Floodlight Illumination of Infinite Wedges

Matthew Cary, Atri Rudra

Computer Sci. and Engr., Univ. of Washington, Seattle, WA 98195-2350, U.S.A.

Ashish Sabharwal [*,1]

Dept. of Computer Science, Cornell University, Ithaca, NY 14853-7501, U.S.A.

Erik Vee [1]

Yahoo! Research, Santa Clara, CA 95054, U.S.A.

Abstract

The floodlight illumination problem asks whether there exists a one-to-one placement of n floodlights illuminating infinite wedges of angles $\alpha_1, \ldots, \alpha_n$ at n sites $p_1, \ldots, p_n$ in a plane such that a given infinite wedge W of angle θ located at point q is completely illuminated by the floodlights. We prove that this problem is NP-hard, closing an open problem posed by Demaine and O'Rourke (CCCG 2001). In fact, we show that the problem is NP-complete even when $\alpha_i = \alpha$ for all $1 \leq i \leq n$ (the *uniform* case) and $\theta = \sum_{i=1}^{n} \alpha_i$ (the *tight* case).

Key words: illumination, art gallery problem, floodlights, NP-completeness

1 Introduction

Illumination problems generalize the well-known art gallery problem (see, e.g., [12, 13]). The task is to mount lights at various sites so that a given region, typically a non-convex polygon, is completely illuminated. The sites can be fixed in advance or not. The region may need to be illuminated from outside

* Corresponding author.
Email addresses: `cary@cs.washington.edu` (Matthew Cary), `atri@cs.washington.edu` (Atri Rudra), `sabhar@cs.cornell.edu` (Ashish Sabharwal), `erikvee@yahoo.com` (Erik Vee).

1 This work was done while the authors were at the University of Washington.

(like a soccer field) or from inside (like an indoor gallery). The lights may behave like ideal light bulbs, illuminating all directions equally, or like floodlights, illuminating a certain angle in a certain direction. We use this latter model of floodlights in this paper. This model is quite natural and captures scenarios involving guards or security cameras with restricted angle of vision. Illumination algorithms using floodlights have focused in the past on illuminating the interior of orthogonal polygons [8, 1] and general polygons with restrictions on the floodlights used [2, 9, 7, 17, 14].There has also been work on the stage illumination problem where one tries to illuminate lines rather than polygons [5].

The problem of illumination of *infinite wedges* by floodlights was introduced by Bose et al. [3]. We refer to Figure 1 for the basic setup and definitions. An infinite wedge $W \subseteq \mathbb{R}^2$ is any one of the four regions into which $\mathbb{R}^2$ is partitioned by two intersecting lines. The boundaries of W are sometimes referred to as the *rays* of W. A *generalized wedge* is any unbounded, convex, polygonal subset of W (even W itself) whose infinite edges are the infinite sub-rays of W. For the floodlight illumination problem, given n sites and n floodlights, the task is to mount these floodlights, one at each site, and orient them so that a given generalized wedge is completely illuminated. Formally,

Definition 1. The FLOODLIGHT ILLUMINATION Problem:
Instance: Sites $p_1, \ldots, p_n$ in $\mathbb{R}^2$, angles $\alpha_1, \ldots, \alpha_n > 0$, and a generalized wedge W of angle θ.
Question: Viewing the angles as spans of floodlights, is there an assignment of angles to sites along with angle orientations, that completely illuminates W?

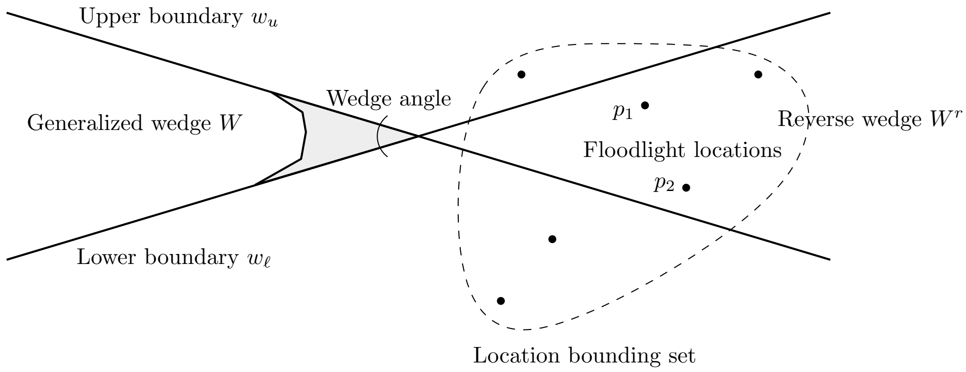

Fig. 1. Basic definitions. W.l.o.g. the axis of W always points along the negative x-axis in $\mathbb{R}^2$.

Let $[n]$ denote the set $\{1, 2, \ldots, n\}$. A couple of natural restrictions of the floodlight illumination problem are the *uniform* case where $\alpha_i = \alpha$ for all $i \in [n]$, and the *tight* case where $\sum_{i=1}^{n} \alpha_i = \theta$. There is clearly no solution to the problem when $\sum_{i=1}^{n} \alpha_i < \theta$. In general, a solution can be described by a mapping of floodlights to sites along with an angle of orientation for each floodlight. Moreover, in the tight case, every solution can alternatively be described by two permutations σ and τ of $[n]$, as observed by Steiger and Streinu [15]. Here σ is an ordering of the floodlights and $p_{\tau(i)}$ is the site at which floodlight of angle $\alpha_{\sigma(i)}$ is mounted. Floodlight orientations in this solution are inferred from σ and τ as follows. First $\alpha_{\sigma(1)}$ is mounted at position $p_{\tau(1)}$ and oriented so that its upper ray is parallel to the upper boundary w_u of W, and then, for $2 \leq i \leq n$, $\alpha_{\sigma(i)}$ is mounted at position $p_{\tau(i)}$ and oriented so that its upper ray is parallel to the lower ray of $\alpha_{\sigma(i-1)}$. The variant of the tight floodlight illumination problem where σ is fixed in advance will be called the *restricted* case. Informally, this says that for algorithmic purposes, the order in which floodlights are mounted is fixed in advance. When talking about the restricted case, we will think of σ as the identity permutation. Observe that a tight and uniform problem is also effectively restricted because all choices of σ are equivalent. Our results show that in general, for every choice of σ, computing τ is NP-complete.

Because of hardness of verification issues surrounding non-algebraic numbers, it is not clear whether the general problem is in the class NP. In fact, it is not obvious that it even has an exponential time solution. Nonetheless, Steiger and Streinu [15] proved that it can indeed be solved in exponential time by formulating it as a bounded quantifier expression in Tarski's algebra [16] and using the result of Grigor'ev [11] on the complexity of deciding the truth value of such expressions. They also proved that the restricted floodlight illumination problem is the dual of a certain *monotone matching* problem with lines and slabs. We will use this latter problem to provide an NP-hardness reduction.

The problem in the tight case does not have complications with non-algebraic numbers because the solution, as mentioned earlier, can be expressed as two permutations on $[n]$. The tight case of the problem is obviously in NP. However, the exact complexity of this problem has been unknown [6]. We resolve this open question by showing the following.

