structural complexity. In the first stage of the reduction, we aim to make "guesses" that determine the relations between the "elements" of the problem (that are the "critical" visibility relations in our case) and thereby elucidate and further binarize them (which, in our case, required to impose order on guards). This part requires exponential time (given that there are exponentially many guesses) and captures the "NP-hardness" of the problem. Then, the second stage of the reduction is to translate each guess into an instance of Monotone 2-CSP. This part, while requires polynomial time, relies on highly non-trivial problem-specific insight—specifically, here we need to assert that the relations considered earlier can be encoded by constraints that are not only binary, but that the functions they involve are *monotone*. We strongly believe that our approach can be proven fruitful to resolve the parameterized complexity of other problems of discrete geometric flavour.

2 Our Methods and Preliminaries

Our Methods. The proof of Theorem 1 consists of four components (see Fig. 2). The first component (in Section 3.1) establishes several structural claims regarding monotone properties of visibility in polygons. Informally, we order the vertices of the polygon according to their appearance on the boundary, and consider each sequence between two reflex vertices to be a "convex region". Then, we argue that for every pair of convex regions, as we "move along" one of them, the (index of the) first vertex in the other region that we see either never becomes smaller or never becomes larger. Symmetrically, this claim also holds for the

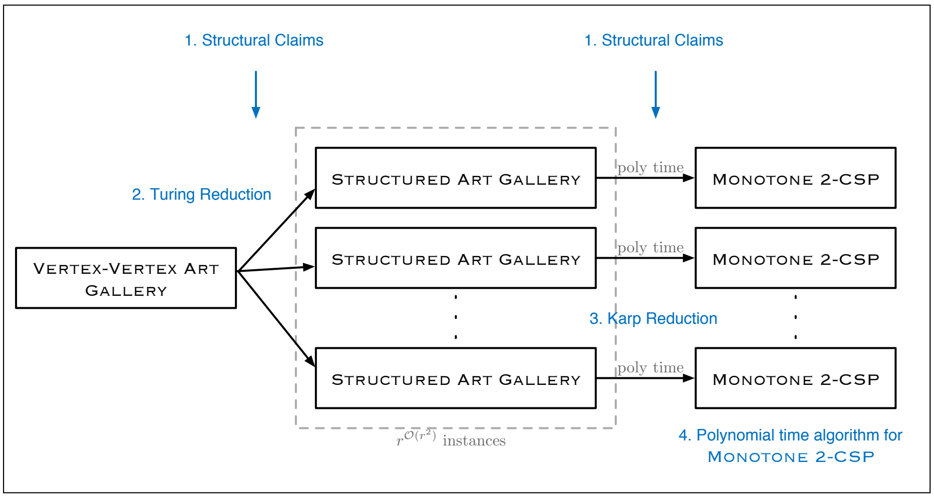

Figure 2 The four components of our proof.

last visible vertices that we encounter. In addition, we argue that if a vertex sees some two vertices in a convex region, then it also sees all vertices in between these two vertices.

Our second component (in Section 3.2) is a Turing reduction to an intermediate problem that we term Structured Art Gallery. Roughly speaking, in this problem, each convex region "announces" how many guards it will contain, and how many guards are necessary to see it completely. In addition, it announces that a prefix of the sequence that forms this region will be guarded by, say, "the i^{th} guard to be placed on region C", then the following subsequence will be guarded by, say, "the j^{th} guard to be placed on region C'", and so on, until it announces how a suffix of it is to be guarded. We stress that the identity of what is "the i^{th} guard to be placed on region C", or what is "the j^{th} guard to be placed on region C'", are of course not known, and should be discovered. Moreover, even the division into subsequences is not known. In Structured Art Gallery, we only focus on solutions that are of the above form. We utilize our second component not only to impose these additional conditions, but also to begin the transition from the usage of visibility-based conditions to function-based constraints. Specifically, functions called first and last will encode, for any vertex v and convex region C, the first and last vertices in C visible to v. To argue that such simple functions encode all necessary information concerning visibility, we make use of the structural claims established earlier.

Our third component (in Section 3.3) is a Karp reduction from Structured Art Gallery to the constraint satisfaction problem, Monotone 2-CSP, discussed in Section 1. This is the part of the proof that most critically relies on all of the structural claims established earlier. Here, we need to translate the constraints imposed by Structured Art Gallery into constraints that comply with the very restricted form of an instance of Monotone 2-CSP, namely, being monotone and involving only two variables. We remark that if one removes the requirement of monotonicity, or allows each constraint to consist of more variables, then the problem can be easily shown to encode Clique and hence become W[1]-hard (see Section 3.3). The translation entails a non-trivial analysis to ensure that all functions are indeed monotone. Specifically, each convex region requires its own set of tailored functions to enforce some relationships between the (unknown) guards it announced to contain and the (unknown) subsequences that these guards are supposed to see. In a sense, our first three components extract the algebraic essence of the Vertex-Vertex Art Gallery problem: by identifying monotone properties and making guesses to ensure binary dependencies between solution elements, the problem is encoded by a restricted constraint satisfaction problem.

Lastly, our fourth component is a relatively simple polynomial-time algorithm for Monotone 2-CSP (see Theorem 2), given in Appendix C, based on a reduction to 2-CNF-SAT. Essentially, the crux is *not* to encode every pair of a variable of Monotone 2-CSP and a potential value for it as a variable of 2-CNF-SAT that signifies equality, because then, although the functions become easily encodable in the language of 2-CNF-SAT, it is unclear how to ensure that each variable of Monotone 2-CSP will be in exactly one pair that corresponds to a variable assigned to truth when satisfying the 2-CNF-SAT formula. Indeed, the naive approach seems futile, because it does not exploit the monotonicity of the input functions. Instead, for each pair of a variable of Monotone 2-CSP and a potential value for it with the exception of 0, we introduce a variable of 2-CNF-SAT signifying that the variable is assigned *at least* the value in the pair. The assignment of value 0 is implicitly encoded by the negation of pairs with the value 1. Then, we can ensure that each variable is assigned exactly one value (when translating a truth assignment for the 2-CNF-SAT instance we created back into an assignment for the Monotone 2-CSP input instance), and by relying on the monotonicty of the input functions, also still be able to encode them correctly in the language of 2-CNF-SAT.

For notational clarity, we describe our proof for Vertex-Vertex Art Gallery. However, all arguments extend in a straightforward manner to solve the annotated generalization of Vertex-Vertex Art Gallery where G and C are each a subset of the vertex set of the polygon. Then, simple discretization procedures yield the positive resolution of the parameterized complexity also of Vertex-Boundary Art Gallery and Boundary-Vertex Art Gallery. For more information, see Appendix G.

Preliminaries. Standard notation not explicitly defined here can be found in Appendix B. We use the abbreviation Art Gallery to refer to Vertex-Vertex Art Gallery. We model a polygon by a graph $P = (V, E)$ with $V = \{1, 2, \ldots, n\}$ and $E = \{\{i, i+1\}\} : i \in \{1, \ldots, n-1\}\} \cup \{\{n, 1\}\}$. For a simple polygon P, we consider the boundary of P as part of its interior. We slightly abuse notation and refer to vertices $i \in V$ where the interior angle of P at i is 180 degrees as convex vertices. We denote the set of reflex vertices of P by $\mathsf{reflex}(P)$, and the set of convex vertices of P by $\mathsf{convex}(P)$. Given a non-convex polygon $P = (V, E)$, we suppose w.l.o.g. that $1 \in V$ is a reflex vertex. We say that a point p *sees* (or is *visible* to) a point q if every point of the line segment $\overline{pq}$ belongs to the interior of P. More generally, a set of points S *sees* a set of points Q if every point in Q is seen by at least one point in S. The definition of a convex polygon asserts that the following holds.

▶ **Observation 2.1.** *Any point within a convex polygon P sees all points within P.*

3 Algorithm for Art Gallery

In this section, we prove that Art Gallery is FPT with respect to r, the number of reflex vertices, by developing an algorithm with running time $2^{\mathcal{O}(r^2 \log r)} n^{\mathcal{O}(1)}$. We first present structural claims that exhibit the monotone way in which vertices in a so called "convex region" see vertices in another such region (Section 3.1). Then, we present a Turing reduction from Art Gallery to a problem called Structured Art Gallery (Section 3.2). Next, based on the claims in Section 3.1, we present our main reduction, which translates Structured Art Gallery to Monotone 2-CSP (Section 3.3). By developing an algorithm for Monotone 2-CSP (Appendix C), we conclude the proof.

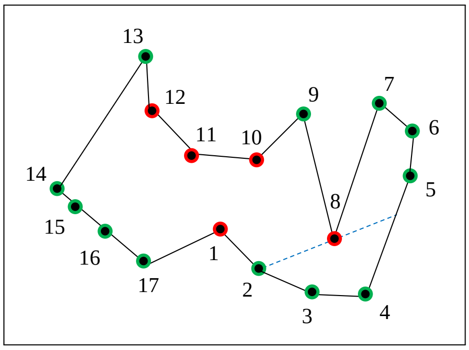

Figure 3 A simple polygon with three maximal convex regions: $[2, 7]$, $[9]$ and $[13, 17]$. Although $2, 5 \in [2, 7]$ belong to the same convex region, they do not see each other.

3.1 Simple Structural Claims

We begin our analysis with the definition of a subsequence of vertices termed a convex region, illustrated in Fig. 3. Below, $j + 1$ for $j = n$ refers to 1. Because we assumed that vertex 1 of any non-convex polygon is a reflex vertex, any convex region $[i, j]$ satisfies $i \neq 1$.

▶ **Definition 3.** *Let $P = (V, E)$ be a simple polygon. A non-empty set of vertices $[i, j] = \{i, i + 1, \ldots, j\}$ is a* convex region *of P if all the vertices in $[i, j]$ are convex. In addition, if $i - 1 \geq 1$ and $j + 1$ are reflex vertices, then $[i, j]$ is a* maximal *convex region.*

In what follows, we would like to argue that for every two (not necessarily distinct) convex regions, one convex region sees the other in a manner that is "monotone" for each "orientation" in which we traverse these regions. To formalize this, we make use of the following notation, illustrated in Fig. 4. For a polygon $P = (V, E)$, a convex region $[i, j]$ of P and a vertex $v \in V$, denote the smallest and largest vertices in $[i, j]$ that are seen by v by $\mathsf{first}(v, [i, j])$ and $\mathsf{last}(v, [i, j])$, respectively. If v sees no vertex in $[i, j]$, define $\mathsf{first}(v, [i, j]) = \mathsf{last}(v, [i, j]) = \mathsf{nil}$. Accordingly, we define two types of monotone views. First, we address the orientation corresponding to first (see Fig. 4). Roughly speaking, we say that the way a convex region $[i, j]$ views a convex region $[i', j']$ is, say, non-decreasing with respect to first, if when we traverse $[i, j]$ from i to j and consider the first vertices in $[i', j']$ that these vertices see, then the sequence of these first vertices (viewed as integers) is a monotonically non-decreasing sequence once we omit all occurrences of nil from it.[2] We further demand that, between two equal vertices in this sequence, no nil occurs. Formally,

▶ **Definition 4.** *Let $P = (V, E)$ be a simple polygon. We say that the way a convex region $[i, j]$ of P views a (not necessarily distinct) convex region $[i', j']$ of P is* non-decreasing *(resp.* non-increasing*) with respect to* first *if for all $t, \widehat{t} \in \{i, i + 1, \ldots, j\}$ such that $t \leq \widehat{t}$, $\mathsf{first}(t, [i', j']) \neq \mathsf{nil}$ and $\mathsf{first}(\widehat{t}, [i', j']) \neq \mathsf{nil}$, we have that*

- *$\mathsf{first}(t, [i', j']) \leq \mathsf{first}(\widehat{t}, [i', j'])$ (resp. $\mathsf{first}(t, [i', j']) \geq \mathsf{first}(\widehat{t}, [i', j'])$), and*
- *if $\mathsf{first}(t, [i', j']) = \mathsf{first}(\widehat{t}, [i', j'])$, then for all $p \in \{t, \ldots, \widehat{t}\}$, $\mathsf{first}(p, [i', j']) = \mathsf{first}(t, [i', j'])$.*[3]

Symmetrically, we address the orientation corresponding to the notation last.

[2] A non-decreasing function (or sequence) is one that *never* decreases but can sometimes *not* increase.
[3] We remark that this condition cannot be replaced by "for all $p \in \{t, \ldots, \widehat{t}\}$, $\mathsf{first}(p, [i', j']) \neq \mathsf{nil}$". For example, in Fig. 4, neither $\mathsf{first}(4, [8, 19])$ nor $\mathsf{first}(6, [8, 19])$ is nil, but $\mathsf{first}(5, [8, 19]) = \mathsf{nil}$.

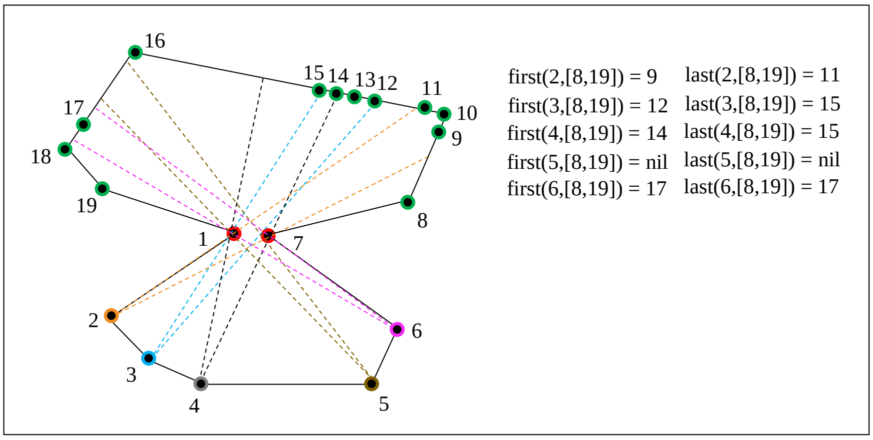

Figure 4 The way $[2,6]$ views $[8,19]$ is non-decreasing with respect to both first and last.

▶ **Definition 5.** *Let $P=(V,E)$ be a simple polygon. We say that the way a convex region $[i,j]$ of P views a (not necessarily distinct) convex region $[i',j']$ of P is* non-decreasing *(resp.* non-increasing*)* with respect to last *if for all $t,\widehat{t}\in\{i,i+1,\ldots,j\}$ such that $t\le\widehat{t}$,* $\mathsf{last}(t,[i',j'])\neq \mathsf{nil}$ *and* $\mathsf{last}(\widehat{t},[i',j'])\neq\mathsf{nil}$*, we have that*

- $\mathsf{last}(t,[i',j'])\le\mathsf{last}(\widehat{t},[i',j'])$ *(resp.* $\mathsf{last}(t,[i',j'])\ge\mathsf{last}(\widehat{t},[i',j'])$*), and*
- *if* $\mathsf{last}(t,[i',j'])=\mathsf{last}(\widehat{t},[i',j'])$*, then for all $p\in\{t,\ldots,\widehat{t}\}$,* $\mathsf{last}(p,[i',j'])=\mathsf{last}(t,[i',j'])$.

The main purpose of this section is to prove the following two lemmas. We believe that some arguments required to establish their proofs might be folklore. For the sake of completeness and self-containment, we present the full details in Appendix D. The first lemma asserts that the subsequence seen by a vertex within a convex region does not contain "gaps".

▶ **Lemma 3.1.** *Let $P=(V,E)$ be a simple polygon, $v\in V$, and $[i,j]$ be a convex region of P. Then, v sees every vertex $t\in[i,j]$ such that* $\mathsf{first}(v,[i,j])\le t\le\mathsf{last}(v,[i,j])$.[4]

The second lemma asserts that views are monotone. Intuitively, whenever we move along a convex region $[i,j]$ while viewing a convex region $[i',j']$ as described earlier, the first vertices (and last vertices) seen form a non-increasing or non-decreasing sequence.[5]

▶ **Lemma 3.2.** *Let $P=(V,E)$ be a simple polygon, and let $[i,j]$ and $[i',j']$ be two (not necessarily distinct) maximal convex regions of P. Then,* (i) *the way in which $[i,j]$ views $[i',j']$ with respect to* first *is either non-decreasing or non-increasing, and* (ii) *the way in which $[i,j]$ views $[i',j']$ with respect to* last *is either non-decreasing or non-increasing.*

3.2 Turing Reduction to Structured Art Gallery

An intermediate step in our reduction from Art Gallery to Monotone 2-CSP addresses an annotated version of Art Gallery, called Structured Art Gallery. Intuitively, in Structured Art Gallery each convex region "announces" how many guards it should contain, and how many guards are to be used to see it completely. In addition, each convex region announces by which *unknown guard* (identified as "the i^{th} guard to be placed on

[4] If v does not see any vertex in $[i,j]$, the claim holds vacuously.
[5] We remark that we do not know whether it is possible that the first vertices would form a non-increasing (or non-decreasing) sequence and the last vertices would not. Our weaker claim suffices for our purposes.

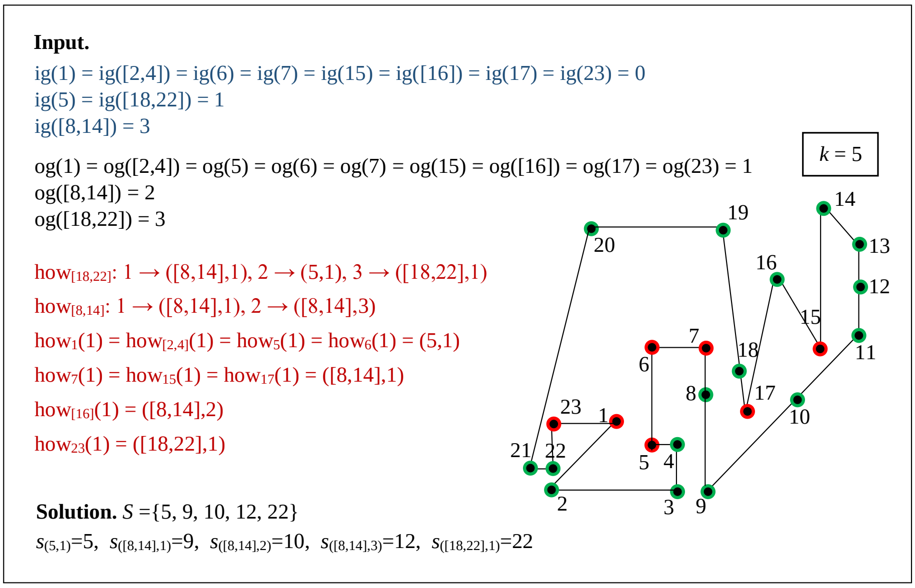

Figure 5 An input and a solution for the Structured Art Gallery problem.

region C" for some i and C) its prefix should be guarded, by which unknown guard a region after this prefix should be guarded, and so on. In what follows, we formally define the Structured Art Gallery problem; then, we present our reduction from Art Gallery to Structured Art Gallery, and afterwards argue that this reduction is correct. For a polygon P, let $\mathcal{C}(P)$ be the set of maximal convex regions of P. Note that $|\mathcal{C}(P)| \leq r$.

Problem Definition. The input of Structured Art Gallery consists of a simple polygon $P = (V, E)$, a non-negative integer $k < r$, and the following functions (see Fig. 5).

- $\mathsf{ig} : \mathcal{C}(P) \cup \mathsf{reflex}(P) \to \{0, \ldots, k\}$, where $\sum_{x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)} \mathsf{ig}(x) \leq k$. Intuitively, for a convex region or reflex vertex x, ig assigns the number of guards to be placed in x.
- $\mathsf{og} : \mathcal{C}(P) \cup \mathsf{reflex}(P) \to \{1, \ldots, k\}$, where for all $x \in \mathsf{reflex}(P)$, $\mathsf{og}(x) = 1$. Intuitively, for a convex region or reflex vertex x, og assigns the number of guards required to see x.
- For each $x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$, $\mathsf{how}_x : \{1, \ldots, \mathsf{og}(x)\} \to (\mathcal{C}(P) \cup \mathsf{reflex}(P)) \times \{1, \ldots, k\}$, where for each (y, i) in the image of how_x, $i \leq \mathsf{ig}(y)$. Intuitively, for any $j \in \{1, \ldots, \mathsf{og}(x)\}$, $\mathsf{how}_x(j) = (y, i)$ indicates that the j^{th} guard required to see x is the i^{th} guard placed in y.

The objective of Structured Art Gallery is to determine whether there exists a set $S \subseteq V$ of size at most k such that the following conditions hold:

1. For each $x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$, $|S \cap x| = \mathsf{ig}(x)$.[6] Accordingly, for each $x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$ and $i \in \{1, \ldots, \mathsf{ig}(x)\}$, let $s_{(x,i)}$ denote the i^{th} largest vertex in $S \cap x$ (see Fig. 5).
2. For each $x \in \mathsf{reflex}(P)$, $s_{\mathsf{how}_x(1)}$ sees x.
3. For each $C \in \mathcal{C}(P)$, the following conditions hold:
   a. $\mathsf{first}(s_{\mathsf{how}_C(1)}, C)$ is the smallest vertex in C.
   b. For every $t \in \{1, \ldots, \mathsf{og}(C) - 1\}$, denote $i = \mathsf{last}(s_{\mathsf{how}_C(t)}, C)$, $j = \mathsf{first}(s_{\mathsf{how}_C(t+1)}, C)$ and $q = \mathsf{last}(s_{\mathsf{how}_C(t+1)}, C)$. Then, *(i)* $i \geq j - 1$, and *(ii)* $i \leq q - 1$. (See Fig. 6.)
   c. $\mathsf{last}(s_{\mathsf{how}_C(\mathsf{og}(C))}, C)$ is the largest vertex in C.

[6] If $x \in \mathsf{reflex}(P)$, by $S \cap x$ we mean $S \cap \{x\}$.

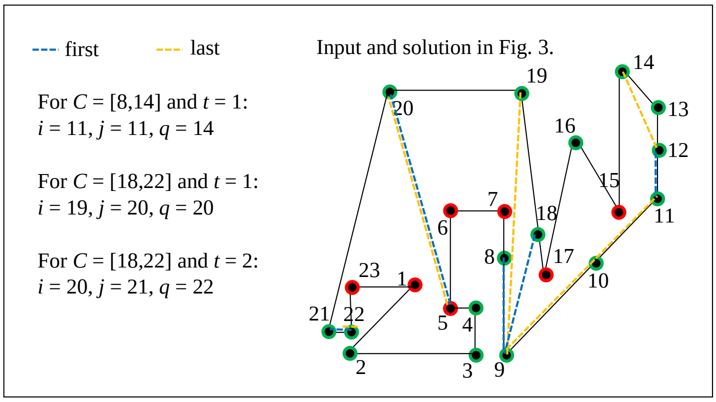

Figure 6 Condition 3b satisfied by a solution for STRUCTURED ART GALLERY.

Informally, Condition 3b states that *(i)* the last vertex in C seen by its t^{th} guard should be at least as large as the predecessor of the first vertex in C seen by its $(t+1)^{th}$ guard, and *(ii)* the last vertex in C seen by its t^{th} guard should be smaller than the last vertex in C seen by its $(t+1)^{th}$ guard. The first condition ensures that no unseen "gaps" are created within C, while the second condition ensures that as the index t grows larger, the last vertex seen by the t^{th} guard grows larger as well. (The second condition will be part of our transition towards the interpretation of the objective of ART GALLERY by *binary* constraints.)

Turing Reduction. Given an instance (P,k) of ART GALLERY, in case $r \leq k$, output YES.[7] Otherwise, the output of the reduction, $\mathsf{reduction}(P,k)$, is the set of all instances $(P,k,\mathsf{ig},\mathsf{og},\{\mathsf{how}_x\}|_{x\in\mathcal{C}(P)\cup\mathsf{reflex}(P)})$ of STRUCTURED ART GALLERY.

Observe that $|\mathcal{C}(P)\cup\mathsf{reflex}(P)| \leq 2r$, and therefore the number of possible functions ig is upper bounded by $(k+1)^{2r}$, the number of possible functions og is upper bounded by k^{2r}, and for each $x \in \mathcal{C}(P)\cup\mathsf{reflex}(P)$, the number of possible functions how_x is upper bounded by $(2rk)^k$. Hence, the number of instances produced is upper bounded by $(k+1)^{2r}\cdot k^{2r}\cdot((2rk)^k)^{2r}$. When $k \leq r$, this number is upper bounded by $r^{\mathcal{O}(r^2)}$. Moreover, the instances in $\mathsf{reduction}(P,k)$ can be enumerated with polynomial delay. Thus,

▶ **Observation 3.1.** *Let (P,k) be an instance of* ART GALLERY*. Then,* $|\mathsf{reduction}(P,k)| = r^{\mathcal{O}(r^2)}$*, and* $\mathsf{reduction}(P,k)$ *is computable in time* $r^{\mathcal{O}(r^2)}n^{\mathcal{O}(1)}$*.*

Correctness. Our proof of correctness crucially relies on Lemma 3.1 and Proposition 1.1.

▶ **Lemma 3.3.** *An instance (P,k) is a* YES*-instance of* ART GALLERY *if and only if there is a* YES*-instance of* STRUCTURED ART GALLERY *in* $\mathsf{reduction}(P,k)$*.*

Proof. Forward Direction. Suppose that (P,k) is a YES-instance of ART GALLERY and that $r > k$. Accordingly, let $S \subseteq V$ be a solution to (P,k). We first define the function $\mathsf{ig}:\mathcal{C}(P)\cup\mathsf{reflex}(P)\rightarrow\{0,\ldots,k\}$ as follows. For each $x\in\mathcal{C}(P)\cup\mathsf{reflex}(P)$, let $\mathsf{ig}(x)=|S\cap x|$. Because $|S|\leq k$ (since S is a solution to (P,k)), we have that $\sum_{x\in\mathcal{C}(P)\cup\mathsf{reflex}(P)}\mathsf{ig}(x)\leq k$. For each $x\in\mathcal{C}(P)\cup\mathsf{reflex}(P)$, we order the vertices in $S\cap x$ from smallest to largest, and denote them accordingly by $s_{(x,1)},s_{(x,2)},\ldots,s_{(x,\mathsf{ig}(x))}$.

[7] To comply with the formal definition of a Turing reduction, by YES we mean a set with a single trivial YES-instance of STRUCTURED ART GALLERY.

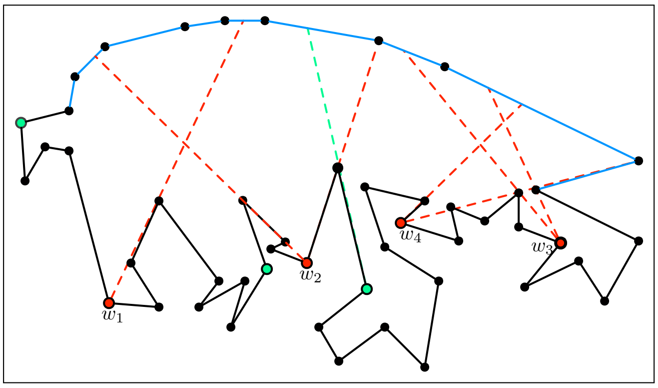

Figure 7 Example of a possible selection of $w_1, w_2, \ldots, w_p$. Solution vertices are colored green and red, and C is colored blue.

Now, we define the functions $\mathsf{og} : \mathcal{C}(P) \cup \mathsf{reflex}(P) \to \{1, \ldots, k\}$ and $\mathsf{how}_x : \{1, \ldots, \mathsf{og}(x)\} \to (\mathcal{C}(P) \cup \mathsf{reflex}(P)) \times \{1, \ldots, k\}$ for all $x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$. For each reflex vertex $x \in \mathsf{reflex}(P)$, define $\mathsf{og}(x) = 1$, and $\mathsf{how}_x(1) = (y, i)$ for some vertex $s_{(y,i)} \in S$ that sees x. The existence of such a vertex $s_{(y,i)}$ follows from the assertion that S is a solution to (P, k). For each convex region $C \in \mathcal{C}(P)$, define $\mathsf{og}(C)$ and how_C as follows. Let W denote the set of vertices in S that see at least one vertex in C. Since W sees C, there exists a vertex in W that sees the smallest vertex in C. Pick such a vertex arbitrarily and denote it by w_1. Now, if w_1 does not see the largest vertex in C, then there exists a vertex in W that sees the smallest vertex in C that is larger than the largest vertex seen by w_1. We pick such a vertex arbitrarily, and denote it by w_2. Next, if w_2 does not see the largest vertex in C, then there exists a vertex in W that sees the smallest vertex in C that is larger than the largest vertex seen by w_2. We pick such a vertex arbitrarily, and denote it by w_3. Similarly, we define $w_4, w_5, \ldots, w_p$, for the appropriate $p \in \{1, \ldots, k\}$ (see Fig. 7). Here, the supposition that $p \leq k$ follows from Lemma 3.1, which implies that $w_i \neq w_j$ for all distinct $i, j \in \{1, \ldots, p\}$. We define $\mathsf{og}(C) = p$, and for all $t \in \{1, \ldots, \mathsf{og}(C)\}$, we define $\mathsf{how}_C(t) = (y, i)$ for the pair $(y, i) \in (\mathcal{C}(P) \cup \mathsf{reflex}(P)) \times \{1, \ldots, k\}$ that satisfies $w_t = s_{(y,i)}$.

Our definitions directly ensure that for each $C \in \mathcal{C}(P)$, the following conditions hold:

1. $\mathsf{first}(s_{\mathsf{how}_C(1)}, C)$ is the smallest vertex in C.
2. For every $t \in \{1, \ldots, \mathsf{og}(C) - 1\}$, denote $i = \mathsf{last}(s_{\mathsf{how}_C(t)}, C)$, $j = \mathsf{first}(s_{\mathsf{how}_C(t+1)}, C)$ and $q = \mathsf{last}(s_{\mathsf{how}_C(t+1)}, C)$. Then, *(i)* $i \geq j - 1$, and *(ii)* $i \leq q - 1$.
3. $\mathsf{last}(s_{\mathsf{how}_C(\mathsf{og}(C))}, C)$ is the largest vertex in C.

By the arguments above, $I = (P, k, \mathsf{ig}, \mathsf{og}, \{\mathsf{how}_x\}|_{x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)})$ is an instance of Structured Art Gallery, and S is a solution to I. Since $I \in \mathsf{reduction}(P, k)$, the proof of the forward direction is complete.

Reverse Direction. If $k \geq r$, then we output Yes (or rather a trivial Yes-instance), and by Proposition 1.1, indeed the input is a Yes-instance as well. Next, suppose that $k < r$, and there is a Yes-instance $I = (P, k, \mathsf{ig}, \mathsf{og}, \{\mathsf{how}_x\}|_{x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)})$ in $\mathsf{reduction}(P, k)$. Accordingly, let $S \subseteq V$ be a solution to I. Then, $|S| \leq k$. Thus, to prove that (P, k) is a Yes-instance of Art Gallery, it suffices to show that S sees V. For each $x \in \mathsf{reflex}(P)$, $s_{\mathsf{how}_x(1)}$ sees x, and therefore S sees $\mathsf{reflex}(P)$.

Now, we show that S sees $\mathsf{convex}(P)$. To this end, we choose a convex region $[i, j] \in \mathcal{C}(P)$, and show that S sees $[i, j]$. Specifically, for each $p \in \{i, \ldots, j\}$, we prove that there is $t \in \{1, \ldots, \mathsf{og}([i, j])\}$ such that $s_{\mathsf{how}_{[i,j]}(t)}$ (which is a vertex in S) sees p. The proof is

$X = \{x_1, x_2, x_3\}$

$C = \{[x_1 \geq f(x_2)], [x_1 \leq g(x_3)], [x_2 \leq h(x_3)], [x_2 \geq 3]\}$

$N = 5$

Solution. $\alpha(x_1) = 2$, $\alpha(x_2) = 3$, $\alpha(x_3) = 5$.

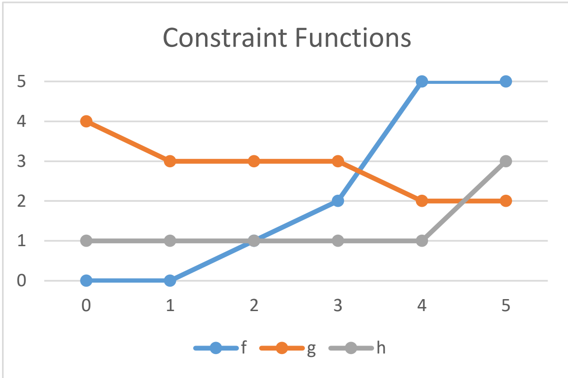

Figure 8 An input for Monotone 2-CSP that has a unique solution.

by induction on p. In the basis, where $p = i$, correctness follows from the assertion that $\mathsf{first}(s_{\mathsf{how}_{[i,j]}(1)}, [i,j])$ is the smallest vertex in $[i,j]$. Now, we suppose that the claim is correct for p, and prove it for $p+1$. By the inductive hypothesis, there is $t \in \{1, \ldots, \mathsf{og}([i,j])\}$ such that $s_{\mathsf{how}_{[i,j]}(t)}$ sees p. If $s_{\mathsf{how}_{[i,j]}(t)}$ sees $p+1$, then we are done. Thus, we now suppose that $s_{\mathsf{how}_{[i,j]}(t)}$ does not see $p+1$. Then, $\mathsf{last}(s_{\mathsf{how}_{[i,j]}(t)}, [i,j]) = p$. We have two cases:

- First, consider the case where $t < \mathsf{og}([i,j])$. Then, because S is a solution to I, the vertex $p = \mathsf{last}(s_{\mathsf{how}_{[i,j]}(t)}, [i,j])$ is larger or equal to $d-1$ for $d = \mathsf{first}(s_{\mathsf{how}_{[i,j]}(t+1)}, [i,j])$. This means that $\mathsf{first}(s_{\mathsf{how}_{[i,j]}(t+1)}, [i,j]) \leq p+1$. Moreover, p is smaller than the vertex $\mathsf{last}(s_{\mathsf{how}_{[i,j]}(t+1)}, [i,j])$. Thus, $p+1 \leq \mathsf{last}(s_{\mathsf{how}_{[i,j]}(t+1)}, [i,j])$. Then, $\mathsf{first}(s_{\mathsf{how}_{[i,j]}(t+1)}, [i,j]) \leq p+1 \leq \mathsf{last}(s_{\mathsf{how}_{[i,j]}(t+1)}, [i,j])$. By Lemma 3.1, this means that $s_{\mathsf{how}_{[i,j]}(t+1)}$ sees $p+1$.
- Second, consider the case where $t = \mathsf{og}([i,j])$. In this case, because S is a solution to I, we have that $\mathsf{last}(s_{\mathsf{how}_{[i,j]}(\mathsf{og}([i,j]))}, [i,j])$ is the largest vertex in $[i,j]$. Thus, $p+1 \leq \mathsf{last}(s_{\mathsf{how}_{[i,j]}(\mathsf{og}([i,j]))}, [i,j])$, which is a contradiction.

This completes the proof. ◀

3.3 Karp Reduction to Monotone 2-CSP

We proceed to the second part of our proof, a reduction from Structured Art Gallery to Monotone 2-CSP.[8] The analysis of the reduction is given in Appendix F.

Problem Definition. The input of Monotone 2-CSP consists of a set X of *variables*, denoted by $X = \{x_1, x_2, \ldots, x_{|X|}\}$, a set C of *constraints*, and $N \in \mathbb{N}$ given in unary. Each constraint $c \in C$ has the form $[x_i \,\mathsf{sign}\, f(x_j)]$ where $i, j \in \{1, \ldots, |X|\}$, $\mathsf{sign} \in \{\geq, \leq\}$ and $f : \{0, \ldots, N\} \to \{0, \ldots, N\}$ is a monotone function. An assignment $\alpha : X \to \{0, \ldots, N\}$ *satisfies* a constraint $c = [x_i \,\mathsf{sign}\, f(x_j)] \in C$ if $[\alpha(x_i) \,\mathsf{sign}\, f(\alpha(x_j))]$ is true. The objective of Monotone 2-CSP is to decide if there exists an assignment $\alpha : X \to \{0, \ldots, N\}$ that satisfies all the constraints in C (see Fig. 11).

If the function f of a constraint $c = [x_i \,\mathsf{sign}\, f(x_j)]$ is constantly β (that is, for every $t \in \{0, \ldots, N\}$, $f(t) = \beta$), then we use the shorthand $c = [x_i \,\mathsf{sign}\, \beta]$. Moreover, we suppose that every constraint represented by a quadruple is associated with two distinct variables.

[8] CSP is an abbreviation of Constraint Satisfaction Problem, and 2 is the maximum arity of a constraint.

Karp Reduction. Given an instance $I = (P, k, \mathsf{ig}, \mathsf{og}, \{\mathsf{how}_x\}|_{x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)})$ of Structured Art Gallery, define an instance $\mathsf{reduction}(I) = (X, C, N)$ of Monotone 2-CSP as follows. Let $k^\star = \sum_{e \in \mathcal{C}(P) \cup \mathsf{reflex}(P)} \mathsf{ig}(e)$, $X = \{x_1, x_2, \ldots, x_{k^\star}\}$ and $N = n + 1$. (Here, $n = |V|$.) Additionally, let bij be an arbitrary bijective function from X to $\{(e, i) : e \in \mathcal{C}(P) \cup \mathsf{reflex}(P), i \in \{1, \ldots, \mathsf{ig}(e)\}\}$. Intuitively, for any variable $x \in X$ with $\mathsf{bij}(x) = (e, i)$, we think of x as the i^{th} guard to be placed in region e. In particular, the value to be assigned to x is the identity of this guard. The values 0 and $n + 1$ are not identities of vertices in V, and we will ensure that no solution assignment assigns them; we note that these two values are useful because they will allow us to exclude assignments that should not be solutions. Next, we define our constraints and show that their functions are monotone.

Association. For each $x \in X$ with $\mathsf{bij}(x) = (e, i)$, we need to ensure that the vertex assigned to x is within the region e. To this end, we introduce the following constraints.

- If $e \in \mathsf{reflex}(P)$, then insert the constraint $[x = e]$. (That is, insert $[x \le e]$ and $[x \ge e]$.)
- Else, $\mathsf{bij}(x) = (e, j)$ for $e \in \mathcal{C}(P)$. Let ℓ and h be the smallest and largest vertices in e, respectively, and insert the constraints $[x \ge \ell]$ and $[x \le h]$.

Let A denote this set of constraints.