Theorem 2. Floodlight Illumination *is NP-hard. The tight, restricted, and uniform versions of the problem are NP-complete.*

This is an immediate consequence of the discussion of duality in Section 2.2 and our NP-completeness result for a uniform version of the monotone matching problem (Theorem 15). We will prove NP-hardness by a reduction from the propositional satisfiability problem 3-Sat to the monotone matching problem. This reduction, which forms the main technical contribution of this paper, involves several kinds of gadgets put together somewhat delicately in order to capture the satisfiability problem.

There are several natural notions of approximation for the floodlight illumination problem. We consider two of these, a *finite-region* approximation where one illuminates all but a small finite region of W, and a *finite-angle* approximation where one illuminates all but an infinite wedge of a small finite angle within W. We prove the following as an immediate consequence of Lemmas 7 and 10.

Theorem 3. *For the tight floodlight illumination problem, computing a finite-region approximation is NP-hard, whereas for any $\varepsilon > 0$, an ε-angle approximation can be constructed in polynomial time.*

The rest of the paper is organized as follows. In Section 2 we describe the dual monotone matching problem. As a warm-up, in Section 3 we make some comments on algorithms for illumination and discuss two notions of approximations. Finally, in Section 4 we discuss in detail our proof of the NP-hardness of the floodlight illumination problem by reducing 3-SAT to the dual problem of monotone matching.

2 Monotone Matching: A Recapitulation of the Duality

In this section, we formally define the monotone matching problem and recapitulate its duality with respect to the restricted floodlight illumination problem.

2.1 Monotone Matching

Suppose we are given n non-vertical lines in the plane $\mathbb{R}^2$, $n+1$ vertical lines defining n finite width vertical slabs, and two points, one on the leftmost vertical line and one on the rightmost. Call this an *n-arrangement of lines, slabs, and points* and denote it by (L, S, λ, ρ) where $L \equiv \{(m_1, c_1), \ldots, (m_n, c_n)\}$ is the set of lines $y = m_i x + c_i$ with $m_i, c_i \in \mathbb{R}$ for $i \in [n]$, $S \equiv \{s_1, \ldots, s_{n+1}\}$ is the set of vertical lines $x = s_j$ with $s_j \in \mathbb{R}$ for $j \in [n]$ forming slabs, and λ and ρ are the two special points in $\mathbb{R}^2$ on the vertical lines $x = s_1$ and $x = s_{n+1}$, respectively. The intersection of any non-vertical line ℓ with a slab s will be referred to as the *segment* of ℓ spanning s. We will think of slabs and segments towards the left-hand-side (i.e., the negative x-axis) as appearing "before" or "previous" to those towards the right-hand-side. Similarly, points and objects in $\mathbb{R}^2$ with larger y-coordinates will be thought of as "above" those with smaller y-coordinates.

A *monotone matching* in (L, S, λ, ρ) is a bijection between the lines and slabs, inducing a set of n line segments, each of which is a portion of a unique line and spans a unique slab, such that the following holds: (1) the left endpoint of the first segment is at or above λ, (2) the left endpoint of each subsequent segment is at or above the right endpoint of the segment in the previous slab, and (3) ρ is at or above the right endpoint of the last segment.

Definition 4. The MONOTONE MATCHING Problem [15]:
Instance: An n-arrangement (L, S, λ, ρ) of lines, slabs, and points in $\mathbb{R}^2$.
Question: Does this arrangement contain a monotone matching?

Analogous to the floodlight illumination case, define the specialized *uniform* version UNIFORM MONOTONE MATCHING to be the problem where all slabs have the same width. The key technical contribution of this paper is a proof that this variant is NP-complete (see Section 4). By the duality argument we will discuss shortly, this implies the hardness of the floodlight illumination problem as well, which is our main focus.

2.2 Duality Between Floodlight Illumination and Monotone Matching

The restricted floodlight problem can be related to the monotone matching problem through duality [15]. The dual of a point p with coordinates (a, b) is the line T_p with equation $y = ax + b$; the dual of a line ℓ with equation $y = mx + c$ is the point T_ℓ with coordinates $(-m, c)$. It is well known that this dual transformation preserves incidence and height ordering; i.e., if a point p intersects a line ℓ then their duals T_p and T_ℓ also intersect, and if p is above ℓ then T_p is above T_ℓ.

We now describe how any restricted floodlight illumination problem can be converted to its dual monotone matching problem (L, S, λ, ρ). Let us first fix some notation (see Figure 1 for an illustration). We will use W to denote the wedge under consideration. The lower and upper boundaries of W will be denoted by w_l and w_u, respectively. We will assume w.l.o.g. that the axis of W points towards the negative x-axis in the plane. Note that this implies that the lines corresponding to w_l and w_u have positive and negative slopes, respectively. Finally, the angle subtended at the apex of W will be called the wedge angle, and the reverse wedge will be denoted by W^r.

The duals of the wedge boundaries w_l and w_u are the points λ and ρ; as w_l has larger slope in the orientation of the figure, its dual λ has smaller x coordinate. The points that form the vertical line containing λ are the duals of the lines that are parallel to w_l. The vertical strip between λ and ρ corresponds to the wedge angle; the larger the wedge angle, the wider the strip. The line q that connects λ and ρ is the dual of the point at which w_l and w_u intersect, forming the apex of the wedge. The segment of q between λ and ρ corresponds to the lines with slope less than w_l and greater than w_u that have common intersection with w_l and w_u; this is exactly the set of lines that lie within $W \cup W^r$.

Each site p_i for $i \in [n]$ corresponds to a line h_i which together make up the set of lines L. As we are in the restricted version of the problem, the angle of the first floodlight can be assumed w.l.o.g. to be α_1. From our discussion in Section 1, the tightness of the problem implies that the first floodlight must be oriented so that its upper ray is parallel to w_u. This floodlight corresponds in the dual to a vertical slab S_1 beginning at ρ and extending to the left a width proportional to α_1 (if S_1 extends from $x = s_1$ to $x = s_2$, then $\alpha = \tan^{-1} s_2 - \tan^{-1} s_1$). The next floodlight corresponds to a slab vertical S_2 extending to the left of S_1, continuing to the final floodlight which is a vertical slab S_n ending at λ.

A solution to the restricted problem is an assignment of sites to floodlights. In the dual this is a 1-1 assignment of lines h_i to slabs S_j for $i, j \in [n]$. The illumination wedge of the first floodlight must overlap w_u, which corresponds to the right endpoint of the segment of the line h_i^1 assigned to s_1 being at or above ρ. Continuing, the right endpoint of the segment associated with S_2 must start at or above the left endpoint of the segment of S_1, and so on, until the left endpoint of the segment at S_n is below λ.

If we flip the dual problem from left to right, in deference to those of us who read from left to right, we see that we have a solution-preserving duality between the restricted floodlight illumination problem and the monotone matching problem with lines. In Section 4, we will restrict our attention to proving that the monotone matching problem is difficult to solve.

Note that the unrestricted tight illumination problem corresponds to an extended matching problem where the widths of slabs are given and must be arranged in a partition of the slab between λ and ρ before a matching of segments to slabs is found. The uniform illumination problem corresponds to the uniform matching problem, where the slabs are all of the same width, making, in particular, their order immaterial.

3 Comments on Illumination Algorithms

We begin this section by describing some relatively simple properties of the floodlight illumination problem, and then discuss finite-angle approximations. The reader interested mostly in the NP-hardness reduction may choose to skip this section and jump directly to Section 4.