Order in a convex region. For all $x, x' \in X$ where $\mathsf{bij}(x) = (C, i)$ and $\mathsf{bij}(x') = (C, j)$ for the same convex region $C \in \mathcal{C}(P)$ and $i < j$, we need to ensure that the vertex assigned to x' is larger than the one assigned to x. To this end, we introduce the constraint $[x' \ge f(x)]$ where f is defined as follows. For all $q \in \{0, \ldots, N - 1\}$, $f(q) = q + 1$, and $f(N) = N$. Let O denote this set of constraints. We note that the constraints in $A \cup O$ together enforce each variable $x \in X$ with $\mathsf{bij}(x) = (C, i)$ for $C \in \mathcal{C}(P)$ to be assigned the i^{th} guard placed in C.

Guarding reflex vertices. For every reflex vertex $y \in \mathsf{reflex}(P)$ with $\mathsf{how}_y(1) = (e, i)$, we need to ensure that the vertex assigned to $x = \mathsf{bij}^{-1}(e, i)$ sees y. To this end, consider two cases. First, suppose that $e \in \mathsf{reflex}(P)$. Then, *(i)* if e does not see y, output No, and *(ii)* else, no constraint is introduced. Second, suppose that $e \in \mathcal{C}(P)$. Denote $\ell = \mathsf{first}(y, e)$ and $h = \mathsf{last}(y, e)$. Then, *(i)* if ℓ (and thus also h) is nil, then output No, and *(ii)* else, introduce the constraints $c^1_y = [x \ge \ell]$ and $c^2_y = [x \le h]$.

Guarding first vertices in convex regions. For every convex region $C = [q, q'] \in \mathcal{C}(P)$ with $\mathsf{how}_C(1) = (e, i)$, we need to ensure that the vertex assigned to $x = \mathsf{bij}^{-1}(e, i)$ sees q, the first vertex of C. To this end, consider two cases. First, suppose that $e \in \mathsf{reflex}(P)$. Then, *(i)* if e does not see q, output No, and *(ii)* else, no constraint is introduced. Second, suppose that $e \in \mathcal{C}(P)$. Denote $\ell = \mathsf{first}(q, e)$ and $h = \mathsf{last}(q, e)$. Then, *(i)* if ℓ is nil, then output No, and *(ii)* else, insert the constraints $c^1_{(C,1)} = [x \ge \ell]$ and $c^2_{(C,1)} = [x \le h]$.

Guarding last vertices in convex regions. For every convex region $C = [q, q'] \in \mathcal{C}(P)$ with $\mathsf{how}_C(\mathsf{og}(C)) = (e, i)$, we need to ensure that the vertex assigned to $x = \mathsf{bij}^{-1}(e, i)$ sees q', the last vertex of C. To this end, consider two cases. First, suppose that $e \in \mathsf{reflex}(P)$. Then, *(i)* if e does not see q', output No, and *(ii)* else, no constraint is introduced. Second, suppose that $e \in \mathcal{C}(P)$. Denote $\ell = \mathsf{first}(q', e)$ and $h = \mathsf{last}(q', e)$. Then, *(i)* if ℓ is nil, then output No, and *(ii)* else, insert the constraints $c^1_{(C,\mathsf{og}(C))} = [x \ge \ell]$ and $c^2_{(C,\mathsf{og}(C))} = [x \le h]$.

Guarding middle vertices in convex regions. For every convex region $C \in \mathcal{C}(P)$ and $t \in \{2, \ldots, \mathsf{og}(C)\}$, we introduce four constraints based on the following notation.

- $(e, \gamma) = \mathsf{how}_C(t)$ and $x = \mathsf{bij}^{-1}(e, \gamma)$. Intuitively, the t^{th} vertex to guard C should be the γ^{th} guard to be placed in e, and its precise identity should be assigned to x. If no vertex in e sees at least one vertex in C, then return No.[9] Let ℓ and h be the smallest and largest vertices in e that see at least one vertex in C, respectively.

- $(e', \gamma') = \mathsf{how}_C(t-1)$ and $x' = \mathsf{bij}^{-1}(e', \gamma')$. Intuitively, the $(t-1)^{\text{th}}$ vertex to guard C should be the γ'^{th} guard to be placed in e', and its precise identity should be assigned to x'. If no vertex in e' sees at least one vertex in C, then return No. Let ℓ' and h' be the smallest and largest vertices in e' that see at least one vertex in C, respectively.

Now, insert the constraints $\widetilde{c}^1_{(C,t)} = [x \geq \ell]$ and $\widetilde{c}^2_{(C,t)} = [x \leq h]$. Intuitively, these two constraints *help* to ensure that x will be assigned a vertex that sees at least one vertex in C. However, these constraints alone are insufficient for this task—ensuring that we pick a guard between two vertices that see vertices in C does not ensure that this guard sees vertices in C.[10] Nevertheless, combined with our final constraints, this task is achieved.

Lastly, we consider two sets of four cases. The first set introduces a constraint to ensure that x, which stands for the t^{th} vertex to guard C, should satisfy that the first vertex in C seen by x is smaller or equal than the vertex larger by 1 than the last vertex in C seen by x', which stands for the $(t-1)^{\text{th}}$ vertex to guard C. On the other hand, the second set introduces a constraint to ensure that the last vertex in C seen by x is larger than the last vertex in C seen by x'. Together, because views have no "gaps", this would imply that x sees the vertex in C that is larger by 1 than the last vertex in C seen by x'. Due to lack of space, we only present the first case of each set. Omitted details can be found in Appendix E. To unify notation, if e (or e') is a reflex vertex, we say that the way e (or e') views C is non-decreasing with respect to both first and last.

First, consider the case where the way e' views C is non-decreasing with respect to last, and the way e views C is non-decreasing with respect to first. We insert a constraint $[x \leq f(x')]$, where f (having domain and range $\{0, \ldots, N\}$) is defined as follows.

- For all $i < \ell'$: $f(i) = 0$. Intuitively, we forbid x to be assigned a vertex smaller than the first vertex in e that can see C.
- For $i = \ell', \ell'+1, \ldots, h'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
  - If **(i)** $a = \mathsf{nil}$, **(ii)** $a + 1 \notin C$, or **(iii)** $\mathsf{first}(j, C) \leq a + 1$ for no $j \in e$, let $f(i) = f(i-1)$. Roughly speaking, given that x' sees C, $a \neq \mathsf{nil}$ (in cases we will care about). Moreover, $a + 1 \in C$ will be ensured by the second set of cases and the way we guard the last vertex of a convex region. Lastly, $\mathsf{first}(j, C) \leq a + 1$ for some $j \in e$ will be ensured using that $f(i-1)$ (unless $f(i-1) = 0$) is a vertex that sees $a + 1$.
  - Else, let j be the largest vertex in e such that $\mathsf{first}(j, C) \leq a + 1$. Define $f(i) = j$. Intuitively, by enforcing x to be smaller or equal than j—the largest vertex in e that might see $a + 1$—we ensure that the following condition holds: the first vertex x sees in C, under the assumption that it is not nil,[11] is smaller or equal to $a + 1$ (because the way e views C is non-decreasing with respect to first).
- For all $i > h'$: $f(i) = N$.

Second, consider the case where the ways e' and e view C are both non-decreasing with respect to last. We insert a constraint $[x \geq f(x')]$, where f is defined as follows.

- For all $i > h'$: $f(i) = N$.
- For $i = h', h'-1, \ldots, \ell'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.

---

[9] In case $e \in \mathsf{reflex}(P)$, we mean that e itself does not see any vertex in C.

[10] For example, in Fig. 4, neither $\mathsf{first}(4, [8, 19])$ nor $\mathsf{first}(6, [8, 19])$ is nil, but $\mathsf{first}(5, [8, 19]) = \mathsf{nil}$.

[11] In the proof, to ensure that this vertex is indeed not nil, we will utilize both sets of cases, together with $\widetilde{c}^1_{(C,t)}$ and $\widetilde{c}^2_{(C,t)}$, to argue that x is between two vertices seen by $a + 1$ and hence must see $a + 1$ itself.

  - If **(i)** $a = \mathsf{nil}$, **(ii)** $a + 1 \notin C$, or **(iii)** $\mathsf{last}(j, C) \geq a + 1$ for no $j \in e$, let $f(i) = f(i+1)$.
  - Else, let j be the smallest vertex in e such that $\mathsf{last}(j, C) \geq a + 1$. Define $f(i) = j$.
- For all $i < \ell'$: $f(i) = 0$.

Here, as the sign is $\geq$ and f is monotonically non-decreasing, f must be defined first for N, then for $N - 1$, and so on. Then, as long as i is such that $\mathsf{last}(j, C) \geq a + 1$ for no $j \in e$ (a case that we want to avoid), $f(i) = N$ and hence $[x \geq f(i)]$ cannot be satisfied.

References

1 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. Irrational guards are sometimes needed. In *33rd International Symposium on Computational Geometry (SoCG)*, pages 3:1–3:15, 2017.

2 Mikkel Abrahamsen, Anna Adamaszek, and Tillmann Miltzow. The art gallery problem is $\exists\mathbb{R}$-complete. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 65–73, 2018.

3 Alok Aggarwal. *The art gallery theorem: its variations, applications and algorithmic aspects*. PhD thesis, The Johns Hopkins University, Baltimore, Maryland, 1986.

4 Bengt Aspvall, Michael F. Plass, and Robert Endre Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979.

5 David Avis and Godfried T. Toussaint. An efficient algorithm for decomposing a polygon into star-shaped polygons. *Pattern Recognition*, 13(6):395–398, 1981.

6 Saugata Basu, Richard Pollack, and Marie-Françoise Roy. On the combinatorial and algebraic complexity of quantifier elimination. *Journal of the ACM*, 43(6):1002–1045, 1996.

7 Pritam Bhattacharya, Subir Kumar Ghosh, and Sudebkumar Prasant Pal. Constant approximation algorithms for guarding simple polygons using vertex guards. *CoRR/arXiv*, abs/1712.05492, 2017.

8 Pritam Bhattacharya, Subir Kumar Ghosh, and Bodhayan Roy. Approximability of guarding weak visibility polygons. *Discrete Applied Mathematics*, 228:109–129, 2017.

9 Édouard Bonnet and Tillmann Miltzow. Parameterized hardness of art gallery problems. In *Proceedings of the 24th Annual European Symposium on Algorithms (ESA)*, pages 19:1–19:17, 2016.

10 Édouard Bonnet and Tillmann Miltzow. An approximation algorithm for the art gallery problem. In *Proceedings of the 33rd International Symposium on Computational Geometry (SoCG)*, pages 20:1–20:15, 2017.

11 Dorit Borrmann, Pedro J. de Rezende, Cid C. de Souza, Sándor P. Fekete, Stephan Friedrichs, Alexander Kröller, Andreas Nüchter, Christiane Schmidt, and Davi C. Tozoni. Point guards and point clouds: solving general art gallery problems. In *Symposuim on Computational Geometry (SoCG)*, pages 347–348, 2013.

12 Hervé Brönnimann and Michael T. Goodrich. Almost optimal set covers in finite vc-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.

13 Vasek Chvátal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*, 18(1):39–741, 1975.

14 Kenneth L. Clarkson. Algorithms for polytope covering and approximation. In *Proceedings of the third workshop on Algorithms and Data Structures (WADS)*, pages 246–252, 1993.

15 Marcelo C. Couto, Pedro J. de Rezende, and Cid C. de Souza. An exact algorithm for minimizing vertex guards on art galleries. *International Transactions in Operational Research*, 18(4):425–448, 2011.

16 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

17 Pedro J. de Rezende, Cid C. de Souza, Stephan Friedrichs, Michael Hemmer, Alexander Kröller, and Davi C. Tozoni. Engineering art galleries. In *Algorithm Engineering - Selected Results and Surveys*, pages 379–417. Springer, 2016.

18 Ajay Deshpande, Taejung Kim, Erik D. Demaine, and Sanjay E. Sarma. A pseudopolynomial time $O(\log n)$-approximation algorithm for art gallery problems. In *Proceedings of the 10th International Workshop on Algorithms and Data Structures (WADS)*, pages 163–174, 2007.

19 R Diestel. *Graph Theory, 4th Edition*. Springer, 2012.

20 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

21 Alon Efrat and Sariel Har-Peled. Guarding galleries and terrains. *Information Processing Letters*, 100(6):238–245, 2006.

22 Stephan Eidenbenz, Christoph Stamm, and Peter Widmayer. Inapproximability of some art gallery problems. In *Proceedings of the 10th Canadian Conference on Computational Geometry (CCCG)*, 1998.

23 Stephan Eidenbenz, Christoph Stamm, and Peter Widmayer. Inapproximability results for guarding polygons and terrains. *Algorithmica*, 31(1):79–113, 2001.

24 Steve Fisk. A short proof of Chvátal's watchman theorem. *Journal of Combinatorial Theory, Series B*, 24(3):374, 1978.

25 Subir Kumar Ghosh. Approximation algorithms for art gallery problems. In *Canadian Information Processing Society Congress*, pages 429–434, 1987.

26 Subir Kumar Ghosh. *Visibility algorithms in the plane*. Cambridge university press, 2007.

27 Subir Kumar Ghosh. Approximation algorithms for art gallery problems in polygons. *Discrete Applied Mathematics*, 158(6):718–722, 2010.

28 Subir Kumar Ghosh and Partha P. Goswami. Unsolved problems in visibility graphs of points, segments, and polygons. *ACM Computing Surveys*, 46(2):22:1–22:29, 2013.

29 P Giannopoulos. Open problems: Guarding problems. *Lorentz Workshop on Fixed-Parameter Computational Geometry, Leiden, the Netherlands*, page 12, 2016.

30 Alexander Gilbers and Rolf Klein. A new upper bound for the vc-dimension of visibility regions. *Computational Geometry*, 47(1):61–74, 2014.

31 P Golovach, D Lokshtanov, S Saurabh, and M Zehavi. Cliquewidth III: The odd case of graph coloring parameterized by cliquewidth. In *28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 262–273, 2018.

32 Gil Kalai and Jiří Matoušek. Guarding galleries where every point sees a large area. *Israel Journal of Mathematics*, 101(1):125–139, 1997.

33 Matthew J. Katz. A PTAS for vertex guarding weakly-visible polygons - an extended abstract. *CoRR*, abs/1803.02160, 2018.

34 Matthew J Katz and Gabriel S Roisman. On guarding the vertices of rectilinear domains. *Computational Geometry*, 39(3):219–228, 2008.

35 James King. Fast vertex guarding for polygons with and without holes. *Computational Geometry*, 46(3):219 – 231, 2013.

36 James King and David G. Kirkpatrick. Improved approximation for guarding simple galleries from the perimeter. *Discrete & Computational Geometry*, 46(2):252–269, 2011.

37 David G. Kirkpatrick. An $O(\log \log \mathsf{OPT})$-approximation algorithm for multi-guarding galleries. *Discrete & Computational Geometry*, 53(2):327–343, 2015.

38 Ali A. Kooshesh and Bernard M. E. Moret. Three-coloring the vertices of a triangulated simple polygon. *Pattern Recognition*, 25(4):443, 1992.

39 Erik Krohn and Bengt J. Nilsson. Approximate guarding of monotone and rectilinear polygons. *Algorithmica*, 66(3):564–594, 2013.

40 D Lee and Arthurk Lin. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory*, 32(2):276–282, 1986.

41 Rolf Niedermeier. Ubiquitous parameterization - invitation to fixed-parameter algorithms. In *Proceedings of the 29th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 84–103, 2004.

42 Rolf Niedermeier. Reflections on multivariate algorithmics and problem parameterization. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 17–32, 2010.

43 Joseph O'rourke. *Art gallery theorems and algorithms*, volume 57. Oxford University Press Oxford, 1987.

44 Joseph O'Rourke and Kenneth J. Supowit. Some NP-hard polygon decomposition problems. *IEEE Transactions on Information Theory*, 29(2):181–189, 1983.

45 Dietmar Schuchardt and Hans-Dietrich Hecker. Two NP-hard art-gallery problems for ortho-polygons. *Mathematical Logic Quarterly*, 41(2):261–267, 1995.

46 Thomas C Shermer. Recent results in art galleries (geometry). *Proceedings of the IEEE*, 80(9):1384–1399, 1992.

47 Jorge Urrutia. Art gallery and illumination problems. *Handbook of computational geometry*, 1(1):973–1027, 2000.

48 Pavel Valtr. Guarding galleries where no point sees a small area. *Israel Journal of Mathematics*, 104(1):1–16, 1998.

A Known Approximation and Exact Algorithms

Approximation Algorithms. The Art Gallery problem has been extensively studied from the viewpoint of approximation algorithms [21, 18, 27, 35, 36, 39, 37, 10, 8, 7, 33] (this list is not comprehensive). Most of these approximation algorithms are based on the fact that the range space defined by the visibility regions has bounded VC-dimension for simple polygons [30, 32, 48], which facilitates the usage of the algorithmic ideas of Clarkson [12, 14]. The current state-of-the-art is as follows. For the Boundary-Point Art Gallery problem, King and Kirkpatrick [36] gave a factor $\mathcal{O}(\log \log \mathsf{OPT})$ approximation algorithm. For the Point-Point Art Gallery problem, Bonnet and Miltzow [10] gave a factor $\mathcal{O}(\log \mathsf{OPT})$ approximation algorithm. Very recently, in a yet unpublished work, Bhattacharya et al. [7] reported a breakthrough: they designed an 18-approximation algorithm for Vertex-Vertex Art Gallery, a (slightly slower) 18-approximation algorithm for Vertex-Boundary Art Gallery, and a 27-approximation algorithm for Vertex-Point Art Gallery. For all of these three variants, the existence of a constant-factor approximation algorithm has been a longstanding open problem, conjectured to be true already in 1987 by Ghosh [25, 27, 28]. The existence of a constant-factor approximation algorithm for Point-Point Art Gallery (or even Boundary-Boundary Art Gallery or Boundary-Point Art Gallery) remains a major open problem. On the negative side, all of these variants are known to be APX-hard [22, 23]. However, restricted classes of polygons, such as weakly-visible polygons [33], give rise to a PTAS.

Exact Algorithms. For an n-vertex polygon P, one can efficiently find a set of $\lfloor \frac{n}{3} \rfloor$ vertices that guard all points within P, matching Chvátal's upper bound [13]. Specifically, Avis and Toussaint [5] presented an $\mathcal{O}(n \log n)$-time divide-and-conquer algorithm for this task. Later, Kooshesh and Moret [38] gave a linear-time algorithm based on Fisk's short proof [24]. However, when we seek an optimal solution, the situation is much more complicated. The first exact algorithm for Point-Point Art Gallery was published in 2002 in the conference version of a paper by Efrat and Har-Peled [21]. They attribute the result to Micha Sharir. Before that time, the problem was not even known to be decidable. The algorithm computes a formula in the first order theory of the reals corresponding to the art gallery instance (with both existential and universal quantifiers), and employs algebraic methods such as the techniques provided by Basu et al. [6], to decide if the formula is true. Given that Point-Point Art Gallery is $\exists\mathbb{R}$-complete [2], it might not be possible to avoid the use of this powerful machinery. However, even for the cases where X=Vertex, the situation is quite grim; we are not aware of *exact* algorithms that achieve substantially better time complexity bounds than brute-force. Nevertheless, over the years, exact algorithms that perform well in practice were developed. For example, see [11, 17, 15].

B Full Preliminaries

We use standard terminology from the book of Diestel [19]. With the exception of the Introduction, the abbreviation Art Gallery refers to Vertex-Vertex Art Gallery.