We start with the following definition of approximation for the floodlight illumination problem.

Definition 5. Let $\mathcal{F}$ be an illumination of a wedge W. $\mathcal{F}$ is a *finite-region approximation* if it illuminates $W \setminus S$, where $S \subseteq \mathbb{R}^2$ is a finite region.

Note that floodlight illumination on a generalized wedge is a special case of a finite-region approximation. Our proof of Theorem 2 will show that the floodlight illumination problem on a wedge is NP-hard. We now prove that at least in the tight case, computing a finite-region approximation is not any easier than computing an exact illumination of a wedge. Thus, computing a finite-region approximation to a floodlight illumination in the tight case is also NP-hard.

The following lemma describes what the shape of an illuminated region might be in the tight case.

Lemma 6. *Suppose the sum of the angles of n floodlights is α. If they illuminate a wedge W of angle α, then the overall illuminated region is of the form $W' \cup S$, where $W' \supseteq W$ is a wedge of angle α aligned with W and $S \subseteq \mathbb{R}^2$ is a finite region.*

Proof. We prove by induction on n the weaker statement that ignores the requirement $W' \supseteq W$ above. This will, however, suffice for the proof because if W' did not include all of W, $W' \cup S$ would also not include all of W since S is finite. This will contradict the precondition.

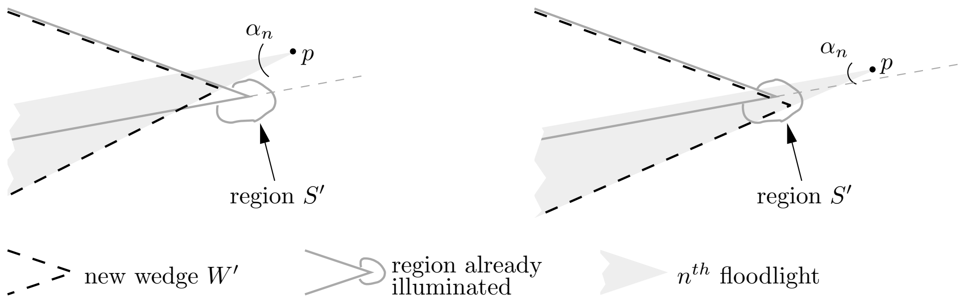

Fig. 2. Two cases for the shape of overall illuminated region in the tight case

For the base case of $n = 1$, we have $W' = W$ and S is the empty region. For $n > 1$, note that the floodlight angles are tight relative to W. Hence, the first $n-1$ floodlights must together cover some wedge of angle precisely $\alpha - \alpha_n$, where α_n is the angle of the n^{th} floodlight (see Figure 2). By induction, the region illuminated by the first $n-1$ floodlights is of the form $W'' \cup S'$, where W'' is a wedge of angle $\alpha - \alpha_n$ and $S' \subseteq \mathbb{R}^2$ is a finite region. Exploiting again the fact that the angles are tight, the only way to extend the already illuminated region in order to cover all of W is to mount the n^{th} floodlight f at a site p that is on or above the lower boundary w_l of $W'' \cup S'$ and have its upper boundary f_u aligned with w_l. Let W' be the wedge of angle α defined by the upper boundary w_u of the already illuminated region and the lower boundary f_l of f. As seen from the two cases in Figure 2, the overall illuminated region is $W' \cup S$, where S is finite. □

We now argue the equivalence of computing an illumination of a wedge and a finite-region approximation to it in the tight case.

Lemma 7. *There is a solution to a tight floodlight illumination problem on a wedge W if and only if there is a finite-region approximation to it.*

Proof. We prove that having a finite-region approximation is sufficient for having an exact solution; the other direction holds trivially. Suppose there is a finite-region approximate illumination $\mathcal{F}$ for W. Let W be of angle α. By definition, $\mathcal{F}$ must illuminate a wedge W^* of angle α that is aligned with W but is possibly contained strictly within W. Since the floodlight angles are tight relative to W^*, by Lemma 6, the overall region illuminated by $\mathcal{F}$ is of the form $W' \cup S$, where W' is a wedge of angle α aligned with W^* (and hence with W) and S is finite. If $W \not\subseteq W'$, then $W' \setminus W$ is an infinite region R. As S is finite, $W' \cup S$ will not cover an infinite portion of this infinite region R of W, contradicting the fact that $\mathcal{F}$ illuminates all but a finite region of W. It follows that $W \subseteq W'$, implying that W is completely illuminated by $\mathcal{F}$ and providing an exact solution. □

This lemma implies that computing a finite-region approximation is NP-hard because computing the exact solution is. It also implies that there is a solution to the tight floodlight problem on a generalized wedge W iff there is a solution to the tight floodlight problem on the underlying normal wedge W'. In this sense, generalized wedges don't make the problem any harder. However, they provide a convenient tool for analysis, allowing, for instance, stronger inductive claims.

The fact that computing a finite-region approximation to a wedge illumination problem is also NP-hard motivates the exploration of other reasonable notions of approximation that leave unlit relatively small but infinite regions of the wedge. One such possibility is a finite-angle approximation, defined as:

Definition 8. Let $\mathcal{F}$ be an illumination of a wedge W. $\mathcal{F}$ is an *ε-angle approximation* if it illuminates $W \setminus S_\varepsilon$, where S_ε is a union of wedges whose total angle is at most ε. When ε is a fixed constant, $\mathcal{F}$ is a *finite-angle approximation.*

We will show that a finite-angle approximation to a wedge illumination in the tight case can be computed in polynomial time. To prove this claim, we first need to make some observations.

Steiger and Streinu [15] gave a polynomial-time duality-based greedy algorithm $\mathcal{A}_{\text{greedy}}$ for the floodlight illumination problem which takes an additional input: the order in which the floodlight angles are chosen, that is, permutation σ from Section 1. Note that for the uniform case, where each floodlight angle is the same, the permutation σ does not come into play and $\mathcal{A}_{\text{greedy}}$ is applicable. The following fact will be useful in designing a polynomial-time algorithm for finite-angle approximations.

Fact 9. *If all sites are contained inside the reverse wedge W^r, then $\mathcal{A}_{greedy}$ successfully illuminates W after any assignment of floodlights to positions.*

For the following argument, we will use as a (temporary) tool the concept of "movable sites" which can be placed anywhere in the plane at the discretion of the solution designer. The idea is that in the presence of two such movable sites, one can always solve any tight problem instance using $\mathcal{A}_{\text{greedy}}$: assign the first and last floodlights (the ones parallel to the wedge boundaries; break ties arbitrarily) to the movable sites, move these sites back and inside the reverse wedge far enough so that every other site is within the reverse of the residual wedge not yet illuminated, and now use Fact 9.

Lemma 10. *For any $\varepsilon > 0$, an ε-angle approximation to the tight floodlight problem can be found efficiently.*

Proof. An ε-angle approximation can be achieved by temporarily adding two new movable sites p_a and p_b, adding two new floodlights f_a and f_b of angle $\varepsilon/2$ each, and proceeding as follows. Mount floodlight f_a at site p_a, orient it so that its upper boundary is parallel to and illuminates the upper boundary w_u of W, and move it far and low enough in W^r so that all other sites are above its lower boundary. Perform a similar operation on f_b and p_b starting with the lower boundary w_l of W. The region W' of W not illuminated by these two floodlights is a generalized wedge of angle $\alpha - \varepsilon$, where α is the wedge angle of W. Further, all remaining sites are contained inside W'^r. By Fact 9, we can illuminate W' completely using the remaining floodlights. Now remove f_a and f_b. Note that this removal can affect at most two wedge-shaped regions of W of angle $\varepsilon/2$. We are therefore left with a valid ε-angle approximation of the illumination of W using the original floodlights and sites. □

Note that an ε-angle approximate illumination only requires all but an ε angle of the wedge to be illuminated "at infinity". It would be interesting to design, if possible, an algorithm for the stronger approximation where the resulting illuminated area is a smaller wedge but located at the same apex as W.

4 NP-Completeness of Monotone Matching

In this section we describe our main construction — a polynomial time reduction from 3-Sat to Uniform Monotone Matching. 3-Sat is well-known to be NP-complete [10, 4]. For concreteness we define it as follows.

Definition 11. The 3-Sat Problem:
Instance: m clauses, each of which is a set of three distinct elements ("literals") chosen from a set of n Boolean variables and assigned a sign, positive or negative.
Question: Is there a True-False assignment to the variables such that each clause contains at least one positively occurring True variable or one negatively occurring False variable?

4.1 Notation and Overview

In the following we will refer to clauses by their index j and will denote the variables by $z_1, \ldots, z_n$. If the i^{th} variable appears in the j^{th} clause positively, we write that z_i occurs in clause j. If the variable appears negatively, we write that $\overline{z_i}$ occurs in clause j.

The Lines. The reduction uses many lines which we group into four categories.

(a) Lines labeled pos_{ij} correspond to a positive occurrence of the variable z_i in clause j. (For regularity, we include pos_{ij} in the construction even when z_i does not appear in clause j.) For convenience, we will use pos_{i*} to denote the set of lines pos_{ij} for $j \in [m]$.
(b) Lines labeled neg_{ij} correspond to the occurrence of $\overline{z}_i$ in clause j. (Again, we include neg_{ij} even when $\overline{z}_i$ does not appear in clause j.) For convenience, we will use neg_{i*} to denote the set of lines neg_{ij} for $j \in [m]$.
(c) Lines labeled aux_{ik} are used as auxiliary lines in one of the gadgets which we refer to as the "variable gadget". They will help ensure that within the i^{th} variable gadget, either all lines pos_{i*} or all lines neg_{i*} will be used.
(d) Lines labeled up_i and $down_i$ are used to ensure that on certain slabs, the monotone matching exits at or below a certain point or enters at or above a certain point. For example, we could ensure that the left endpoint of the segment taken in the tenth slab is at or above $y = 0$. (We will see more on this later.)

The Slabs. We use several unit width slabs grouped into four categories.

(a) We call the first $2mn + 5n$ slabs the *variable phase*. The variable phase actually consists of n variable gadgets, one for each variable. In the i^{th} variable gadget, we will ensure that either all pos_{i*} will be used by subsequent "clause" gadgets or all neg_{i*} will be used. Intuitively, this corresponds to setting x_i to False or to True, respectively.

(b) The next $50m^3n^2$ slabs will be referred to as the *buffer phase*. Although nothing interesting happens in the buffer phase, we need it to ensure that the slopes of the lines pos_{ij} and neg_{ij} are not too large.

(c) The next $3m$ slabs are called the *clause phase*. The clause phase consists of m gadgets, which we call "clause gadgets". As an example, suppose that the j^{th} clause is $z_1 \vee \overline{z}_2 \vee z_3$. Then the j^{th} clause gadget will ensure that the only way to cross one of the slabs is by using a segment from pos_{1j}, neg_{2j}, or pos_{3j}. More specifically, if z_1 and z_3 are set to False and z_2 is set to True, then the lines pos_{1j}, pos_{3j}, and neg_{2j} will have been used already in the variable phase. Otherwise, at least one the those segments will be available.

(d) The final phase is the *cleanup phase* which consists of $m(n-1)$ slabs. Since the monotone matching may have a choice of what segment to use in the clause phase, we need to ensure that segments from all lines are taken at some point between the first and the last slab. The cleanup phase facilitates this.

4.2 Description of the Gadgets

We utilize three kinds of gadgets: *variable gadgets*, *clause gadgets*, and *forcing gadgets*. As we described briefly in the previous section, the i^{th} variable gadget is used to enforce the condition that either all of the pos_{i*} are used up in the variable phase (corresponding to z_i being set to False), or all of the neg_{i*} are used up in the variable phase (corresponding to z_i being set to True). In the j^{th} clause gadget, the monotone matching will be able to cross a particular slab only if at least one of the lines corresponding to the literals in the j^{th} clause has not previously been used in the variable phase. (This happens only if the literal corresponding to that line is True.)

Finally, as part of the problem input, we specify that the monotone matching must enter at or above a certain point on the first slab and leave at or below a certain point on the final slab. For the reduction, it will be useful for us to make similar specifications for several other slabs as well in the monotone matching instance. For example, we may wish to specify that the monotone matching must enter at or above $y = 0$ on the m^{th} slab. The forcing gadget allows us to do this.

In each of the following gadget descriptions, we will avoid specifying the exact coordinates for the lines and slabs, instead describing only the necessary conditions the gadget must obey. The exact coordinates will be specified in Section 4.3.

The Variable Gadget. Throughout our description, fix $i \in [n]$. As we stated before, the i^{th} variable gadget is used to ensure that either all pos_{i*} are used or all neg_{i*} are used. We construct aux_{ik} for $k = 0, \ldots, m+2$ so that this is guaranteed in any valid matching. It may be helpful to refer to Figure 3 during the discussion.

For now, let x_i be a non-negative integer, and let $y_i \in \mathbb{R}$. (We will specify the values of x_i, y_i in the next section.) The i^{th} variable gadget uses $2m+3$ slabs, each of width 1, stretching from $x = x_i$ to $x = x_i + 2m + 3$. We will use a forcing gadget (described shortly) to ensure that any valid monotone matching for the x_i^{th} slab starts at or above $y = y_i + 1$ and ends at or below $y = y_i$.

The lines aux_{ik} for $k = 0, 1, \ldots, m{+}2$, and pos_{i*} and neg_{i*} satisfy the following conditions. See Figure 3 for a diagram.