Polygons. A *simple polygon* P is a flat shape consisting of n straight, *non-intersecting* line segments that are joined pair-wise to form a closed path. The line segments that make up a polygon, called *edges*, meet only at their endpoints, called *vertices*. Any polygon can be modeled by a graph $P = (V, E)$ with $V = \{1, 2, \dots, n\}$ and $E = \{\{i, i+1\}\} : i \in \{1, \dots, n-1\}\} \cup \{\{n, 1\}\}$ where every vertex $i \in V$ is associated with a point (x_i, y_i) on the Euclidean plane. A simple polygon P encloses a region, called its *interior*, that has a measurable area. We consider the boundary of P as part of its interior. A vertex $i \in V$ is a *reflex* (resp. *convex*) vertex if the interior angle of P at i is larger (resp. smaller) than 180 degrees. If $i \in V$ is not a reflex vertex, then either i is a convex vertex or the interior angle of P at i is exactly 180 degrees. We slightly abuse notation and refer to all non-reflex vertices as convex vertices. We denote the set of reflex vertices of P by $\mathsf{reflex}(P)$, and the set of convex vertices of P by $\mathsf{convex}(P)$. A *convex polygon* P is a simple polygon such that for every two points p and q on the boundary (or interior) of P, no point of the line segment $\overline{pq}$ is strictly outside P. In a convex polygon, all interior angles are less than or equal to 180 degrees, while in a strictly convex polygon all interior angles are less than 180 degrees. Given a non-convex polygon $P = (V, E)$, we suppose w.l.o.g. that $1 \in V$ is a reflex vertex.

Visibility. Let $P = (V, E)$ be a simple polygon. We say that a point p *sees* (or is *visible* to) a point q if every point of the line segment $\overline{pq}$ belongs to the interior (including the boundary) of P. More generally, a set of points S *sees* a set of points Q if every point in Q is seen by at least one point in S. Note that if a point p sees a point q, then the point q sees the point p as well. Moreover, a vertex $v \in V$ necessarily sees itself and its two neighbors in P. The definition of a convex polygon asserts that the following observation holds.

▶ **Observation B.1.** *Any point within a convex polygon P sees all points within P.*

Parameterized Complexity. Every instance of a parameterized problem is accompanied by a parameter k. A parameterized problem Π is *fixed-parameter tractable (*FPT*)* if there is an algorithm that, given an instance (I, k) of Π, solves it in time $f(k)|I|^{\mathcal{O}(1)}$ where f is some computable function of k. Under reasonable complexity-theoretic assumptions, there are parameterized problems (such as W[1]-hard problems) that are not FPT. For more information, we refer the reader to monographs such as [16, 20].

C Algorithm for Monotone 2-CSP

In this section, we design a polynomial time algorithm for Monotone 2-CSP, running in time $\mathcal{O}((|X| + |C|) \cdot N)$. We obtain this algorithm by reducing the given instance (X, C, N) to an instance of 2-SAT. We note that without monotonicity or arity bound, the problem is W[1]-hard, while when we have both these conditions (and arity is at most two), then our algorithm shows that the problem is polynomial time solvable. Indeed, to see the necessity for monotonicity, consider a reduction from Multicolored Clique to 2-CSP as follows. For each vertex and edge in the hypothetical solution, we create a variable. That is, we have a variable x_i, for each $i \in [k]$, and for every distinct $i, j \in [k]$, where $i < j$, we have a variable e_{ij}. We can define two functions f^1_{ij} and f^2_{ij} which return the vertex from i^{th} and j^{th} part incident to the edge e_{ij}, respectively. Now we add constraints of the form $x_i = f^1_{ij}(e_{ij})$ and $x_j = f^1_{ij}(e_{ji})$, for $i < j$. Notice that the selected set of vertices and edges form a clique if and only if the 2-CSP is satisfied for the respective assignment. Critically, note that the functions that we create are not monotone. Hence, the problem is W[1]-hard, without the monotonicity condition. The necessity for monotonicity is given by Fomin et al. [31], who showed that for arity 4 and when a requirement stronger than monotonicity is imposed, the problem is W[1]-hard.

If the function f of a constraint $c = [x_i \,\mathsf{sign}\, f(x_j)]$ is constantly β (that is, for every $t \in \{0, \ldots, N\}$, $f(t) = \beta$), then we use the shorthand $c = [x_i \,\mathsf{sign}\, \beta]$. Moreover, we suppose that every constraint represented by a quadruple is associated with two distinct variables.

Let (X, C, N) be an instance of Monotone 2-CSP. We create a 2-CNF-SAT formula $\mathcal{C}$ as follows (we only describe its variables and clauses). For each $x \in X$ and $d \in \{0, 1, \ldots, N, N+$

$1\}$, we create a variable $x[d]$. (Setting $x[d] = 1$ will be interpreted as $x \geq d$.) We now describe the clauses that we create.

Ensuring Valid Assignments for Variables. We need to ensure that x is assigned some value from $\{0, 1, \ldots, N\}$. Thus, for each $x \in X$, $x \geq 0$ should always be satisfied. To ensure this, we add the clause $(x[0])$ to $\mathcal{C}$, for every $x \in X$. Similarly, to ensure that $x \leq N$, we add the clause $(\neg x[N + 1])$ to $\mathcal{C}$, for each $x \in X$.

Encoding Order Implications. For each $x \in X$ and $d \in \{1, 2, \ldots, N, N + 1\}$, we add the clause $(x[d] \to x[d - 1])$ to $\mathcal{C}$. (The above clauses ensure that if $x \geq d$, then $x \geq d - 1$ also holds.)

Encoding constant functions. Consider a constraint of the form $c = [x \leq \beta]$, where $\beta \in \{0, 1, \ldots, N\}$. We add the clause $(\neg x[\beta + 1])$ to $\mathcal{C}$. Next consider a constraint of the form $c = [x \geq \beta]$, where $\beta \in \{0, 1, \ldots, N\}$. (We can safely assume that $\beta < N + 1$, otherwise we can correctly report that the instance is a no-instance.) We add the clause $(x[\beta])$ to $\mathcal{C}$.

Encoding non-constant functions. We encode $c = [x_i \,\mathsf{sign}\, f(x_j)] \in C$ based on different cases of $\mathsf{sign} \in \{\leq, \geq\}$ and whether f is non-increasing or non-decreasing.

1. $\mathsf{sign} = \geq$ and f is non-decreasing. For each $d \in \{0, 1, \ldots, N\}$, we add the clause $(x_j[d] \to x_i[f(d)])$.
2. $\mathsf{sign} = \geq$ and f is non-increasing. For each $d \in \{0, 1, \ldots, N\}$, we add the clause $(\neg x_j[d + 1] \to x_i[f(d)])$.
3. $\mathsf{sign} = \leq$ and f is non-decreasing. For each $d \in \{0, 1, \ldots, N\}$, we add the clause $(\neg x_j[d] \to \neg x_i[f(d) + 1])$.
4. $\mathsf{sign} = \leq$ f is non-increasing. For each $d \in \{0, 1, \ldots, N\}$, we add the clause $(x_j[d] \to \neg x_i[f(d) + 1])$.

In the following lemma we prove the correctness of our reduction.

► **Lemma C.1.** *(X, C, N) is a yes-instance of* Monotone 2-CSP *if and only if $\mathcal{C}$ is a yes-instance of* 2-SAT.

Proof. Let Z be the set of variables of $\mathcal{C}$. For one direction assume that (X, C, N) is a yes-instance of Monotone 2-CSP, and let $\alpha : X \to \{0, 1, \ldots, N\}$ be its solution. We construct an assignment $\varphi : Z \to \{0, 1\}$ as follows. Consider $x \in X$ and $d \in \{0, 1, \ldots, N, N + 1\}$. If $\alpha(x) \leq d$, then we set $\varphi(x[d]) = 1$, otherwise, we set $\varphi(x[d]) = 0$. We will show that φ is a satisfying assignment for $\mathcal{C}$. For $x \in X$ as $\alpha(x) \in \{0, 1, \ldots, N\}$, the clauses $(x[0])$ and $(\neg x[N + 1])$ are satisfied, thus the clauses ensuring valid assignments for variables and clauses for order implications are satisfied. Consider $\beta \in \{0, 1, \ldots, N\}$ and $x \in X$. If $[x \leq \beta] \in C$, then by the construction of φ, the clause $(\neg x[\beta + 1]) \in \mathcal{C}$ is satisfied. Similarly, if $[x \geq \beta] \in C$, then the clause $(x[\beta]) \in \mathcal{C}$ is satisfied. Thus, all the clauses encoding constant functions are satisfied. Now consider a constraint $c = [x_i \,\mathsf{sign}\, f(x_j)] \in C$, and consider the following cases based on $\mathsf{sign} \in \{\leq, \geq\}$ and whether f is non-decreasing or non-decreasing.

1. If $\mathsf{sign} = \geq$ and f is non-decreasing, then for each $d \in \{0, 1, \ldots, N\}$, we have the clause $(x_j[d] \to x_i[f(d)])$ in $\mathcal{C}$. We show that all the above clauses are satisfied by φ. Consider some $d \in \{0, 1, \ldots, N\}$. If $d > \alpha(x_j)$, then $\varphi(x_j[d]) = 0$, and thus $(x_j[d] \to x_i[f(d)])$ is satisfied. Now consider the case when $d \leq \alpha(x_j)$. As α is a solution for the instance (X, C, N), we have $f(\alpha(x_j)) \leq \alpha(x_i)$. As f is non-decreasing, we have $f(d) \leq f(\alpha(x_j)) \leq \alpha(x_i)$. Thus we can conclude that $(x_j[d] \to x_i[f(d)])$ is satisfied by φ.
2. If $\mathsf{sign} = \geq$ and f is non-increasing, then for each $d \in \{0, 1, \ldots, N\}$, we have $(\neg x_j[d+1] \to x_i[f(d)]) \in \mathcal{C}$. Consider some $d \in \{0, 1, \ldots, N\}$. If $d < \alpha(x_j)$, then $\varphi(x_j[d + 1]) = 1$, and

thus $(\neg x_j[d+1] \to x_i[f(d)])$ is satisfied. Now consider the case when $d \geq \alpha(x_j)$, and $\varphi(x_j[d+1]) = 0$. As α is a solution for the instance (X, C, N), we have $f(\alpha(x_j)) \leq \alpha(x_i)$. As f is non-increasing, we have $f(d) \leq f(\alpha(x_j)) \leq \alpha(x_i)$. Thus we can conclude that $(\neg x_j[d+1] \to x_i[f(d)])$ is satisfied by φ.

3. If $\mathsf{sign} = \leq$ and f is non-decreasing, then for each $d \in \{0, 1, \ldots, N\}$, we have $(\neg x_j[d] \to \neg x_i[f(d)+1]) \in \mathcal{C}$. Consider some $d \in \{0, 1, \ldots, N\}$. If $d \leq \alpha(x_j)$, then $\varphi(x_j[d]) = 1$, and thus $(\neg x_j[d] \to \neg x_i[f(d)+1])$ is satisfied by φ. Now consider the case when $d > \alpha(x_j)$, and $\varphi(x_j[d]) = 0$. As α is a solution for the instance (X, C, N) and f is non-decreasing, we have $\alpha(x_i) \leq f(\alpha(x_j)) \leq f(d)$. Thus, $\varphi(x_i[f(d)+1]) = 0$, and we can conclude that $(\neg x_j[d] \to \neg x_i[f(d)+1])$ is satisfied by φ.
4. If $\mathsf{sign} = \leq$ f is non-increasing, for each $d \in \{0, 1, \ldots, N\}$, we have $(x_j[d] \to \neg x_i[f(d)+1]) \in \mathcal{C}$. Consider some $d \in \{0, 1, \ldots, N\}$. If $d > \alpha(x_j)$, then $\varphi(x_j[d]) = 0$, and thus $(x_j[d] \to \neg x_i[f(d)+1])$ is satisfied. Now consider the case when $d \leq \alpha(x_j)$, and $\varphi(x_j[d]) = 1$. As α is a solution for the instance (X, C, N) and f is non-increasing, we have $\alpha(x_i) \leq f(\alpha(x_j)) \leq f(d)$. Thus, $\varphi(x_i[f(d)+1]) = 0$, and we can conclude that $(x_j[d] \to \neg x_i[f(d)+1])$ is satisfied by φ.

The above discussions cover all clauses in $\mathcal{C}$, thus we can conclude that $\mathcal{C}$ is a yes-instance of 2-SAT.

For the other direction, let $\mathcal{C}$ be a yes-instance of 2-SAT, and let $\varphi : Z \to \{0,1\}$ be its solution. From the clauses for encoding valid assignments and order implications, for each $x \in X$, there is $d_x \in \{0, 1, \ldots, N\}$, such that for all $d \in \{0, 1, \ldots, d_x\}$, we have $x[d] = 1$ and for any $d' \in \{d+1, d+2, \ldots, N, N+1\}$, we have $x[d] = 0$. We construct $\alpha : X \to \{0, 1, \ldots, N\}$, by setting $\alpha(x) = d_x$, where $x \in X$. We argue that α is a solution for the instance (X, C, N). Consider a clause of the form $[x \leq \beta] \in C$, where $\beta \in \{0, 1, \ldots, N\}$. As the clause $(\neg x[\beta+1]) \in \mathcal{C}$ is satisfied by φ, we have $\alpha(x) = d_x \leq \beta$. Thus, $[x \leq \beta] \in C$ is satisfied by α. Next, consider a clause of the form $[x \geq \beta] \in C$, for some $\beta \in \{0, 1, \ldots, N\}$. As $(x[\beta]) \in \mathcal{C}$ is satisfied by φ, we have $\alpha(x) = d_x \geq \beta$. Now consider a constraint $c = [x_i \,\mathsf{sign}\, f(x_j)] \in C$, and consider the following cases based on $\mathsf{sign} \in \{\leq, \geq\}$ and whether f is non-decreasing or non-increasing.

1. If $\mathsf{sign} = \geq$ and f is non-decreasing, then for each $d \in \{0, 1, \ldots, N\}$, we have the clause $(x_j[d] \to x_i[f(d)])$ in $\mathcal{C}$. Note that we have $\varphi(x_j[d_{x_j}]) = 1$ and hence, $\varphi(x[f(d_{x_j})]) = 1$. Thus, $d_{x_i} \geq f(d_{x_j})$. Hence we can conclude that $\alpha(x_j) = d_{x_i} \geq f(\alpha(x_j))$.
2. If $\mathsf{sign} = \geq$ and f is non-increasing, then for each $d \in \{0, 1, \ldots, N\}$, we have $(\neg x_j[d+1] \to x_i[f(d)]) \in \mathcal{C}$. Note that $\varphi(x_j[d_{x_j}+1]) = 0$. Thus $\alpha(x_i) = d_x \geq f(\alpha(x_j))$.
3. If $\mathsf{sign} = \leq$ and f is non-decreasing, then for each $d \in \{0, 1, \ldots, N\}$, we have $(\neg x_j[d] \to \neg x_i[f(d)+1]) \in \mathcal{C}$. As $\varphi(x_j[d_{x_j}+1]) = 0$, we must have $d_x \leq f(d_{x_j})$. Thus, $\alpha(x_i) = d_x \leq f(\alpha(x_j))$.
4. If $\mathsf{sign} = \leq$ f is non-increasing, for each $d \in \{0, 1, \ldots, N\}$, we have $(x_j[d] \to \neg x_i[f(d)+1]) \in \mathcal{C}$. As $\varphi(x_j[d_{x_j}]) = 1$, we must have $d_x \leq f(d_{x_j})$. Thus, we have $\alpha(x_i) = d_x \leq f(\alpha(x_j))$.

Thus, we can conclude that (X, C, N) is a yes-instance of Monotone 2-CSP. ◀

2-SAT admits an algorithm running in time $\mathcal{O}(n+m)$, where n is the number of variables and m is the number of clauses [4]. This together with the construction of the 2-SAT instance $\mathcal{C}$ for the given instance (X, C, N) of Monotone 2-CSP and Lemma C.1, implies Theorem 2.

D Proofs Omitted from Section 3.1

D.1 Proof of Lemma 3.1

Suppose that v sees some vertex in $[i,j]$, else the proof is trivial. Denote $\ell = \mathsf{first}(v,[i,j])$ and $h = \mathsf{last}(v,[i,j])$. We consider two cases. First, suppose that $v \notin [i,j]$. Define a polygon $Q = (V_Q, E_Q)$ by $V_Q = \{\ell, \ell+1, \ldots, h\} \cup \{v\}$ and $E_Q = \{\{t, t+1\} : t \in \{\ell, \ldots, h-1\}\} \cup \{\{\ell, v\}, \{h, v\}\}$. Clearly, Q is simple. Since $[i,j]$ is a convex region of P, we have that Q is a simple polygon such that the interior angle at t in Q, for any $t \in \{\ell, \ell+1, \ldots, h\}$, is at most 180 degrees. Thus, the only vertex in Q that can be a reflex vertex is v. Moreover, since P contains both $\overline{\ell v}$ and $\overline{hv}$, we have that Q is contained in P. By Observation 2.1 and Proposition 1.1, this means that for all $t \in \{\ell, \ldots, h\}$, v sees t.

Second, suppose that $v \in [i,j]$. Define a polygon $Q = (V_Q, E_Q)$ by $V_Q = \{\ell, \ell+1, \ldots, v\}$ and $E_Q = \{\{t, t+1\} : t \in \{\ell, \ldots, v-1\}\} \cup \{\{\ell, v\}\}$ and a polygon $Q' = (V'_Q, E'_Q)$ by $V'_Q = \{v, v+1, \ldots, h\}$ and $E'_Q = \{\{t, t+1\} : t \in \{v, \ldots, h-1\}\} \cup \{\{h, v\}\}$. Clearly, both polygons are simple and convex. Moreover, since P contains both $\overline{\ell v}$ and $\overline{hv}$, we have that both Q and Q' are contained in P. By Observation 2.1, this means that for all $t \in \{\ell, \ldots, h\}$, v sees t. ◀

D.2 Proof of Lemma 3.2

We only prove the first statement in Lemma 3.2. (The proof of the second statement is symmetric.) To this end, we first analyze how a convex region sees itself, and afterwards we analyze how one convex region sees a different convex region. Having completed this analysis, we present the proof of the lemma.

Interaction within the same region. First, we analyze how a convex region sees itself.

▶ **Lemma D.1.** *Let $P = (V, E)$ be a simple polygon. Let $[i,j]$ a convex region of P. Let $\ell, h \in [i,j]$ be two vertices that see each other, where $\ell \leq h$. For all $x, y \in \{\ell, \ell+1, \ldots, h\}$, $x \leq y$, the vertices x and y see each other.*

Proof. Define the polygon $Q = (V_Q, E_Q)$ by $V_Q = \{\ell, \ell+1, \ldots, h\}$ and $E_Q = \{\{t, t+1\} : t \in \{\ell, \ldots, h-1\}\} \cup \{\{\ell, h\}\}$. Since $[i,j]$ is a convex region of P and the line segment $\overline{\ell h}$ is contained in P, we have that Q is a convex polygon that is contained in P. By Observation 2.1, this means that any two vertices of Q see each other. ◀

We utilize Lemma D.1 in order to prove the following result.

▶ **Lemma D.2.** *Let $P = (V, E)$ be a simple polygon. Let $[i,j]$ be a convex region of P. Let ℓ and h be two vertices in $[i,j]$ such that $\ell \leq h$, $x = \mathsf{first}(\ell,[i,j]) \neq \mathsf{nil}$, and $y = \mathsf{first}(h,[i,j]) \neq \mathsf{nil}$. Then, for all $t \in \{\ell, \ell+1, \ldots, h\}$, $\min\{x,y\} \leq \mathsf{first}(t,[i,j]) \leq \max\{x,y\}$.*

Proof. Suppose that $\ell < h-1$, else the proof is complete. Let $t \in \{\ell+1, \ldots, h-1\}$. Suppose, by way of contradiction, that either $\mathsf{first}(t,[i,j]) < \min\{x,y\}$ or $\max\{x,y\} < \mathsf{first}(t,[i,j])$. First, assume that $\mathsf{first}(t,[i,j]) < \min\{x,y\}$. Because every vertex sees itself, we have that $\min\{x,y\} \leq \ell$. Thus, $\mathsf{first}(t,[i,j]) < \ell < t$. By Lemma D.1, this implies that ℓ sees $\mathsf{first}(t,[i,j])$. However, this is contradiction because $x = \mathsf{first}(\ell,[i,j])$ while $\mathsf{first}(t,[i,j]) < x$. Second, assume that $\max\{x,y\} < \mathsf{first}(t,[i,j])$. Since every vertex sees itself, we have that $\mathsf{first}(t,[i,j]) \leq t$, and hence $\max\{x,y\} < t$. In particular, $y < t < h$. By Lemma D.1, this implies that t sees y. However, this is contradiction because $y < \mathsf{first}(t,[i,j])$. ◀

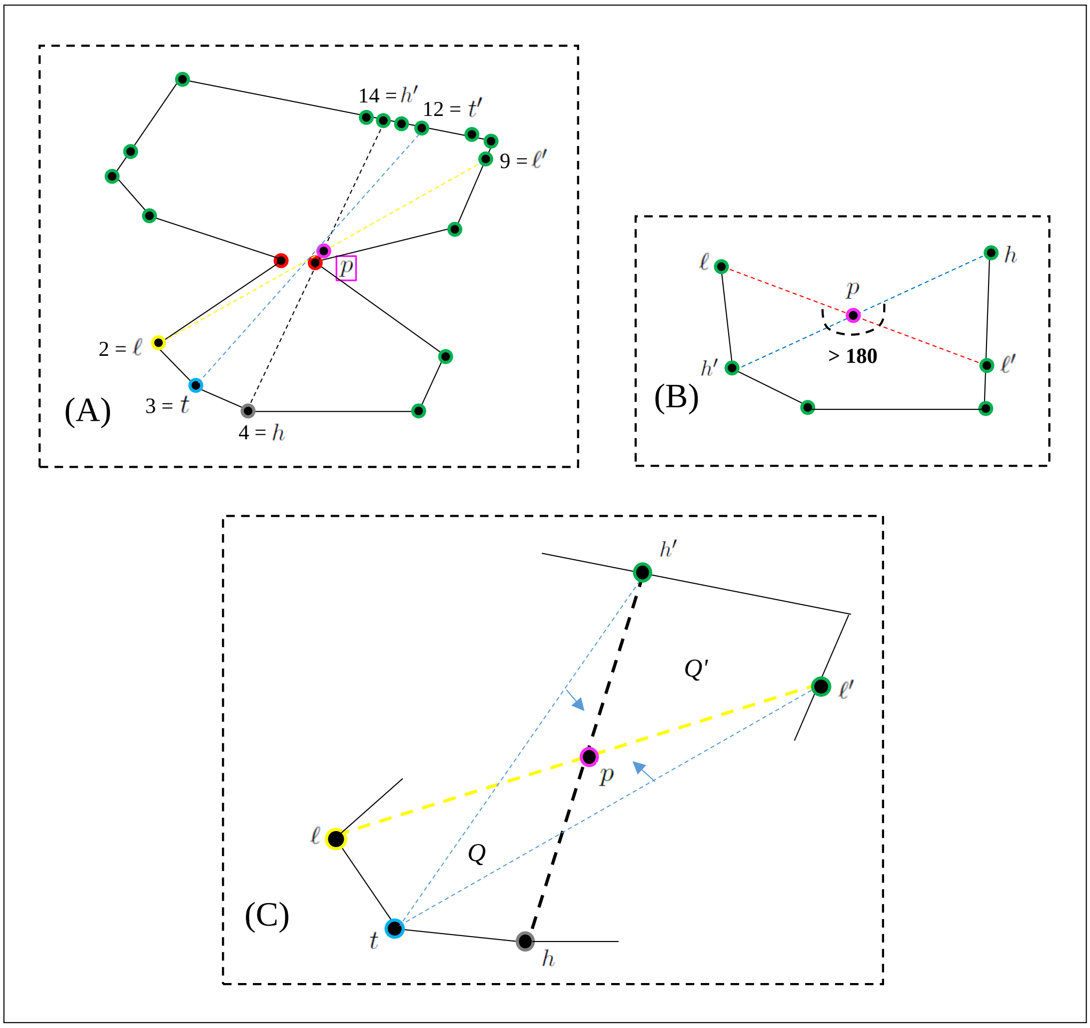

Figure 9 (A) The vertices $\ell, \ell', h, h', t, t'$ and p in the proof of Lemma D.5. The polygon is the same as the one in Fig. 4. (B) A contradiction in the proof of Lemma D.5: the vertices ℓ' and h' belong to the same convex region as the vertices ℓ and h. (C) The line segment $\overline{tt'}$ must intersect both $\overline{\ell\ell'}$ and $\overline{hh'}$.

Interaction between two distinct regions. Second, we analyze how one convex region sees a different convex region. For this purpose, we first argue that certain line segments intersect. Then, we consider the case where they intersect in a single point, and the case where they intersect in more than a single point.

▶ **Lemma D.3.** *Let $P = (V, E)$ be a simple polygon. Let $[i, j]$ and $[i', j']$ be distinct maximal convex regions of P. Let ℓ and h be vertices in $[i, j]$ such that $\ell \leq h$, $\ell' = \mathsf{first}(\ell, [i', j']) \neq \mathsf{nil}$ and $h' = \mathsf{first}(h, [i', j']) \neq \mathsf{nil}$. Then, the line segments $\overline{\ell\ell'}$ and $\overline{hh'}$ intersect.*

Proof. Suppose, by way of contradiction, that $\overline{\ell\ell'}$ and $\overline{hh'}$ do not intersect. Then, $\ell \neq h$ and $\ell' \neq h'$. Define a polygon $Q = (V_Q, E_Q)$ by $V_Q = \{\ell, \ell+1, \ldots, h\} \cup \{\min(\ell', h'), \ldots, \max(\ell', h')\}$ and $E_Q = \{\{t, t+1\} : t \in \{\ell, \ldots, h-1\}\} \cup \{\{t', t'+1\} : t' \in \{\min(\ell', h'), \ldots, \max(\ell', h') - 1\}\} \cup \{\{\ell, \ell'\}, \{h, h'\}\}$. For any vertex $v \in V_Q \setminus \{\ell, h, \ell', h'\}$, the interior angle at v is the same in Q and P. Moreover, for each any $v \in \{\ell, h, \ell', h'\}$, because $\overline{\ell\ell'}$ and $\overline{hh'}$ are contained in P, the interior angle at v in Q is at most the interior angle at v in P. Thus, since $[i,j]$ and $[i',j']$ are convex region of P, we have that any interior angle of Q is at most 180 degrees. Moreover, because the line segments $\overline{\ell\ell'}$ and $\overline{hh'}$ do not intersect, we have that Q is simple. Thus, Q is a convex polygon contained in P. By Observation 2.1, h sees ℓ' in Q, and ℓ sees h' in Q. In turn, this implies that h sees ℓ' in P, and ℓ sees h' in P. If $\ell' < h'$, then $\ell' < h' = \mathsf{first}(h,[i',j'])$, which is a contradiction. Hence, $\ell' > h'$. However, then $h' < \ell' = \mathsf{first}(\ell,[i',j'])$, which is a contradiction. ◀

Now, we analyze the case where the intersection consists of a single point.

▶ **Lemma D.4.** *Let $P = (V,E)$ be a simple polygon. Let $[i,j]$ and $[i',j']$ be two distinct maximal convex regions of P. Let ℓ and h be two vertices in $[i,j]$ such that $\ell \leq h$, $\ell' = \mathsf{first}(\ell,[i',j']) \neq \mathsf{nil}$, $h' = \mathsf{first}(h,[i',j']) \neq \mathsf{nil}$ and the line segments $\overline{\ell\ell'}$ and $\overline{hh'}$ intersect at a single point. Then, for all $t \in \{\ell,\ell+1,\ldots,h\}$, either $\mathsf{first}(t,[i',j']) = \mathsf{nil}$ or $\min\{\ell',h'\} \leq \mathsf{first}(t,[i',j']) \leq \max\{\ell',h'\}$.*

Proof. Suppose that $\ell < h-1$, else the proof is complete. Let p denote the unique point where $\overline{\ell\ell'}$ and $\overline{hh'}$ intersect. Define two polygons as follows (see Fig. 9(A)).

- The first polygon $Q = (V_Q, E_Q)$ is given by $V_Q = \{\ell,\ldots,h\} \cup \{p\}$ and $E_Q = \{\{t,t+1\} : t \in \{\ell,\ldots,h-1\}\} \cup \{\{h,p\},\{p,\ell\}\}$.
- The second polygon $Q' = (V_{Q'}, E_{Q'})$ is given by $V_{Q'} = \{\min(\ell',h'),\ldots,\max(\ell',h')\} \cup \{p\}$ and $E_{Q'} = \{\{t',t'+1\} : t' \in \{\min(\ell',h'),\ldots,\max(\ell',h')-1\}\} \cup \{\{h',p\},\{p,\ell'\}\}$.

We claim that Q and Q' are convex polygons contained in P. We prove this claim only for Q since the proof for Q' is symmetric. First, since $[i,j]$ is a convex region of P, and the line segments $\overline{\ell p}$ and $\overline{hp}$ intersect only at p and are contained in P, we have that Q is a simple polygon that is contained in P. Moreover, every interior angle at t in Q, for all $t \in \{\ell,\ell+1,\ldots,h\}$, is at most the interior angle at t in P, and hence it is at most 180 degrees. Now, consider the interior angle at p in Q. If this angle were larger than 180 degrees, then ℓ' and h' would have belonged to $[i,j]$ (see Fig. 9(B)), which yields a contradiction since $[i,j]$ and $[i',j']$ are distinct maximal convex regions of P. Thus, Q is convex.

Towards the proof that for all $t \in \{\ell+1,\ldots,h-1\}$, either $\mathsf{first}(t,[i',j']) = \mathsf{nil}$ or $\min\{\ell',h'\} \leq \mathsf{first}(t,[i',j']) \leq \max\{\ell',h'\}$, choose some $t \in \{\ell+1,\ldots,h-1\}$, and denote $t' = \mathsf{first}(t,[i',j'])$. If $t' = \mathsf{nil}$, then we are done. Thus, suppose that $t' \neq \mathsf{nil}$. By Lemma D.3, the line segment $\overline{tt'}$ intersects both $\overline{\ell\ell'}$ and $\overline{hh'}$. Since the polygons Q and Q' are convex, and because t belongs to Q, this is only possible if t' belongs to the boundary of Q' that coincides with the convex region $[i',j']$ of P (see Fig. 9(C)). From this, we conclude that $\min\{\ell',h'\} \leq t' \leq \max\{\ell',h'\}$. ◀

Secondly, we analyze the case where the intersection consists of more than a single point.

▶ **Lemma D.5.** *Let $P = (V,E)$ be a simple polygon. Let $[i,j]$ and $[i',j']$ be two distinct maximal convex regions of P. Let ℓ and h be two vertices in $[i,j]$ such that $\ell \leq h$, $\ell' = \mathsf{first}(\ell,[i',j']) \neq \mathsf{nil}$, $h' = \mathsf{first}(h,[i',j']) \neq \mathsf{nil}$ and the line segments $\overline{\ell\ell'}$ and $\overline{hh'}$ intersect at more than one point. Then, for all $t \in \{\ell,\ell+1,\ldots,h\}$, $\min\{\ell',h'\} = \mathsf{first}(t,[i',j']) = \max\{\ell',h'\}$.*

Proof. Since $[i,j]$ is a convex region of P, and because $\overline{\ell\ell'}$ and $\overline{hh'}$ intersect at more than one point, we have that the interior angle at t in P, for all $t \in \{\ell+1,\ldots,h-1\}$, is exactly 180 degrees (see Fig. 10(A)). Then, ℓ sees h' and h sees ℓ', which implies that $\ell' = h'$. Thus, one of the line segments $\overline{\ell h'}$ and $\overline{hh'}$ is a subsegment of the other. Without loss of generality,

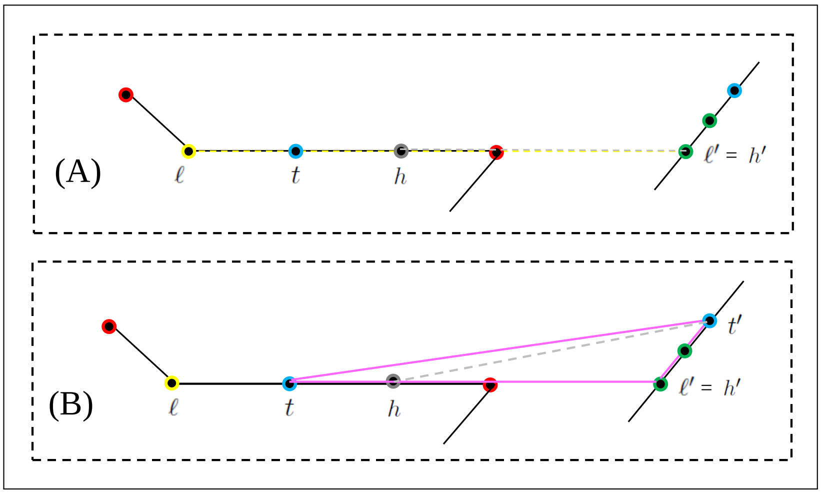

Figure 10 (A) The vertices ℓ, ℓ', h, h' and t in the proof of Lemma D.5. (B) The polygon defined in the proof of Lemma D.5.

suppose that $\overline{hh'}$ is a subsegment of $\overline{\ell h'}$, and that $\overline{\ell h'}$ and $\overline{hh'}$ are parallel to the x axis. Note that this means that the interior angle at h in P is also 180 degrees.

Suppose that $\ell < h-1$, else the proof is complete. Let $t \in \{\ell+1, \ldots, h-1\}$, and denote $t' = \mathsf{first}(t, [i', j'])$. We need to prove that $t' = h'$. Suppose, by way of contradiction, that $t' \neq h'$. Because t sees h', this means that $t' < h'$. Observe that t sees h', and t does not see any vertex in $[i', j']$ whose y-coordinate is lower than the y-coordinate of h'. Thus, the y-coordinate of t' is larger than the one of t. Then, the polygon defined by $\overline{t't}, \overline{th}, \overline{hh'}$ and $\overline{q(q+1)}$ for all $q \in \{t', \ldots, h'-1\}$ is convex and contained in P (see Fig. 10(B)). However, by Observation 2.1, this means that h sees t', and hence $\mathsf{first}(h, [i', j'])$ cannot be equal to h' (because $t' < h'$). We have thus reached a contradiction, which concludes the proof. ◀

From Lemmas D.3, D.4 and D.5, we derive the following result.

▶ **Lemma D.6.** *Let $P = (V, E)$ be a simple polygon. Let $[i, j]$ and $[i', j']$ be two distinct maximal convex regions of P. Let ℓ and h be two vertices in $[i, j]$ such that $\ell \leq h$, $\ell' = \mathsf{first}(\ell, [i', j']) \neq \mathsf{nil}$, $h' = \mathsf{first}(h, [i', j']) \neq \mathsf{nil}$. Then, for all $t \in \{\ell, \ell+1, \ldots, h\}$, either $\mathsf{first}(t, [i', j']) = \mathsf{nil}$ or $\min\{\ell', h'\} \leq \mathsf{first}(t, [i', j']) \leq \max\{\ell', h'\}$.*

Proof of the first statement of Lemma 3.2. Suppose, by way of contradiction, that the way in which $[i, j]$ views $[i', j]$ with respect to first is neither non-decreasing nor non-increasing. Then, there exist $x, y, z \in \{i, i+1, \ldots, j\}$ such that $x < y < z$, $\mathsf{first}(x, [i', j']) \neq \mathsf{nil}$, $\mathsf{first}(z, [i', j']) \neq \mathsf{nil}$, and

1. $\max\{\mathsf{first}(x, [i', j']), \mathsf{first}(z, [i', j'])\} < \mathsf{first}(y, [i', j'])$, or
2. $\min\{\mathsf{first}(x, [i', j']), \mathsf{first}(z, [i', j'])\} > \mathsf{first}(y, [i', j'])$, or
3. $\mathsf{first}(x, [i', j']) = \mathsf{first}(z, [i', j'])$ and $\mathsf{first}(y, [i', j']) = \mathsf{nil}$.

If $\mathsf{first}(x, [i', j']) = \mathsf{first}(z, [i', j'])$, then by Lemma 3.1, $\mathsf{first}(x, [i', j'])$ sees t for all $t \in \{x, x+1, \ldots, z\}$. Thus, the third condition cannot be satisfied. If $[i, j] \neq [i', j']$, then Lemma D.6 implies that neither of the first two conditions can be satisfied. Otherwise, if $[i, j] = [i', j']$, then Lemma D.2 implies that neither of the first two conditions can be satisfied. Thus, we necessarily reach a contradiction. ◀

E The Two Sets of Four Cases Omitted from Section 3.3

In this section, we present the full specification of the two sets of four cases that are part of the definition of constraints to guard the middle vertices of convex regions.[12] We remind that to unify notation, in case e (resp. e') is a reflex vertex, we say that the way e (resp. e') views C is non-decreasing with respect to both first and last. Here, Lemma 3.2 ensures that at least one case in each set is satisfied. We start with the first set of four cases.

1. The way e' views C is non-decreasing with respect to last, and the way e views C is non-decreasing with respect to first. We insert a constraint $[x \leq f(x')]$, where $f : \{0,\ldots,N\} \rightarrow \{0,\ldots,N\}$ is defined as follows.
   - For all $i < \ell'$: $f(i) = 0$.
   - For $i = \ell', \ell'+1, \ldots, h'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
     - If **(i)** $a = \mathsf{nil}$, **(ii)** $a+1 \notin C$, or **(iii)** $\mathsf{first}(j, C) \leq a+1$ for no $j \in e$, then $f(i) = f(i-1)$.
     - Otherwise, let j be the largest vertex in e such that $\mathsf{first}(j, C) \leq a+1$, and define $f(i) = j$.
   - For all $i > h'$: $f(i) = N$.