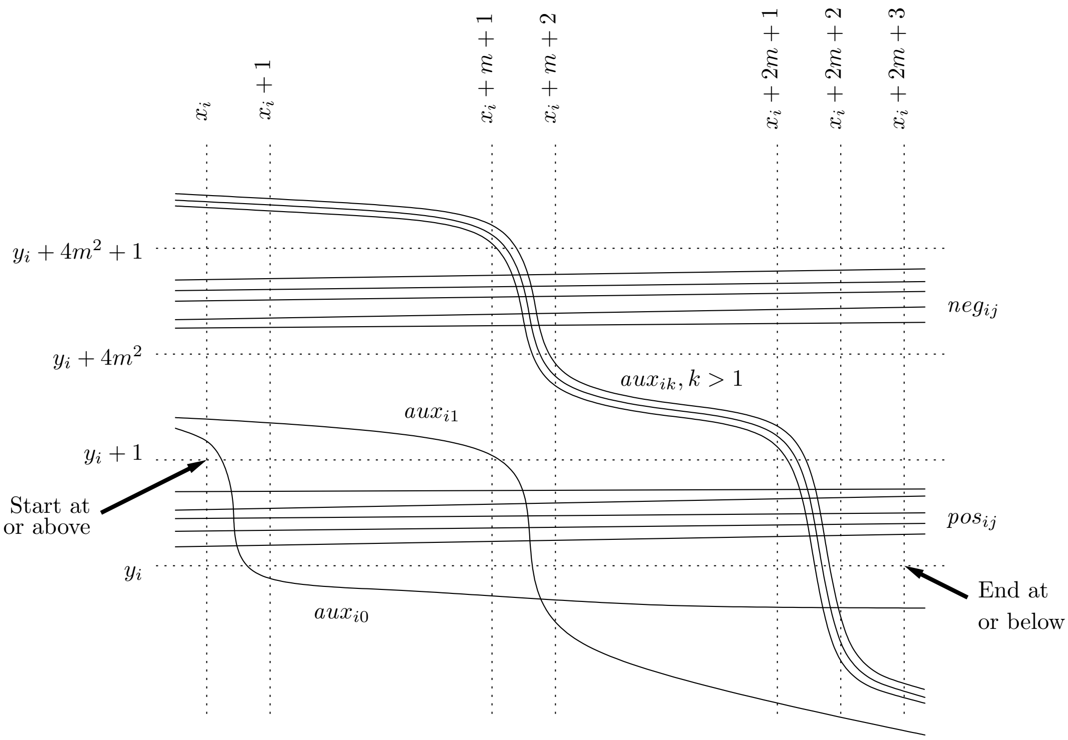

Fig. 3. The variable gadget for z_i. Note the figure is not to scale; in particular, the aux lines have been deformed due to non-uniform shrinking along the y-axis.

(1) The lines pos_{i*} do not intersect each other in the i^{th} variable gadget (that is, between $x = x_i$ and $x = x_i + 2m + 3$). Further, they all lie between $y = y_i$ and $y = y_i + 1$ in the i^{th} variable gadget.
(2) The lines neg_{i*} do not intersect each other in the i^{th} variable gadget (that is, between $x = x_i$ and $x = x_i + 2m + 3$). Further, they all lie between

$y = y_i + 4m^2$ and $y = y_i + 4m^2 + 1$ in the i^{th} variable gadget.

(3) At $x = x_i$, the line aux_{i0} lies at or above $y = y_i + 1$. From $x = x_i + 1$ to $x = x_i + 2m + 3$, the line aux_{i0} lies at or below $y = y_i$.

(4) From $x = x_i$ to $x = x_i + m + 1$, the line aux_{i1} lies at or above $y = y_i + 1$ and above the line aux_{i0}. At $x = x_i + m + 2$, the line aux_{i1} lies below aux_{i0} (and remains below aux_{i0} throughout the rest of the slabs).

(5) The lines aux_{ik} are parallel for all $k = 2, 3, \ldots, m+2$ and aux_{ik} is below $aux_{ik'}$ for $k < k'$. Throughout the i^{th} variable gadget, the line aux_{i1} lies below aux_{ik} for $k > 1$. From $x = x_i$ to $x = x_i + m + 1$, the lines aux_{ik} for $k > 1$ lie at or above $y = y_i + 4m^2 + 1$. From $x = x_i + m + 2$ to $x = x_i + 2m + 1$, the lines aux_{ik} for $k > 1$ lie between $y = y_i + 1$ and $y = y_i + 4m^2$. At $x = x_i + 2m + 2$, the lines aux_{ik} for $k > 1$ lie below aux_{i0}.

Given these conditions, we have the following lemma. Let y^i_{max} denote the y-coordinate of $aux_{i,m+2}$ at $x = x_i$ and y^i_{min} denote the y-coordinate of aux_{i1} at $x = x_i + 2m + 3$.

Lemma 12. *Suppose that pos_{i*}, neg_{i*}, and aux_{ik} for $k = 0, \ldots, m+2$ satisfy the previous conditions. Further, suppose that all other lines in the construction either lie above $y = y^i_{max}$ or below $y = y^i_{min}$ throughout the i^{th} variable gadget. For any valid monotone matching that begins at or above $y = y_i + 1$ at $x = x_i$ and ends at or below $y = y_i$ at $x = x_i + 2m + 3$,*

(a) all of the lines aux_{ik} for $k = 0, 1, \ldots, m+2$ are used in the i^{th} variable gadget, and

(b) either all of the lines pos_{i*} or all of the lines neg_{i*} are used in the i^{th} variable gadget (but not both).

Proof. First of all, we cannot take any line other than pos_{i*}, neg_{i*}, or aux_{ik} for $k = 0, \ldots, m+2$ in the i^{th} variable gadget, since we cannot reach any line lying below $y = y^i_{min}$ for the entire gadget and taking any line that lies above $y = y^i_{max}$ makes it impossible to finish below $y = y_i$ in the last slab of the gadget. Hence, throughout this proof, we only need to consider pos_{i*}, neg_{i*}, and aux_{ik}.

We have $2m+3$ slabs and $3m+3$ active lines in the gadget. Out of these, there are only 3 slabs in which the monotone matching may start above $y = y_i$ and end below $y = y_i$: the slab from $x = x_i$ to $x = x_i+1$, the slab from $x = x_i+m+1$ to $x = x_i + m + 2$, and the slab from $x = x_i + 2m + 1$ to $x = x_i + 2m + 2$.

Case i. Suppose that the monotone matching does not go from above $y = y_i$ to below $y = y_i$ in the slab from $x = x_i$ to $x = x_i + 1$. Then from $x = x_i$ to $x = x_i + m + 1$, the monotone matching cannot use any of the pos_{i*}. It follows that the matching cannot use the line aux_{i1} in the slab from

$x = x_i + m + 1$ to $x = x_i + m + 2$ since that line is too low. Hence, the first time it passes below $y = y_i$ is at the slab from $x = x_i + 2m + 1$ to $x = x_i + 2m + 2$. At this point, it is too late to use any pos_{i*}, since they all end too high at $x = x_i + 2m + 3$. Hence, the monotone matching cannot use any of the pos_{ij}. (Note that the line aux_{i0} may instead be used.) Since there are precisely $2m + 3$ slabs and precisely $2m + 3$ available lines, each of these must be used. That is, each of the neg_{i*} must be used.

Case ii. Suppose instead that the monotone matching does go from above $y = y_i$ to below $y = y_i$ in the slab from $x = x_i$ to $x = x_i + 1$. Specifically, the monotone matching must use aux_{i0} in this slab. We will prove by contradiction that the monotone matching never uses any neg_{ij} in the i^{th} variable gadget. To this end, suppose that the monotone matching does use some neg_{ij}.

First of all, note that the monotone matching cannot use any neg_{i*} for $x \geq x_i + m + 1$; the reason for this is that from this point on, there is no way to get lower than $y = y_i + 4m^2$ if we do use some neg_{i*}. Therefore, the monotone matching must have used neg_{i*} before $x = x_i + m + 1$. This implies that at $x = x_i + m + 1$, the monotone matching must use aux_{ik} for some $k > 1$. Furthermore, since the aux_{ik} for $k > 1$ lie above aux_{i0},aux_{i1}, and pos_{ij} from $x = x_i + m + 1$ to $x = x_i + 2m + 2$, the monotone matching must use only the aux_{ik} for $k > 1$ from $x = x_i + m + 1$ to $x = x_i + 2m + 2$. But now the monotone matching is out of options. Since there are precisely $m + 1$ such lines aux_{ik} for $k = 2, \ldots, m + 2$ and the matching traversed precisely $m + 1$ slabs, it must have used every such aux_{ik}. It follows that the only available lines on the slab from $x = x_i + 2m + 2$ to $x = x_i + 2m + 3$ are aux_{i0}, pos_{i*}, and neg_{i*}. The lines pos_{i*}, neg_{i*} all end too high and, unlike case (i), aux_{i0} has already been used. Hence, we have a contradiction.

It follows that if the monotone matching uses aux_{i0} in the slab from $x = x_i$ to $x = x_i + 1$, then it never uses neg_{ij} for any $j = 1, \ldots, m$. Since the matching traverses precisely $2m + 3$ slabs, and there are precisely $2m + 3$ lines available, it must use each of these lines precisely once. In particular, it must use all of pos_{i*}.

This finishes the proof. □

The Clause Gadget. The clause gadget is rather simple. Fix $j \in [m]$. Let x'_j, y'_j be positive integers, whose precise values will be specified in the next section.

If the j^{th} clause is $z_{i_1} \vee z_{i_2} \vee z_{i_3}$, then we construct our lines so that $pos_{i_1 j}$, $pos_{i_2 j}$, and $pos_{i_3 j}$ all lie above $y = y'_j$ and at or below $y = y'_j + 1$ from $x = x'_j$ to $x = x'_j + 1$, and no other lines lie between these y values in that slab. Similarly, if the j^{th} clause is $\overline{z}_{i_1} \vee z_{i_2} \vee z_{i_3}$, then we construct our lines so that $neg_{i_1 j}$, $pos_{i_2 j}$, and $pos_{i_3 j}$ all lie above $y = y'_j$ and at or below $y = y'_j + 1$ from $x = x'_j$ to $x = x'_j + 1$, and no other lines lie between these y values in that slab. We define the clause gadget analogously for the remaining 6 cases. See Figure 4 for an example with clause $z_1 \vee \overline{z}_2 \vee z_3$.

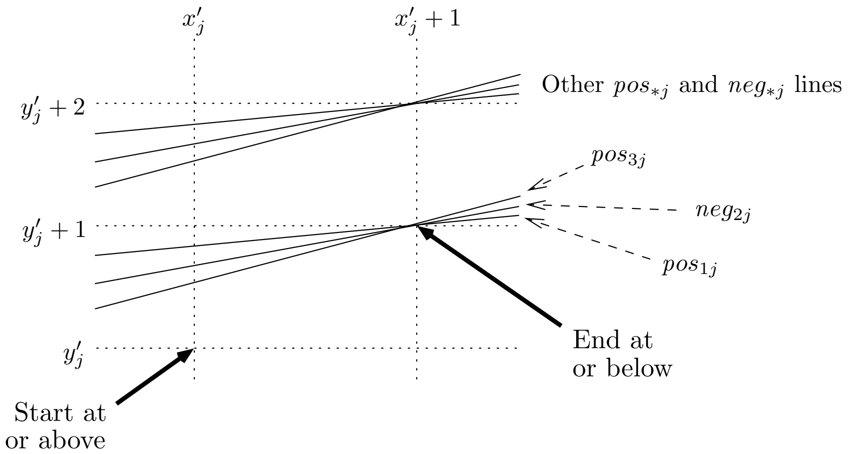

Fig. 4. The clause gadget for clause j, $(z_1 \vee \overline{z_2} \vee z_3)$. For clarity, the lines are shown to be much steeper than they will be in the final construction. *pos* and *neg* lines from other clauses would appear above or below this figure.

Using a forcing gadget, we ensure that any valid monotone matching starts at or above $y = y'_j$ at $x = x'_j$, and ends at or below $y = y'_j + 1$ at $x = x'_j + 1$. From this, the following result follows immediately.

Lemma 13. *Let ℓ_1, ℓ_2, ℓ_3 be the lines corresponding to the literals in the j^{th} clause, as specified above. Then there can be a valid monotone matching only if at least one of ℓ_1, ℓ_2, ℓ_3 has not already been used in the variable phase.*

The Forcing Gadget. Let $s = (2m+5)n + 50m^3n^2 + 3m + m(n-1)$ denote the total number of slabs. Let $h > 0$ be such that the lines pos_{i*}, neg_{i*}, and aux_{ik} all remain strictly between $y = -h$ and $y = h$ throughout the s slabs. Such an h exists as there are no vertical lines in the construction; with the choice of parameters used in the next section, $h = 240m^4n^2$ is sufficient.

Let $e_1, \ldots, e_\ell$ and $s_1, \ldots, s_\ell$ be sequences of numbers so that $-h < e_i < h$ and $-h < s_i < h$ for all $i \in [\ell]$. Further, let $x''_1, x''_2, \ldots, x''_\ell$ be a sequence of positive integers such that $x''_{i+1} \geq x''_i + 2$ for all $i \in [\ell - 1]$.

We can force any valid matching to end at or below $y = e_i$ at $x = x''_i$ and start at or above $y = s_i$ at $x = x''_i + 2$, in the following way. The construction is shown in Figure 5.

For each $i \in [\ell]$, construct line $down_i$ so that it passes through the point (x''_i, e_i) and has slope $-6h$, and construct line up_i so that it passes through the point $(x''_i + 2, s_i)$ and has slope $4h$. The pairs $(down_i, up_i)$ will be referred to as the *forcing gadgets*.

Lemma 14. *For all $i \in [\ell]$, let lines $down_i$ and up_i be constructed as described*

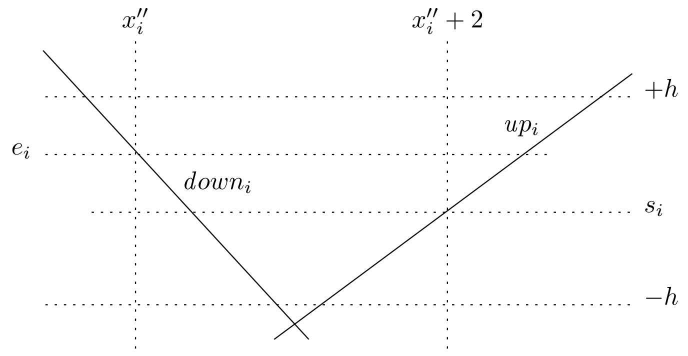

Fig. 5. The forcing gadget. It is not drawn to scale; the lines up_i and $down_i$ should be much steeper.

above. Then any valid monotone matching will use $down_i$ on the slab from x_i'' to $x_i'' + 1$, and will use up_i on the slab from $x_i'' + 1$ to $x_i'' + 2$.

Proof. By the construction of the *up* and *down* lines, it is enough to show that no such line is used in a slab where it is either completely below $-h$, or has any point above h.

We first show that no line is used completely below $-h$. Suppose for contradiction sake that this first occurs in a slab s. Clearly, s cannot be the left-most slab. Let t be the slab previous to s. The line used in t must be a *down* line as it does not lie completely below $-h$ in t. Moreover, if y_1 is the y-coordinate of its intersection with the right side of t, $y_1 > -h - 6h = -7h$. Suppose the line used in s were an *up* line. Let $y_2 < -h$ be the y-coordinate of the *up* line used in s with the right side of s, and let y_3 be the y-coordinate of that same line with the right side of the slab following s. Then $y_1 > -7h$ implies $y_2 > -7h + 4h = -3h$. Hence $y_3 > -3h + 4h > h$. However, by construction, no *up* line begins below $-h$ and ends above h across a single slab.