   Monotonicity. We claim that f is monotonically non-decreasing. To show this, we choose some $i \in \{1,\ldots,N\}$. If $i \leq \ell'$ or $i > h'$, then it is clear that $f(i) \geq f(i-1)$. Now, suppose that $\ell' < i \leq h'$. If $a = \mathsf{nil}$, $a+1 \notin C$ or $\mathsf{first}(j, C) \leq a+1$ for no $j \in e$, then it is clear that $f(i) \geq f(i-1)$. Hence, we next suppose that this is not the case. Then, j is well-defined. To prove that $f(i) \geq f(i-1)$, we need to show that $f(i-1) \leq j$. Let $\widehat{i}$ be the largest vertex in $\{\ell', \ldots, i-1\}$ such that $\widehat{a} = \mathsf{last}(\widehat{i}, C) \neq \mathsf{nil}$, $\widehat{a}+1 \in C$, and there is a vertex $\widehat{j} \in e$ such that $\mathsf{first}(\widehat{j}, C) \leq \widehat{a}+1$. (If such a vertex does not exist, then $f(i-1) = 0$, and we are done.) Denote $\widehat{j} = f(\widehat{i})$. Note that it suffices to show that $j \geq \widehat{j}$. Because the way e' views C is non-decreasing with respect to last, we have that $\widehat{a} \leq a$. Then, because the way e views C is non-decreasing with respect to first, we have that $j \geq \widehat{j}$.
2. The way e' views C is non-decreasing with respect to last, and the way e views C is non-increasing with respect to first. We insert a constraint $[x \geq f(x')]$, where $f : \{0,\ldots,N\} \rightarrow \{0,\ldots,N\}$ is defined as follows.
   - For all $i < \ell'$: $f(i) = N$.
   - For $i = \ell', \ell'+1, \ldots, h'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
     - If **(i)** $a = \mathsf{nil}$, **(ii)** $a+1 \notin C$, or **(iii)** $\mathsf{first}(j, C) \leq a+1$ for no $j \in e$, then $f(i) = f(i-1)$.
     - Otherwise, let j be the smallest vertex in e such that $\mathsf{first}(j, C) \leq a+1$, and define $f(i) = j$.
   - For all $i > h'$: $f(i) = 0$.

   Monotonicity. We claim that f is monotonically non-increasing. To show this, we choose some $i \in \{1,\ldots,N\}$. If $i \leq \ell'$ or $i > h'$, then it is clear that $f(i) \leq f(i-1)$. Now, suppose that $\ell' < i \leq h'$. If $a = \mathsf{nil}$, $a+1 \notin C$ or $\mathsf{first}(j, C) \leq a+1$ for no $j \in e$, then it is clear that $f(i) \leq f(i-1)$. Hence ,we next suppose that this is not the case. Then, j is well-defined. To prove that $f(i) \leq f(i-1)$, we need to show that $f(i-1) \geq j$. Let $\widehat{i}$ be the largest vertex in $\{\ell', \ldots, i-1\}$ such that $\widehat{a} = \mathsf{last}(\widehat{i}, C) \neq \mathsf{nil}$, $\widehat{a}+1 \in C$, and there is a vertex $\widehat{j} \in e$ such that $\mathsf{first}(\widehat{j}, C) \leq \widehat{a}+1$. (If such a vertex does not exist, then $f(i-1) = N$, and we are done.) Denote $\widehat{j} = f(\widehat{i})$. Note that it suffices to show that

[12] We remark that we do not know whether all of these cases can be realized geometrically.

$j \leq \widehat{j}$. Because the way e' views C is non-decreasing with respect to last, we have that $\widehat{a} \leq a$. Then, because the way e views C is non-increasing with respect to first, we have that $j \leq \widehat{j}$.

3. The way e' views C is non-increasing with respect to last, and the way e views C is non-decreasing with respect to first. We insert a constraint $[x \leq f(x')]$, where $f : \{0,\dots,N\} \to \{0,\dots,N\}$ is defined as follows.
- For all $i > h'$: $f(i) = 0$.[13]
- For $i = h', h'-1, \dots, \ell'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
  - If **(i)** $a = \mathsf{nil}$, **(ii)** $a+1 \notin C$, or **(iii)** $\mathsf{first}(j, C) \leq a+1$ for no $j \in e$, then $f(i) = f(i+1)$.
  - Otherwise, let j be the largest vertex in e such that $\mathsf{first}(j, C) \leq a+1$, and define $f(i) = j$.
- For all $i < \ell'$: $f(i) = N$.

Monotonicity. We claim that f is monotonically non-increasing. To show this, we choose some $i \in \{0,\dots,N-1\}$. If $i < \ell'$ or $i \geq h'$, then it is clear that $f(i) \geq f(i+1)$. Now, suppose that $\ell' \leq i < h'$. If $a = \mathsf{nil}$, $a+1 \notin C$ or $\mathsf{first}(j, C) \leq a+1$ for no $j \in e$, then it is clear that $f(i) \geq f(i+1)$. Hence, we next suppose that this is not the case. Then, j is well-defined. To prove that $f(i) \geq f(i+1)$, we need to show that $j \geq f(i+1)$. Let $\widehat{i}$ be the smallest vertex in $\{i+1,\dots,h'\}$ such that $\widehat{a} = \mathsf{last}(\widehat{i}, C) \neq \mathsf{nil}$, $\widehat{a}+1 \in C$, and there is a vertex $\widehat{j} \in e$ such that $\mathsf{first}(\widehat{j}, C) \leq \widehat{a}+1$. (If such a vertex does not exist, then $f(i+1) = 0$, and we are done.) Denote $\widehat{j} = f(\widehat{i})$. Note that it suffices to show that $\widehat{j} \leq j$. Because the way e' views C is non-increasing with respect to last, we have that $a \geq \widehat{a}$. Then, because the way e views C is non-decreasing with respect to first, we have that $\widehat{j} \leq j$.

4. The way e' views C is non-increasing with respect to last, and the way e views C is non-increasing with respect to first. We insert a constraint $[x \geq f(x')]$, where $f : \{0,\dots,N\} \to \{0,\dots,N\}$ is defined as follows.
- For all $i > h'$: $f(i) = N$.
- For $i = h', h'-1, \dots, \ell'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
  - If **(i)** $a = \mathsf{nil}$, **(ii)** $a+1 \notin C$, or **(iii)** $\mathsf{first}(j, C) \leq a+1$ for no $j \in e$, then $f(i) = f(i+1)$.
  - Otherwise, let j be the smallest vertex in e such that $\mathsf{first}(j, C) \leq a+1$, and define $f(i) = j$.
- For all $i < \ell'$: $f(i) = 0$.

Monotonicity. We claim that f is monotonically non-decreasing. To show this, we choose some $i \in \{0,\dots,N-1\}$. If $i \geq h'$ or $i < \ell'$, then it is clear that $f(i+1) \geq f(i)$. Now, suppose that $\ell' \leq i < h'$. If $a = \mathsf{nil}$, $a+1 \notin C$ or $\mathsf{first}(j, C) \leq a+1$ for no $j \in e$, then it is clear that $f(i+1) \geq f(i)$. Hence, we next suppose that this is not the case. Then, j is well-defined. To prove that $f(i+1) \geq f(i)$, we need to show that $j \leq f(i-1)$. Let $\widehat{i}$ be the smallest vertex in $\{i+1,\dots,h'\}$ such that $\widehat{a} = \mathsf{last}(\widehat{i}, C) \neq \mathsf{nil}$, $\widehat{a}+1 \in C$, and there is a vertex $\widehat{j} \in e$ such that $\mathsf{first}(\widehat{j}, C) \leq \widehat{a}+1$. (If such a vertex does not exist, then $f(i+1) = N$, and we are done.) Denote $\widehat{j} = f(\widehat{i})$. Note that it suffices to show that $\widehat{j} \geq j$. Because the way e' views C is non-increasing with respect to last, we have that

[13] In the third and fourth cases, unlike the first and second cases, we first define f for integers $i > h'$ rather than for integers $i < \ell'$. The correctness of the reduction relies on this choice of design (we further elaborate on this in footnote 17 in the proof).

$a \geq \widehat{a}$. Then, because the way e views C is non-increasing with respect to first, we have that $\widehat{j} \geq j$.

Let us now give the second set of four cases. Here, each proof of monotonicity follows from arguments similar to those given for the first set, and therefore it is omitted.

1. The ways e' and e view C are both non-decreasing with respect to last. We insert a constraint $[x \geq f(x')]$, where $f : \{0, \ldots, N\} \to \{0, \ldots, N\}$ is defined as follows.
   - For all $i > h'$: $f(i) = N$.
   - For $i = h', h' - 1, \ldots, \ell'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
     - If **(i)** $a = \mathsf{nil}$, **(ii)** $a + 1 \notin C$, or **(iii)** $\mathsf{last}(j, C) \geq a + 1$ for no $j \in e$, then $f(i) = f(i + 1)$.
     - Otherwise, let j be the smallest vertex in e such that $\mathsf{last}(j, C) \geq a + 1$, and define $f(i) = j$.
   - For all $i < \ell'$: $f(i) = 0$.
2. The way e' views C is non-decreasing with respect to last, and the way e views C is non-increasing with respect to last. We insert a constraint $[x \leq f(x')]$, where $f : \{0, \ldots, N\} \to \{0, \ldots, N\}$ is defined as follows.
   - For all $i > h'$: $f(i) = 0$.
   - For $i = h', h' - 1, \ldots, \ell'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
     - If **(i)** $a = \mathsf{nil}$, **(ii)** $a + 1 \notin C$, or **(iii)** $\mathsf{last}(j, C) \geq a + 1$ for no $j \in e$, then $f(i) = f(i + 1)$.
     - Otherwise, let j be the largest vertex in e such that $\mathsf{last}(j, C) \geq a + 1$, and define $f(i) = j$.
   - For all $i < \ell'$: $f(i) = N$.
3. The way e' views C is non-increasing with respect to last, and the way e views C is non-decreasing with respect to last. We insert a constraint $[x \geq f(x')]$, where $f : \{0, \ldots, N\} \to \{0, \ldots, N\}$ is defined as follows.
   - For all $i < \ell'$: $f(i) = N$.
   - For $i = \ell', \ell' + 1, \ldots, h'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
     - If **(i)** $a = \mathsf{nil}$, **(ii)** $a + 1 \notin C$, or **(iii)** $\mathsf{last}(j, C) \geq a + 1$ for no $j \in e$, then $f(i) = f(i - 1)$.
     - Otherwise, let j be the smallest vertex in e such that $\mathsf{last}(j, C) \geq a + 1$, and define $f(i) = j$.
   - For all $i > h'$: $f(i) = 0$.
4. The ways e' and e view C are both non-increasing with respect to last. We insert a constraint $[x \leq f(x')]$, where $f : \{0, \ldots, N\} \to \{0, \ldots, N\}$ is defined as follows.
   - For all $i < \ell'$: $f(i) = 0$.
   - For $i = \ell', \ell' + 1, \ldots, h'$: Denote $a = \mathsf{last}(i, C)$. We have two subcases.
     - If **(i)** $a = \mathsf{nil}$, **(ii)** $a + 1 \notin C$, or **(iii)** $\mathsf{last}(j, C) \geq a + 1$ for no $j \in e$, then $f(i) = f(i - 1)$.
     - Otherwise, let j be the largest vertex in e such that $\mathsf{last}(j, C) \geq a + 1$, and define $f(i) = j$.
   - For all $i > h'$: $f(i) = N$.

F Computation Time and Correctness of the Reduction in Section 3.3

The following observation directly follows from the definition of our reduction.

▶ **Observation F.1.** *For an instance* $I = (P, k, \mathsf{ig}, \mathsf{og}, \{\mathsf{how}_x\}|_{x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)})$ *of* Structured Art Gallery*,* $|X| = \mathcal{O}(r)$ *where* $\mathsf{reduction}(I) = (X, C, N)$*. Moreover,* reduction *is computable in polynomial time.*

To establish the correctness of our reduction, we start with the reverse direction.

▶ **Lemma F.1.** *Let* $I = (P, k, \mathsf{ig}, \mathsf{og}, \{\mathsf{how}_x\}|_{x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)})$ *be an instance of* Structured Art Gallery*, and denote* $\mathsf{reduction}(I) = (X, C, N)$*. If* (X, C, N) *is a* Yes*-instance of* Monotone 2-CSP*, then* I *is a* Yes*-instance of* Structured Art Gallery*.*

Proof. Suppose that (X, C, N) is a Yes-instance of Monotone 2-CSP. Accordingly, let $\alpha : X \to \{0, \ldots, N\}$ be a solution to (X, C, N). By the constraints in A, we have that for all $x \in X$, for $(e, i) = \mathsf{bij}^{-1}(x)$, it holds that $\alpha(x) \in e$.[14] In particular, for $S = \{\alpha(x) : x \in X\}$, we have that $S \subseteq V$. In what follows, we show that S is a solution to I, which would conclude the proof. Because $|X| \leq k$, we immediately have that $|S| \leq k$. Thus, it remains to show that Conditions 1, 2 and 3 in the definition of the objective of Structured Art Gallery are satisfied.

Condition 1. First, note that for each convex region or reflex vertex $y \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$, $|S \cap y| = |\{x \in X : (y, i) = \mathsf{bij}^{-1}(x) \text{ for some } i \in \{1, \ldots, \mathsf{ig}(y)\}\}| = \mathsf{ig}(y)$. Here, the first equality followed from the definition of S, and the last equality followed from the the fact that bij is bijective. Accordingly, for each $y \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$ and $i \in \{1, \ldots, \mathsf{ig}(y)\}$, let $s_{(y,i)}$ denote the i^{th} largest vertex in $S \cap y$; by the constraints in $A \cup O$, we have that $s_{(y,i)} = \alpha(x)$ for $x = \mathsf{bij}^{-1}(y, i)$.

Condition 2. Consider some reflex vertex $y \in \mathsf{reflex}(P)$, and denote $(e, i) = \mathsf{how}_{(y,1)}$. First, suppose that $e \in \mathsf{reflex}(P)$. Then, e sees y, else we would have outputted No. By the constraints in A, we have that $e = s_{\mathsf{how}_y(1)} \in S$, and hence $s_{\mathsf{how}_y(1)} \in S$ sees y. Second, suppose that $e \in \mathcal{C}(P)$. Then, since α satisfies the constraints c_y^1 and c_y^2, for the variable $x \in X$ that satisfies $\mathsf{bij}(x) = (e, i)$, we have that $\mathsf{first}(y, e) \leq \alpha(x) \leq \mathsf{last}(y, e)$. By Lemma 3.1, this means that $\alpha(x)$ sees y. Thus, because $s_{\mathsf{how}_y(1)} = s_{(e,i)} = \alpha(x)$, we have that $s_{\mathsf{how}_y(1)}$ sees y.

Condition 3a. In what follows, consider some convex region $C \in \mathcal{C}(P)$. Here, we need to show that $\mathsf{first}(s_{\mathsf{how}_C(1)}, C)$ is the smallest vertex in C. Denote $(e, i) = \mathsf{how}_C(1)$ and $x = \mathsf{bij}^{-1}(e, i)$. Additionally, denote the first vertex in C by q. First, suppose that $e \in \mathsf{reflex}(P)$. Then, e sees q, else we would have outputted No. By the constraints in A, we have that $e = s_{\mathsf{how}_C(1)} \in S$. Thus, $s_{\mathsf{how}_C(1)}$ sees q (which means that $\mathsf{first}(s_{\mathsf{how}_C(1)}, C)$ is the smallest vertex in C). Second, suppose that $e \in \mathcal{C}$. Let $\ell = \mathsf{first}(q, e)$ and $h = \mathsf{last}(q, e)$. If ℓ (and h) is nil, then we would have outputted No. Thus, by the constraints $c^1_{(C,1)}$ and $c^2_{(C,1)}$, we have that $\ell \leq \alpha(x) \leq h$. By Lemma 3.1, this means that $\alpha(x)$ sees q. Thus, because $s_{\mathsf{how}_C(1)} = s_{(e,i)} = \alpha(x)$, we have that $s_{\mathsf{how}_C(1)}$ sees q.

Condition 3c. Here, we need to show that $\mathsf{last}(s_{\mathsf{how}_C(\mathsf{og}(C))}, C)$ is the largest vertex in C. Denote $(e, i) = \mathsf{how}_C(\mathsf{og}(C))$ and $x = \mathsf{bij}^{-1}(e, i)$. Additionally, denote the last vertex in C by q. First, suppose that $e \in \mathsf{reflex}(P)$. Then, e sees q, else we would have outputted No. By the constraints in A, we have that $e = s_{\mathsf{how}_C(\mathsf{og}(C)} \in S$. Thus, $s_{\mathsf{how}_C(\mathsf{og}(C)}$ sees q (which means that $\mathsf{last}(s_{\mathsf{how}_C(\mathsf{og}(C))}, C)$ is the largest vertex in C). Second, suppose that $e \in \mathcal{C}$. Let $\ell = \mathsf{first}(q, e)$ and $h = \mathsf{last}(q, e)$. If ℓ (and h) is nil, then we would have outputted No. Thus,

[14] If $e \in \mathsf{reflex}(P)$, by $\alpha(x) \in e$ we mean $\alpha(x) = e$.

by the constraints $c^1_{(C,\mathsf{og}(C))}$ and $c^2_{(C,\mathsf{og}(C))}$, we have that $\ell \leq \alpha(x) \leq h$. By Lemma 3.1, this means that $\alpha(x)$ sees q. Thus, because $s_{\mathsf{how}_C(\mathsf{og}(C))} = s_{(e,i)} = \alpha(x)$, we have that $s_{\mathsf{how}_C(\mathsf{og}(C))}$ sees q.

Condition 3b. Lastly, we need to show that for every $t \in \{1, \dots, \mathsf{og}(C) - 1\}$, it holds that

$$\mathsf{first}(s_{\mathsf{how}_C(t+1)}, C) - 1 \leq \mathsf{last}(s_{\mathsf{how}_C(t)}, C) \leq \mathsf{last}(s_{\mathsf{how}_C(t+1)}, C) - 1.$$

Rephrased differently, we need to show that for every $t \in \{2, \dots, \mathsf{og}(C)\}$, it holds that

$$\mathsf{first}(s_{\mathsf{how}_C(t)}, C) - 1 \leq \mathsf{last}(s_{\mathsf{how}_C(t-1)}, C) \leq \mathsf{last}(s_{\mathsf{how}_C(t)}, C) - 1.$$

Observe that these inequalities encompass the requirement that $s_{\mathsf{how}_C(t)}$ sees at least one vertex in C. (Indeed, 1 cannot be subtracted from nil, and nil cannot be smaller or larger than an integer.) For $t = 1$, we only claim that $s_{\mathsf{how}_C(1)}$ sees at least one vertex in C. Now, the proof is by induction on t.[15] In the basis, where $t = 1$, the claim holds since we have already proved that Condition 3a is satisfied. Next, we suppose that the claim is true for all $t' \in \{1, \dots, t-1\}$, and prove it for $t \in \{2, \dots, \mathsf{og}(C)\}$

Denote $(e, \gamma) = \mathsf{how}_C(\mathsf{og}(t))$ and $x = \mathsf{bij}^{-1}(e, \gamma)$. In addition, denote $(e', \gamma') = \mathsf{how}_C(\mathsf{og}(t-1))$ and $x' = \mathsf{bij}^{-1}(e', \gamma')$. By the constraints in $A \cup O$, we have that $s_{\mathsf{how}_C(\mathsf{og}(t))} = s_{(e,\gamma)} = \alpha(x)$ and $s_{\mathsf{how}_C(\mathsf{og}(t-1))} = s_{(e',\gamma')} = \alpha(x')$. Denote $a = \mathsf{last}(\alpha(x'), C)$, and observe that $a \neq \mathsf{nil}$ by the inductive hypothesis. With this notation, our task is to show that *(i)* $a \geq b - 1$ for $b = \mathsf{first}(\alpha(x), C)$, and *(ii)* $a \leq q - 1$ for $q = \mathsf{last}(\alpha(x), C)$. If $a + 1 \notin C$, then the second condition cannot be satisfied. Therefore, it suffices to show that

1. either $a + 1 \in C$ or $a \geq b - 1$ for $b = \mathsf{first}(\alpha(x), C)$, and
2. $a \leq q - 1$ for $q = \mathsf{last}(\alpha(x), C)$.