Now suppose the line used in s were a *down* line, say $down_i$. Let $down_j$ be the *down* line used in t, so that $j < i$. As $down_j$ straddles $-h$ in t, then $down_i$ straddles $-h$ in a slab at or after s, contradicting our assumption that $down_i$ lies completely below $-h$ in s.

We now show no *up* or *down* line is used in a slab where it has any point above h. Suppose for contradiction s is the last slab where this happens. If the line used in s is ends above h, then the line used in the following slab must start above h as well, which cannot happen as only *up* or *down* lines reach that high and we have assumed s is the last slab where an *up* or *down* line is used above h. Therefore the line used in s is a *down* line, say $down_i$, which straddles h. Then s must be the slab preceding x_i''. Consider up_i. By previous arguments, up_i must be used in its correct slab r, two slabs from s. But by construction, up_i starts below $-h$ in r, and there are no *down* lines available in the correct place in the slab between s and r. □

4.3 Putting it Together

We now describe, as part of the proof of the following theorem, how these gadgets are put together in order to construct a reduction from the propositional satisfiability problem. Refer to Figure 6 for a pictorial overview.

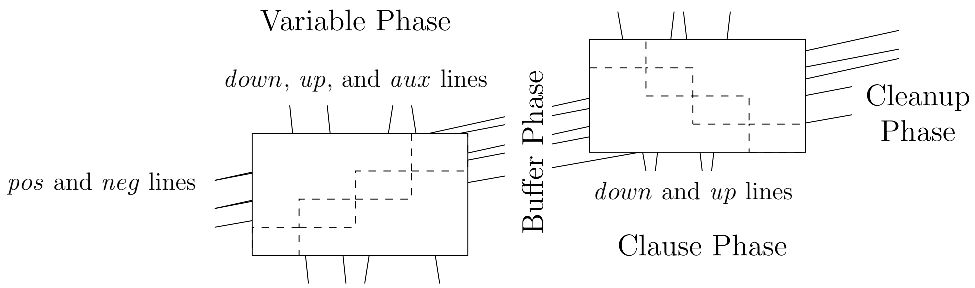

Fig. 6. The overall picture, not to scale. The smaller boxes in the variable and clause phases depict the arrangement of the variable and clause gadgets, respectively.

Theorem 15. UNIFORM MONOTONE MATCHING *is NP-complete*

Proof. Membership in NP follows from the fact that a potential matching can be validated in polynomial time. We now specify the exact coordinates of the construction of Section 4.2 and show that the construction in Section 4.2 has a monotone matching if and only if the corresponding formula has a satisfying assignment.

Following the notation of the previous section, set $x_i = (i-1)(2m+5)$ and $y_i = -10m^2n + 10m^2(i-1)$. The i^{th} variable gadget in the **variable phase** is arranged from $x = x_i$ to $x = x_i + 2m + 3$, and will lie between $y = y_i$ and $y = y_i + 10m^2$. After each variable gadget, we add a forcing gadget, taking 2 slabs, to ensure that any valid monotone matching ends at or below $y = y_i$ at $x = x_i + 2m + 3$ and starts at or above $y = y_i + 10m^2 + 1 = y_{i+1} + 1$ at $x = x_i + 2m + 5 = x_{i+1}$.

Following again the notation of the previous section, set $x'_j = 50m^3n^2 + (2m+5)n + 3(j-1)$ and $y'_j = 2m - 2j$. The j^{th} clause gadget in the **clause phase** spans the space from $x = x'_j$ to $x = x'_j + 1$. After each clause gadget, we add a forcing gadget, taking 2 slabs, to ensure that any valid monotone matching ends at or below $y = y'_j + 1$ at $x = x'_j + 1$ and starts at or above $y = y'_j - 2 = y'_{j+1}$ at $x = x'_j + 3 = x'_{j+1}$.

The region between the last variable gadget and the first clause gadget forms the **buffer phase**.

Assume that $m \geq 5$. We specify the lines as follows:

- **The *pos* lines**. Let $i \in [n]$ and $j \in [m]$. If z_i appears in the j^{th} clause as a positive literal, define pos_{ij} as the unique line that goes through the points $(-1, y_i)$ and $(x'_j + 1, y'_j + 1)$. If z_i does not appear in the j^{th} clause as a positive literal, define pos_{ij} as the unique line that goes through the points $(-1, y_i)$ and $(x'_j + 1, y'_j + 2)$. Notice that in either case, the size $50m^3n^2$ of the buffer phase is large enough so that for $m \geq 5$ the slope of pos_{ij} is less than $1/(3mn)$. This implies that at the end of the variable phase, pos_{ij} is below $y_i + 1$. Hence, pos_{ij} satisfies the first condition for the i^{th} variable gadget. Furthermore, pos_{ij} lies between $y = y'_j$ and $y = y'_j + 1$ for $x = x'_j$ to $x = x'_j + 1$ if and only if z_i appears in the j^{th} clause as a positive literal.
- **The *neg* lines**. Let $i \in [n]$ and $j \in [m]$. If $\overline{z}_i$ appears in the j^{th} clause as a negative literal, define neg_{ij} as the unique line that goes through the points $(-1, y_i + 4m^2)$ and $(x'_j + 1, y'_j + 1)$. If $\overline{z}_i$ does not appear in the j^{th} clause as a negative literal, define neg_{ij} as the unique line that goes through the points $(-1, y_i + 4m^2)$ and $(x'_j + 1, y'_j + 2)$. Notice again that in either case, the slope of neg_{ij} is less than $1/(3mn)$ which implies that at the end of the variable phase, neg_{ij} lies below $y_i + 4m^2 + 1$. Hence, neg_{ij} satisfies the second condition for the i^{th} variable gadget. Furthermore, neg_{ij} lies between $y = y'_j$ and $y = y'_j + 1$ for $x = x'_j$ to $x = x'_j + 1$ if and only if $\overline{z}_i$ appears in the j^{th} clause as a negative literal.
- **The *aux* lines**. For each $i \in [n]$, define aux_{i0} to be the unique line of slope -1 passing through the points (x_i, y_i+1), (x_i+1, y_i), (x_i+m+2, y_i-m-1), and (x_i+2m+2, y_i-2m-1). Notice that aux_{i0} satisfies the third condition for the i^{th} variable gadget.

  For each $i \in [n]$, define aux_{i1} to be the unique line of slope $-2m$ passing through the points $(x_i, y_i + 2m^2 + 2m + 1)$, $(x_i + m + 1, y_i + 1)$, $(x_i + m + 2, y_i - 2m + 1)$, and $(x_i + 2m + 3, y_i - 2m^2 - 4m + 1)$. Notice that for $m \geq 3$, aux_{i1} satisfies the fourth condition for the i^{th} variable gadget. This choice of parameters has $y^i_{min} = y_i - 2m^2 - 4m + 1$.