The first set of four cases[16] is necessary mainly to prove the first condition above, and the second set of four cases is necessary mainly to prove the second condition above. However, to rule out the possibility that $b = q = \mathsf{nil}$, the first set of four cases is also required to prove the second condition, and the second set of four cases is also required to prove the first one. Thus, both conditions are proved simultaneously. In this context, let $c = [x\,\mathsf{sign}\,f(x')]$ be the constraint that was introduced due to appropriate case from the first set of four cases, and let $\widehat{c} = [x\widehat{\mathsf{sign}}\widehat{f}(x')]$ be the constraint that was introduced due to the appropriate case from the second set of four cases. We consider eight cases, depending on the way e' views C with respect to last, and the way e views C with respect to both first and last.

Case 1 of First Set. In this case, we suppose that the way e' views C is non-decreasing with respect to last, and the way e views C is non-decreasing with respect to first. Then, sign is equal to $\leq$. Moreover, in this case, $f(\alpha(x'))$ is defined as follows. (Here, recall that the possibility that $a = \mathsf{nil}$ has already been ruled out.) If $a + 1 \notin C$ or $\mathsf{first}(j, C) \leq a + 1$ for no $j \in e$, then $f(\alpha(x')) = f(\alpha(x') - 1)$. Otherwise, $f(\alpha(x'))$ is the largest vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a + 1$. In what follows, we suppose that $a + 1 \in C$ for the sake of the proof of Condition 1, else the proof of this condition is complete.

Since α is a solution to (X, C, N), we have that $\alpha(x) \leq f(\alpha(x'))$. In particular, since $\alpha(x) \notin \{0, N\}$ (because $\alpha(x) \in S$ and $S \subseteq V$), we have that $f(\alpha(x')) \neq 0$. To proceed our

[15] Here, induction is not mandatory. Instead, we can rely on the constraints marked with a tilde. However, these constraints are required for a different purpose (rather than only to encompass the inductive hypothesis). To highlight this, we prefer to use induction.

[16] See "guarding the middle vertices in a convex region" in Section 3.3.

analysis, we define δ and $a^\star$ as follows. Let δ be the largest vertex, not larger than $\alpha(x')$, such that $f(\delta) = f(\alpha(x'))$ and the following conditions hold for $a^\star = \mathsf{last}(\delta, C)$:

1. $a^\star \neq \mathsf{nil}$ and $a^\star + 1 \in C$;
2. $f(\alpha(x'))$ is the largest vertex $v \in e$ such that $\mathsf{first}(v, C) \leq a^\star + 1$.

The existence of such δ follows from the definition of f and because $f(\alpha(x')) \neq 0$. Since $\delta \leq \alpha(x')$ and the way e' views C is non-decreasing with respect to last, we have that $a^\star \leq a$. Thus, $\mathsf{first}(f(\alpha(x')), C) \leq a^\star + 1 \leq a + 1$. By the definition of $f(\alpha(x'))$, this means that $f(\alpha(x'))$ is the largest vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a + 1$. Because $\alpha(x) \leq f(\alpha(x')) = j$ and the way e views C is non-decreasing with respect to first, we have that either $\mathsf{first}(\alpha(x), C) \leq \mathsf{first}(j, C)$ or $\mathsf{first}(\alpha(x), C) = \mathsf{nil}$. In the first scenario, $b \leq a + 1$, hence the proof of Condition 1 is complete. (The second scenario is addressed ahead.)

Case 1 of First Set + Case 1 of Second Set. In this case, we suppose that e views C is non-decreasing with respect to last. Then, $\widehat{\mathsf{sign}}$ is equal to $\geq$. Moreover, in this case, $\widehat{f}(\alpha(x'))$ is defined as follows. (Here, recall that the possibility that $a = \mathsf{nil}$ has already been ruled out.) If $a + 1 \notin C$ or $\mathsf{last}(\widehat{j}, C) \geq a + 1$ for no $\widehat{j} \in e$, then $\widehat{f}(\alpha(x')) = \widehat{f}(\alpha(x') + 1)$. Otherwise, $\widehat{f}(\alpha(x'))$ is the smallest vertex $\widehat{j} \in e$ such that $\mathsf{last}(\widehat{j}, C) \geq a + 1$.

Since α is a solution to (X, C, N), we have that $\alpha(x) \geq \widehat{f}(\alpha(x'))$. In particular, since $\alpha(x) \notin \{0, N\}$ (because $\alpha(x) \in S$ and $S \subseteq V$), we have that $\widehat{f}(\alpha(x')) \neq N$. To proceed our analysis, we define $\widehat{\delta}$ and $\widehat{a}^\star$ as follows. Let $\widehat{\delta}$ be the smallest vertex, not smaller than $\alpha(x')$, such that $\widehat{f}(\widehat{\delta}) = \widehat{f}(\alpha(x'))$ and the following conditions hold for $\widehat{a}^\star = \mathsf{last}(\delta, C)$:

1. $\widehat{a}^\star \neq \mathsf{nil}$ and $\widehat{a}^\star + 1 \in C$;
2. $\widehat{f}(\alpha(x'))$ is the smallest vertex $\widehat{v} \in e$ such that $\mathsf{last}(\widehat{v}, C) \geq \widehat{a}^\star + 1$.

The existence of such $\widehat{\delta}$ follows from the definition of $\widehat{f}$ and because $\widehat{f}(\alpha(x')) \neq N$.[17] Since $\widehat{\delta} \geq \alpha(x')$ and the way e' views C is non-decreasing with respect to last, we have that $\widehat{a}^\star \geq a$. Thus, $\mathsf{last}(\widehat{f}(\alpha(x')), C) \geq \widehat{a}^\star + 1 \geq a + 1$, and hence $a + 1 \in C$. By the definition of $\widehat{f}(\alpha(x'))$, this means that $\widehat{f}(\alpha(x'))$ is the smallest vertex $\widehat{j} \in e$ such that $\mathsf{last}(\widehat{j}, C) \geq a + 1$. Because $\alpha(x) \geq \widehat{f}(\alpha(x')) = \widehat{j}$ and the way e views C is non-decreasing with respect to last, we have that either $\mathsf{last}(\alpha(x), C) \geq \mathsf{last}(\widehat{j}, C)$ or $\mathsf{last}(\alpha(x), C) = \mathsf{nil}$. In the first case, $q \geq a + 1$, hence the proof of Condition 2 is complete.

We are left with the scenario where $\mathsf{first}(\alpha(x), C) = \mathsf{last}(\alpha(x), C) = \mathsf{nil}$. To handle this scenario, recall that $\widehat{j} \leq \alpha(x) \leq j$, and $\mathsf{first}(j, C) \leq a+1 \leq \mathsf{last}(\widehat{j}, C)$. Because the way e views C is non-decreasing with respect to both first and last, the first chain of inequalities implies that $\mathsf{first}(\widehat{j}, C) \leq \mathsf{first}(j, C)$ and $\mathsf{last}(\widehat{j}, C) \leq \mathsf{last}(j, C)$. Thus, $\mathsf{first}(j, C) \leq a + 1 \leq \mathsf{last}(j, C)$ and $\mathsf{first}(\widehat{j}, C) \leq a + 1 \leq \mathsf{last}(\widehat{j}, C)$. By Lemma 3.1, we have that both j and $\widehat{j}$ see $a + 1$. In turn, by Lemma 3.1 and since $\widehat{j} \leq \alpha(x) \leq j$, this means that $\alpha(x)$ sees $a + 1$, which is a contradiction to $\mathsf{first}(\alpha(x), C) = \mathsf{last}(\alpha(x), C) = \mathsf{nil}$. Thus, this scenario cannot occur.

Case 1 of First Set + Case 2 of Second Set. In this case, we suppose that the way e views C is non-increasing with respect to last. Then, $\widehat{\mathsf{sign}}$ is equal to $\leq$. Moreover, in this case, $\widehat{f}(\alpha(x'))$ is defined as follows. (Here, recall that the possibility that $a = \mathsf{nil}$ has already been ruled out.) If $a + 1 \notin C$ or $\mathsf{last}(\widehat{j}, C) \geq a + 1$ for no $\widehat{j} \in e$, then $\widehat{f}(\alpha(x')) = \widehat{f}(\alpha(x') + 1)$. Otherwise, $\widehat{f}(\alpha(x'))$ is the largest vertex $\widehat{j} \in e$ such that $\mathsf{last}(\widehat{j}, C) \geq a + 1$.

Since α is a solution to (X, C, N), we have that $\alpha(x) \leq \widehat{f}(\alpha(x'))$. In particular, since $\alpha(x) \notin \{0, N\}$ (because $\alpha(x) \in S$ and $S \subseteq V$), we have that $\widehat{f}(\alpha(x')) \neq 0$. To proceed our

[17] If the function f were defined first for $i < \ell'$ rather than for $i > h'$, then the existence of $\widehat{\delta}$ would not have followed. Specifically, we need the integer that "propagates" in the definition of $\widehat{f}$ to be N rather than 0 because we have the assertion $\alpha(x) \geq \widehat{f}(\alpha(x'))$ rather than $\alpha(x) \leq \widehat{f}(\alpha(x'))$.

analysis, we define $\widehat{\delta}$ and $\widehat{a}^\star$ as follows. Let $\widehat{\delta}$ be the smallest vertex, not smaller than $\alpha(x')$, such that $\widehat{f}(\widehat{\delta}) = \widehat{f}(\alpha(x'))$ and the following conditions hold for $\widehat{a}^\star = \mathsf{last}(\delta, C)$:

1. $\widehat{a}^\star \neq \mathsf{nil}$ and $\widehat{a}^\star + 1 \in C$;

2. $\widehat{f}(\alpha(x'))$ is the largest vertex $\widehat{v} \in e$ such that $\mathsf{last}(\widehat{v}, C) \geq \widehat{a}^\star + 1$.

The existence of such $\widehat{\delta}$ follows from the definition of $\widehat{f}$ and because $\widehat{f}(\alpha(x')) \neq 0$. Since $\widehat{\delta} \geq \alpha(x')$ and the way e' views C is non-decreasing with respect to last, we have that $\widehat{a}^\star \geq a$. Thus, $\mathsf{last}(\widehat{f}(\alpha(x')), C) \geq \widehat{a}^\star + 1 \geq a + 1$, and hence $a + 1 \in C$. By the definition of $\widehat{f}(\alpha(x'))$, this means that $\widehat{f}(\alpha(x'))$ is the largest vertex $\widehat{j} \in e$ such that $\mathsf{last}(\widehat{j}, C) \geq a + 1$. Because $\alpha(x) \leq \widehat{f}(\alpha(x')) = \widehat{j}$ and the way e views C is non-increasing with respect to last, we have that either $\mathsf{last}(\alpha(x), C) \geq \mathsf{last}(\widehat{j}, C)$ or $\mathsf{last}(\alpha(x), C) = \mathsf{nil}$. In the first case, $q \geq a + 1$, hence the proof of Condition 2 is complete.

We are left with the scenario where $\mathsf{first}(\alpha(x), C) = \mathsf{last}(\alpha(x), C) = \mathsf{nil}$. To handle this scenario, recall that $\alpha(x) \leq \min(\widehat{j}, j)$. Due to the constraint $\widetilde{c}^1_{(C,t)} = [x \geq \ell]$, we have that $\ell \leq \alpha(x)$, and therefore $\ell \leq \min(\widehat{j}, j)$. Moreover, by the definition of ℓ, it sees at least one vertex in C. Thus, since the way e views C is non-decreasing with respect to first and non-increasing with respect to last, we have that $\mathsf{first}(\ell, C) \leq \mathsf{first}(j, C) \leq \mathsf{last}(j, C) \leq \mathsf{last}(\ell, C)$. By Lemma 3.1, this means that ℓ sees $\mathsf{first}(j, C)$. In turn, by Lemma 3.1 and since $\ell \leq \alpha(x) \leq j$, this means that $\mathsf{first}(j, C)$ sees $\alpha(x)$, which is a contradiction to $\mathsf{first}(\alpha(x), C) = \mathsf{last}(\alpha(x), C) = \mathsf{nil}$. Thus, this scenario cannot occur.

The proofs of the other three cases follow the same lines as the proof of the first case. For the sake of illustration, we give the details of the second case.

Case 2 of First Set. In this case, we suppose that the way e' views C is non-decreasing with respect to last, and the way e views C is non-increasing with respect to first. Then, sign is equal to $\geq$. Moreover, in this case, $f(\alpha(x'))$ is defined as follows. (Here, recall that the possibility that $a = \mathsf{nil}$ has already been ruled out.) If $a + 1 \notin C$ or $\mathsf{first}(j, C) \leq a + 1$ for no $j \in e$, then $f(\alpha(x')) = f(\alpha(x') - 1)$. Otherwise, $f(\alpha(x'))$ is the smallest vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a + 1$. In what follows, we suppose that $a + 1 \in C$ for the sake of the proof of Condition 1, else the proof of this condition is complete.

Since α is a solution to (X, C, N), we have that $\alpha(x) \geq f(\alpha(x'))$. In particular, since $\alpha(x) \notin \{0, N\}$ (because $\alpha(x) \in S$ and $S \subseteq V$), we have that $f(\alpha(x')) \neq N$. To proceed our analysis, we define δ and $a^\star$ as follows. Let δ be the largest vertex, not larger than $\alpha(x')$, such that $f(\delta) = f(\alpha(x'))$ and the following conditions hold for $a^\star = \mathsf{last}(\delta, C)$:

1. $a^\star \neq \mathsf{nil}$ and $a^\star + 1 \in C$;

2. $f(\alpha(x'))$ is the smallest vertex $v \in e$ such that $\mathsf{first}(v, C) \leq a^\star + 1$.

The existence of such δ follows from the definition of f and because $f(\alpha(x')) \neq N$. Since $\delta \leq \alpha(x')$ and the way e' views C is non-decreasing with respect to last, we have that $a^\star \leq a$. Thus, $\mathsf{first}(f(\alpha(x')), C) \leq a^\star + 1 \leq a + 1$. By the definition of $f(\alpha(x'))$, this means that $f(\alpha(x'))$ is the smallest vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a + 1$. Because $\alpha(x) \geq f(\alpha(x')) = j$ and the way e views C is non-increasing with respect to first, we have that either $\mathsf{first}(\alpha(x), C) \leq \mathsf{first}(j, C)$ or $\mathsf{first}(\alpha(x), C) = \mathsf{nil}$. In the first scenario, $b \leq a + 1$, hence the proof of Condition 1 is complete.

Case 2 of First Set + Case 1 of Second Set. In this case, we suppose that e views C is non-decreasing with respect to last. Then, $\widehat{\mathsf{sign}}$ is equal to $\geq$. By repeating the *exact* same arguments given in "Case 1 of First Set + Case 1 of Second Set", we derive that either $\mathsf{last}(\alpha(x), C) \geq \mathsf{last}(\widehat{j}, C)$ or $\mathsf{last}(\alpha(x), C) = \mathsf{nil}$. Indeed, all the arguments presented up to that point are oblivious to the way in which e views C with respect to first. In the first case (where $\mathsf{last}(\alpha(x), C) \geq \mathsf{last}(\widehat{j}, C)$), $q \geq a + 1$, hence the proof of Condition 2 is complete.

We are left with the scenario where $\mathsf{first}(\alpha(x), C) = \mathsf{last}(\alpha(x), C) = \mathsf{nil}$. To handle this scenario, recall that $\max(\widehat{j}, j) \leq \alpha(x)$. Due to the constraint $\widetilde{c}^2_{(C,t)} = [x \leq h]$, we have that $\alpha(x) \leq h$, and therefore $\max(\widehat{j}, j) \leq h$. Moreover, by the definition of h, it sees at least one vertex in C. Thus, since the way e views C is non-increasing with respect to first and non-decreasing with respect to last, we have that $\mathsf{first}(h, C) \leq \mathsf{first}(j, C) \leq \mathsf{last}(j, C) \leq \mathsf{last}(h, C)$. By Lemma 3.1, this means that h sees $\mathsf{first}(j, C)$. In turn, by Lemma 3.1 and since $j \leq \alpha(x) \leq h$, this means that $\mathsf{first}(j, C)$ sees $\alpha(x)$, which is a contradiction to $\mathsf{first}(\alpha(x), C) = \mathsf{last}(\alpha(x), C) = \mathsf{nil}$. Thus, this scenario cannot occur.

Case 2 of First Set + Case 2 of Second Set. In this case, we suppose that e views C is non-increasing with respect to last. By repeating the *exact* same arguments given in "Case 1 of First Set + Case 2 of Second Set", we derive that either $\mathsf{last}(\alpha(x), C) \geq \mathsf{last}(\widehat{j}, C)$ or $\mathsf{last}(\alpha(x), C) = \mathsf{nil}$. Indeed, all the arguments presented up to that point are oblivious to the way in which e views C with respect to first. In the first case (where $\mathsf{last}(\alpha(x), C) \geq \mathsf{last}(\widehat{j}, C)$), $q \geq a + 1$, hence the proof of Condition 2 is complete.

We are left with the scenario where $\mathsf{first}(\alpha(x), C) = \mathsf{last}(\alpha(x), C) = \mathsf{nil}$. To handle this scenario, recall that $j \leq \alpha(x) \leq \widehat{j}$, and $\mathsf{first}(j, C) \leq a+1 \leq \mathsf{last}(\widehat{j}, C)$. Because the way e views C is non-increasing with respect to both first and last, the first chain of inequalities implies that $\mathsf{first}(\widehat{j}, C) \leq \mathsf{first}(j, C)$ and $\mathsf{last}(\widehat{j}, C) \leq \mathsf{last}(j, C)$. Thus, $\mathsf{first}(j, C) \leq a + 1 \leq \mathsf{last}(j, C)$ and $\mathsf{first}(\widehat{j}, C) \leq a + 1 \leq \mathsf{last}(\widehat{j}, C)$. By Lemma 3.1, we have that both j and $\widehat{j}$ see $a + 1$. In turn, by Lemma 3.1 and since $j \leq \alpha(x) \leq \widehat{j}$, this means that $\alpha(x)$ sees $a + 1$, which is a contradiction to $\mathsf{first}(\alpha(x), C) = \mathsf{last}(\alpha(x), C) = \mathsf{nil}$. Thus, this scenario cannot occur. ◀

Now, we prove the correctness of the forward direction.

▶ **Lemma F.2.** *Let $I = (P, k, \mathsf{ig}, \mathsf{og}, \{\mathsf{how}_x\}|_{x \in \mathcal{C}(P) \cup \mathsf{reflex}(P)})$ be an instance of* Structured Art Gallery*, and denote $\mathsf{reduction}(I) = (X, C, N)$. If I is a* Yes*-instance of* Structured Art Gallery*, then (X, C, N) is a* Yes*-instance of* Monotone 2-CSP*.*

Proof. Suppose that I is a Yes-instance of Structured Art Gallery. Accordingly, let $S \subseteq V$ be a solution to I. Then, $|S| \leq k$, and the following conditions hold:

1. For each $y \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$, $|S \cap y| = \mathsf{ig}(y)$. Accordingly, for each $y \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$ and $i \in \{1, \ldots, \mathsf{ig}(y)\}$, let $s_{(y,i)}$ denote the i^{th} largest vertex in $S \cap y$.
2. For each $y \in \mathsf{reflex}(P)$, $s_{\mathsf{how}_y(1)}$ sees y.
3. For each $C \in \mathcal{C}(P)$, the following conditions hold:
   - **a.** $\mathsf{first}(s_{\mathsf{how}_C(1)}, C)$ is the smallest vertex in C.
   - **b.** For every $t \in \{1, \ldots, \mathsf{og}(C) - 1\}$, denote $a = \mathsf{last}(s_{\mathsf{how}_C(t)}, C)$, $j = \mathsf{first}(s_{\mathsf{how}_C(t+1)}, C)$ and $q = \mathsf{last}(s_{\mathsf{how}_C(t+1)}, C)$. Then, *(i)* $a \geq j - 1$, and *(ii)* $a \leq q - 1$.
   - **c.** $\mathsf{last}(s_{\mathsf{how}_C(\mathsf{og}(C))}, C)$ is the largest vertex in C.

In order to define an assignment $\alpha : X \to \{0, \ldots, N\}$, let $x \in X$. Denote $\mathsf{bij}(x) = (e, i)$. Accordingly, let t denote the i^{th} largest vertex t in $S \cap e$, namely, $s_{(e,i)}$. Then, define $\alpha(x) = t$. Since for $e \in \mathcal{C}(P) \cup \mathsf{reflex}(P)$, $|S \cap e| = \mathsf{ig}(e)$, and by the definition of the bijection bij, we have that t is well-defined. In what follows, we argue that α is a solution to (X, C, N). First, by the definition of α, it is clear that all of the constraints in $A \cup O$ are satisfied.

Guarding reflex vertices. Consider some $y \in \mathsf{reflex}(P)$. Note that $s_{\mathsf{how}_y(1)}$ sees y. Denote $(e, i) = \mathsf{how}_y(1)$. If $e \in \mathsf{reflex}(P)$, then e sees y and no constraint is introduced. Next, suppose that $e \in \mathcal{C}(P)$. Let $x \in X$ be the variable that satisfies $\mathsf{bij}(x) = \mathsf{how}_y(1)$. Denote $\ell = \mathsf{first}(y, e)$ and $h = \mathsf{last}(y, e)$. Since $s_{\mathsf{how}_y(1)}$ sees y, neither ℓ nor h is nil. We thus have the constraints $c^1_y = [x \geq \ell]$ and $c^2_y = [x \leq h]$. To prove that α satisfies them, we need to show that $\ell \leq \alpha(x) \leq h$. However, this directly follows from the fact that $\alpha(x) = s_{\mathsf{how}_y(1)}$ sees y.

In what follows, we consider some $C \in \mathcal{C}(P)$, and show that α satisfies all of the constraints introduced in the context of C.

Guarding the first vertex in a convex region. First, denote $(e, i) = \mathsf{how}_C(1)$ and $x = \mathsf{bij}^{-1}(e, i)$. In addition, denote the first vertex in C by q. Observe that $\mathsf{first}(s_{\mathsf{how}_C(1)}, C) = q$, which means that $s_{\mathsf{how}_C(1)}$ sees q. If $e \in \mathsf{reflex}(P)$, then e sees q and no constraint is introduced. Next, suppose that $e \in \mathcal{C}(P)$. Let $\ell = \mathsf{first}(q, e)$ and $h = \mathsf{last}(q, e)$. Since $s_{\mathsf{how}_C(1)}$ sees q, neither ℓ nor h is nil. We thus have the constraints $c^1_{(C,1)} = [x \geq \ell]$ and $c^2_{(C,1)} = [x \leq h]$. To prove that α satisfies them, we need to show that $\ell \leq \alpha(x) \leq h$. However, this directly follows from the fact that $\alpha(x) = s_{\mathsf{how}_C(1)}$ sees q.

Guarding the last vertex in a convex region. Secondly, denote $(e, i) = \mathsf{how}_C(\mathsf{og}(C))$ and $x = \mathsf{bij}^{-1}(e, i)$. In addition, denote the last vertex in C by q. Observe that $\mathsf{last}(s_{\mathsf{how}_C(\mathsf{og}(C))}, C) = q$, which means that $s_{\mathsf{how}_C(\mathsf{og}(C))}$ sees q. If $e \in \mathsf{reflex}(P)$, then e sees q and no constraint is introduced. Next, suppose that $e \in \mathcal{C}(P)$. Let $\ell = \mathsf{first}(q, e)$ and $h = \mathsf{last}(q, e)$. Since $s_{\mathsf{how}_C(\mathsf{og}(C))}$ sees q, neither ℓ nor h is nil. We thus have the constraints $c^1_{(C,\mathsf{og}(C))} = [x \geq \ell]$ and $c^2_{(C,\mathsf{og}(C))} = [x \leq h]$. To prove that α satisfies them, we need to show that $\ell \leq \alpha(x) \leq h$. However, this directly follows from the fact that $\alpha(x) = s_{\mathsf{how}_C(\mathsf{og}(C))}$ sees q.

Guarding the middle vertices in a convex region. Lastly, choose some $t \in \{2, \dots, \mathsf{og}(C)\}$. Denote $(e, i) = \mathsf{how}_C(t)$, $x = \mathsf{bij}^{-1}(e, i)$, $(e', i') = \mathsf{how}_C(t-1)$ and $x' = \mathsf{bij}^{-1}(e', i')$. Note that $\alpha(x) = s_{\mathsf{how}_C(t)} \in e$ and $\alpha(x') = s_{\mathsf{how}_C(t-1)} \in e'$. Recall that since S is a solution, we have that the vertex $a = \mathsf{last}(s_{\mathsf{how}_C(t-1)}, C)$ is *(i)* larger or equal to $b-1$ where $b = \mathsf{first}(s_{\mathsf{how}_C(t)}, C)$, and *(ii)* smaller than $q = \mathsf{last}(s_{\mathsf{how}_C(t)}, C)$. Note that $a = \mathsf{last}(\alpha(x'), C)$, $b = \mathsf{first}(\alpha(x), C)$ and $q = \mathsf{last}(\alpha(x), C)$. This implies that $a(x) \in e$ sees at least one vertex in C as well as that $a(x') \in e'$ sees at least one vertex in C. In particular, four constraints are introduced, and it is immediate that both $\widetilde{c}^1_{(C,t)}$ and $\widetilde{c}^2_{(C,t)}$ are satisfied.

In what follows, we need to show that α satisfies the constraints inserted in our two sets of four cases, which depend on the way e' views C with respect to last, and the way e views C with respect to both first and last. In the analysis of all cases below, when we identify $f(\alpha(x'))$, we rely on the fact that $a \neq \mathsf{nil}$ and $a + 1 \in C$ (because $a \leq q + 1$ and $q \in C$). Moreover, for the first set of four cases, we rely on the fact that there exists a vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a + 1$ (because $b \leq a + 1$). For the second set set of four cases, we rely on the fact that there exists a vertex $j \in e$ such that $\mathsf{last}(j, C) \geq a + 1$ (because $a \leq q - 1$). Here, the analysis of some of the cases is identical (e.g., the first and third cases of the first set); however, recall that in other proofs, these cases were analyzed differently (e.g., in the proof of monotonicity).

Case 1 of First Set. The way e' views C is non-decreasing with respect to last, and the way e views C is non-decreasing with respect to first. Let $c = [x \leq f(x')]$ be the constraint inserted in this case. To prove that α satisfies c, we need to show that $\alpha(x) \leq f(\alpha(x'))$. By the discussion before the case analysis, $f(\alpha(x'))$ is the largest vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a + 1$. Then, we need to show that $\alpha(x) \leq j$. However, since $\mathsf{first}(\alpha(x), C) \leq a + 1$, the inequality follows.

Case 2 of First Set. The way e' views C is non-decreasing with respect to last, and the way e views C is non-increasing with respect to first. Let $c = [x \geq f(x')]$ be the constraint inserted in this case. To prove that α satisfies c, we need to show that $\alpha(x) \geq f(\alpha(x'))$. By the discussion before the case analysis, $f(\alpha(x'))$ is the smallest vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a + 1$.

Then, we need to show that $\alpha(x) \geq j$. However, since $\mathsf{first}(\alpha(x), C) \leq a+1$, the inequality follows.

Case 3 of First Set. The way e' views C is non-increasing with respect to last, and the way e views C is non-decreasing with respect to first. Let $c = [x \leq f(x')]$ be the constraint inserted in this case. To prove that α satisfies c, we need to show that $\alpha(x) \leq f(\alpha(x'))$. By the discussion before the case analysis, $f(\alpha(x'))$ is the largest vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a+1$. Then, we need to show that $\alpha(x) \leq j$. However, since $\mathsf{first}(\alpha(x), C) \leq a+1$, the inequality follows.

Case 4 of First Set. The way e' views C is non-decreasing with respect to last, and the way e views C is non-increasing with respect to first. Let $c = [x \geq f(x')]$ be the constraint inserted in this case. To prove that α satisfies c, we need to show that $\alpha(x) \geq f(\alpha(x'))$. By the discussion before the case analysis, $f(\alpha(x'))$ is the smallest vertex $j \in e$ such that $\mathsf{first}(j, C) \leq a+1$. Then, we need to show that $\alpha(x) \geq j$. However, since $\mathsf{first}(\alpha(x), C) \leq a+1$, the inequality follows.

Case 1 of Second Set. The ways e' and e view C are both non-decreasing with respect to last. Let $c = [x \geq f(x')]$ be the constraint inserted in this case. To prove that α satisfies c, we need to show that $\alpha(x) \geq f(\alpha(x'))$. By the discussion before the case analysis, $f(\alpha(x'))$ is the smallest vertex $j \in e$ such that $\mathsf{last}(j, C) \geq a+1$. Then, we need to show that $\alpha(x) \geq j$. However, since $\mathsf{last}(\alpha(x), C) \geq a+1$, the inequality follows.

Case 2 of Second Set. The ways e' and e view C are non-decreasing and non-increasing, respectively, with respect to last. Let $c = [x \leq f(x')]$ be the constraint inserted in this case. To prove that α satisfies c, we need to show that $\alpha(x) \leq f(\alpha(x'))$. By the discussion before the case analysis, $f(\alpha(x'))$ is the largest vertex $j \in e$ such that $\mathsf{last}(j, C) \geq a+1$. Then, we need to show that $\alpha(x) \leq j$. However, since $\mathsf{last}(\alpha(x), C) \geq a+1$, the inequality follows.

Case 3 of Second Set. The ways e' and e view C are non-increasing and non-decreasing, respectively, with respect to last. Let $c = [x \geq f(x')]$ be the constraint inserted in this case. To prove that α satisfies c, we need to show that $\alpha(x) \geq f(\alpha(x'))$. By the discussion before the case analysis, $f(\alpha(x'))$ is the smallest vertex $j \in e$ such that $\mathsf{last}(j, C) \geq a+1$. Then, we need to show that $\alpha(x) \geq j$. However, since $\mathsf{last}(\alpha(x), C) \geq a+1$, the inequality follows.

Case 4 of Second Set. The ways e' and e view C are both non-increasing with respect to last. Let $c = [x \leq f(x')]$ be the constraint inserted in this case. To prove that α satisfies c, we need to show that $\alpha(x) \leq f(\alpha(x'))$. By the discussion before the case analysis, $f(\alpha(x'))$ is the largest vertex $j \in e$ such that $\mathsf{last}(j, C) \geq a+1$. Then, we need to show that $\alpha(x) \leq j$. However, since $\mathsf{last}(\alpha(x), C) \geq a+1$, the inequality follows. ◀

G Discretization for Boundary-Vertex Art Gallery and Vertex-Boundary Art Gallery

In this section we show how we can discretize the given polygon to solve Boundary-Vertex Art Gallery and Vertex-Boundary Art Gallery, using the techniques used by our algorithm for Vertex-Boundary Art Gallery.

We create a set $\mathsf{Ess}(P)$ of "essential points" of P, which will be useful for "discretization".

▶ **Definition 6.** *Consider a simple polygon P with $V(P) = \{1, 2, \cdots, n\}$ and $E(P) = \{\{i, i+1\} : i \in [n]\}$ (computation modulo n). The* essential set *of P is the set* $\mathsf{Ess}(P)$ *constructed as follows. Initially,* $\mathsf{Ess}(P)$ *contains all the vertices of P. For every distinct*

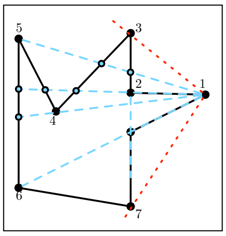

Figure 11 A (partial) illustration of the construction of $\mathsf{Ess}(P)$. The labelled vertices are the vertices of the polygon, whereas the blue vertices are the newly added vertices.

vertices $i, j \in [n]$, consider the line L_{ij} containing i and j. For each edge $e = \{i', j'\}$ which is not a sub-segment of L_{ij}, we add the intersection point (if it exists) of L_{ij} and the line segment $\overline{i'j'}$, to the set $\mathsf{Ess}(P)$.

Note that $\mathsf{Ess}(P)$ can be computed in polynomial time. (We remark that by constructing $\mathsf{Ess}(P)$ more carefully (than what we do), we may optimize its size, but we choose to construct it this way to keep the definition simple.) Let P_1 be the polygon with vertex set $\mathsf{Ess}(P)$, obtained from P by sub-dividing edges of P (possibly multiple times).

In the Boundary-Vertex Art Gallery problem, the guards are placed on the boundary of P and the objective is to guard the vertices of P. In the next lemma shows that if the given instance (P, k) of Boundary-Vertex Art Gallery is a yes-instance, then there is a solution which places guards only at vertices from P_1.

▶ **Lemma G.1.** *Let (P, k) be a yes-instance of* Boundary-Vertex Art Gallery*. Then there is a solution $S \subseteq V(P_1)$ to the instance (P, k) of* Boundary-Vertex Art Gallery*.*

Proof. Consider a minimal solution S to (P, k), where S is a set of points from the boundary of P of size at most k, and S is a solution that maximizes $|V(P_1) \cap S|$. We will show that $S \subseteq V(P_1)$. Towards a contradiction suppose that $S \not\subseteq V(P_1)$, and consider a point $q \in S \setminus V(P_1)$. As $q \notin V(P_1)$, there is a unique edge in P_1 containing it, denote that edge by $e = \{u, w\}$, where $u < w$. Let $S' = (S \setminus \{q\}) \cup \{u\}$. We will show that S' is also a solution for the instance (P, k), thus contradicting the choice of S. To prove that S' is a solution, it is enough to show that for every $v \in V(P)$ that is seen by q, u also sees v. Consider some $v \in V(P)$ that is seen by q. Towards a contradiction assume that u does not see v. Let T be the triangle defined by v, u and q. As u does not see v and $q \notin V(P_1)$, T is a non-degenerate triangle. Also the line segment $\overline{uv}$ is not completely contained in P (or P_1), and thus there is a reflex vertex v^* from P that is either strictly contained inside T or contained in the line segment $\overline{vq}$. In either case, the line L containing v and v^* intersects $\overline{uq}$ at a point different than u. This contradicts that $\{u, w\}$ is the edge in P_1 containing q, where $q \notin V(P_1)$. This concludes the proof. ◀

We now briefly explain how we can obtain an FPT algorithm for Boundary-Vertex Art Gallery using the techniques that we used in Section 3 and Lemma G.1. Let (P, k) be an instance of Boundary-Vertex Art Gallery, and define P_1 as was described earlier. The first component of our algorithm for Vertex-Vertex Art Gallery was a Turing reduction to a structured form of Art Gallery, called Structured Art Gallery (see Section 3.2). We can define a Structured Boundary-Vertex Art Gallery which takes an additional input, which is the set of vertices to be guarded. In additional to all other inputs, we provide P_1 as the input polygon and $V(P) \subseteq V(P_1)$ as the set of vertices to be guarded. The safeness of the above Turing reduction can be obtained from Lemma G.1 and arguments similar to the one used for the proof of Lemma 3.3. The next step is to reduce the structured instance to an instance of MONOTONE CSP. We follow similar procedure as given in Section 3.3, but we restrict the ranges for the functions to vertices appearing in $V(P)$. Finally, we resolve the instance by solving the instances of MONOTONE CSP, using Theorem 2. From the above discussions we can obtain the following theorem.

▶ **Theorem 7.** BOUNDARY-VERTEX ART GALLERY *is* FPT *parameterized by* r*, the number of reflex vertices. In particular, it admits an algorithm with running time* $r^{\mathcal{O}(r^2)}n^{\mathcal{O}(1)}$.

Next we turn to VERTEX-BOUNDARY ART GALLERY. Recall that in the VERTEX-BOUNDARY ART GALLERY problem, the guards are to be placed on the vertices of P and the goal is to guard the whole boundary of P. We obtain P_1 from P as was described earlier. Furthermore, we obtain P_2 from P_1 by sub-dividing each edge of P_1 exactly once. In the next lemma we show that any set that guards all vertices of P_2, guards the whole boundary of P.

▶ **Lemma G.2.** *Let* (P, k) *be an instance of* VERTEX-BOUNDARY ART GALLERY*. Consider a set* $S \subseteq V(P)$ *of size at most* k*, such that for each* $v \in V(P_2)$*, there is* $s \in S$ *that sees* v*. Then* S *is a solution to the instance* (P, k) *of* VERTEX-BOUNDARY ART GALLERY*.*

Proof. Consider a point p in the boundary of P which is not a vertex of P_2. Let $\{u, w\}$ be the edge in P_1 that contains p strictly in its interior. By the construction of P_2, there is a vertex $v \in V(P_2) \setminus V(P_1)$ contained strictly inside the line segment $\overline{uw}$. Consider $s \in S$ such that s sees v. We will show that s sees p. Towards a contradiction, suppose that s does not see p. Consider the triangle T formed by p, v and s. As s does not see p, we can conclude that T is non-degenerate and $\overline{ps}$ is not completely contained in P. Thus, there is a reflex vertex $\widehat{v}$ which is either strictly contained inside T, or contained in the line segment $\overline{sv}$. If $\widehat{v}$ is strictly contained in the interior of T, then we can contradict that $\{u, w\}$ is the edge in P_1 containing p. Otherwise, if $\widehat{v}$ is contained in the line segment $\overline{sv}$, and we can obtain a contradiction to the fact that $v \in V(P_2) \setminus V(P_1)$. Thus, we obtain that s sees p. This concludes the proof. ◀

Now we explain how we can obtain an FPT algorithm for VERTEX-BOUNDARY ART GALLERY using the techniques that we used in Section 3 and Lemma G.2. Let (P, k) be an instance of VERTEX-BOUNDARY ART GALLERY, and define P_1 and P_2, as was described earlier. Again we define a structured form of the problem called STRUCTURED BOUNDARY-VERTEX ART GALLERY, which takes an additional set of vertices from which the guards can be selected. We give Turing reduction from VERTEX-BOUNDARY ART GALLERY to STRUCTURED BOUNDARY-VERTEX ART GALLERY, where apart from the other inputs, the input polygon is P_2 and the set from which we are allowed to select guards is $V(P)$. We can obtain the correctness of the above Turing reduction using Lemma G.2 and arguments similar to the one used for the proof of Lemma 3.3. The next step is to reduce the structured instance to an instance of MONOTONE CSP. We follow similar procedure as given in Section 3.3, but this time we restrict the domains for the functions to vertices appearing in $V(P)$. Finally, we resolve the instance by solving the instances of MONOTONE CSP, using Theorem 2. From the above discussions we can obtain the following theorem.

▶ **Theorem 8.** VERTEX-BOUNDARY ART GALLERY *is* FPT *parameterized by* r*, the number of reflex vertices. In particular, it admits an algorithm with running time* $r^{\mathcal{O}(r^2)}n^{\mathcal{O}(1)}$.