  For each $i \in [n]$ and $k = 2, \ldots, m + 2$, define aux_{ik} to be the unique line of slope $-4m$ passing through the points $(x_i, y_i + 8m^2 + 4m + k - 1)$, $(x_i + m + 1, y_i + 4m^2 + k - 1)$, $(x_i + m + 2, y_i + 4m^2 - 4m + k - 1)$, $(x_i + 2m + 1, y_i + k - 1)$, and $(x_i + 2m + 2, y_i - 4m + k - 1)$. It is not hard to verify that for $m \geq 5$, the aux_{ik} for $k > 1$ satisfy the fifth condition for the i^{th} variable gadget. With these parameters, $y^i_{max} = y_i + 8m^2 + 5m + 1$.
- **The *up* and *down* lines**. Let $h = 240m^4n^2$ and $s = (2m+5)n + 50m^3n^2 + 3m + m(n-1)$ as defined earlier be the total number of slabs in the construction. We now describe the parameters for the forcing gadgets. First, in the variable phase, for $i \in [n]$, let $down_i$ be the unique line with slope $-6h$ that passes through the point (x_i+2m+3, y_i), and let up_i be the unique line with slope $4h$ that passes through the point $(x_i + 2m + 5, y_i + 10m^2 + 1)$. In the notation of the previous section, $x''_i = x_i + 2m + 3$, $e_i = y_i$, and $s_i = y_i + 10m^2 + 1$. In the buffer phase, for $i = n + 1, \ldots, n + 25m^3n^2$, let $down_i$ be the unique line with slope $-6h$ that passes through the point $(x_n + 2m - 2n + 2i + 3, 0)$, and let up_i be the unique line with slope $4h$ that

passes through the point $(x_n+2m-2n+2i+5, y_1')$. That is, in the notation of the previous section, $x_i'' = x_n + 2m - 2n + 2i + 3$, $e_i = 0$, and $s_i = y_1'$. Finally, in the clause phase, for $j = 1, \ldots, m$, let $down_{n+25m^3n^2+j}$ be the unique line with slope $-6h$ that passes through the point $(x_j' + 1, y_j' + 1)$, and let $up_{n+25m^3n^2+j}$ be the unique line with slope $4h$ that passes through the point $(x_j' + 3, y_j' - 2)$. That is, $x''_{n+25m^3n^2+j} = x_j' + 1$, $e_i = y_j' + 1$, and $s_i = y_j' - 2$.

To finish the description of the monotone matching problem, we set the starting point START to be $y = y_1 + 1$ and the ending point END to be $y = 240m^4n^2$.

We see from the descriptions of the lines that each of them operates within the relevant gadgets in the appropriate ways. However, we also need to check that lines interact only with the correct gadgets and not elsewhere.

To this end, first notice that both pos_{ij} and neg_{ij} have positive slopes less than $1/(3mn)$. Hence, throughout the entire variable phase, each of these lines rises less than 1. Hence, each pos_{ij} remains between $y = y_i$ and $y = y_i + 1$ throughout the variable phase, and likewise, each neg_{ij} remains between $y = y_1 + 4m^2$ and $y = y_i + 4m^2 + 1$ throughout the variable phase. Therefore, the pos_{ij} and neg_{ij} only interact with their corresponding variable gadget. Furthermore, by our construction, if $\hat{\jmath} < j$ then $pos_{i\hat{\jmath}}$ and $neg_{i\hat{\jmath}}$ are both strictly above $y = y_j' + 2$ from $x = x_j'$ to $x_j' + 1$. Likewise, if $\hat{\jmath} > j$, then $pos_{i\hat{\jmath}}$ and $neg_{i\hat{\jmath}}$ are both strictly below $y = y_j'$ from $x = x_j'$ to $x_j' + 1$. Hence, the pos_{ij} and neg_{ij} do not violate the conditions of the clause gadget.

Second, notice that each of the aux_{ik} for $k = 0, 1, \ldots, m + 2$ have negative slopes. Thus at x_{i+1}, aux_{i0} lies below $y_i < y_{min}^{i+1} = y_{i+1} - 2m^2 - 4m + 1 = y_i + 8m^2 - 4m + 1$. Since all other auxiliary lines for the i^{th} gadget are below aux_{i0}, the auxiliary lines at variable gadget i do not interact with variable gadget i' for $i' > i$. Similarly, because of our assumption that $m \geq 5$, for $x < x_i$, aux_{i0} (and hence all auxiliary lines for the i^{th} gadget) are above $y_i > y_{max}^{i-1} = y_{i-1} + 8m^2 + 5m + 1 = y_i - 2m^2 - 5m - 1$. Hence, aux_{ik} only interacts with the i^{th} variable gadget.

Third, notice that the lines pos_{ij}, neg_{ij}, and aux_{ik} all lie between $y = -240m^4n^2$ and $y = 240m^4n^2$ for all slabs, since (a) they all pass through the region between $y = y_i$ and $y = y_i + 4n^2 + 1$ (which is relatively close to $y = 0$) in their respective variable gadgets, (b) the maximum positive slope for all of these lines is less than 1, (c) the most negative slope for all of these lines is $-4m$, and (d) the total number of slabs s is less than $60m^3n^2$ for $m \geq 5$. Hence our choice of $h = 240m^4n^2$ suffices.

Hence, the conditions for Lemmas 12, 13, and 14 hold for our construction. We now prove that there is a valid monotone matching in this construction iff the original formula is satisfiable.

Part i. Suppose that there is a valid monotone matching. For each $i \in [n]$, set z_i to False if all of the pos_{ij} are used in the i^{th} variable gadget, and set z_i to True if all of the neg_{ij} are used in the i^{th} variable gadget. By Lemma 12, exactly one of the two conditions must occur. Let ℓ_1, ℓ_2, ℓ_3 be the lines corresponding to the literals appearing in the j^{th} clause. By Lemma 13, at least one of ℓ_1, ℓ_2, ℓ_3 was not used in the variable phase. But this means that the corresponding literal is True. Hence, each clause is satisfied and we have a satisfying assignment.

Part ii. Suppose that there is a satisfying assignment. We will argue that there is a valid monotone matching. If z_i is False in the satisfying assignment, then for the i^{th} variable gadget, take aux_{i0}, followed by pos_{ij} for $j = 1, \ldots, m$ in order. (By construction, we see that the pos_{i*} do not intersect in the i^{th} variable gadget, and in fact pos_{ij} lies below $pos_{i\hat{j}}$ for $j < \hat{j}$.) Then take the line aux_{i1}, followed by aux_{ik} for $k > 1$ in order. We thus arrive below $y = y_i$ when $x = x_i + 2m + 3$.

If z_i is True in the satisfying assignment, then for the i^{th} variable gadget, take aux_{i1}, followed by neg_{ij} for $j = 1, \ldots, m$ in order. (We use the fact that neg_{ij} lies below $neg_{i\hat{j}}$ for $j < \hat{j}$ in the i^{th} variable gadget.) Then take the lines aux_{ik} for $k > 1$ in order. We thus arrive below the line aux_{i0} when $x = x_i + 2n + 2$. Therefore, in the final slab of the gadget, we may take line aux_{i0}, arriving below $y = y_i$.

For each $i = 1, \ldots, n + 25m^3n^2 + m$, for the slab from $x = x''_i$ to $x = x''_i + 1$, take $down_i$, and for the slab from $x = x''_i + 1$ to $x = x'' + 2$, take up_i.

For each $j \in [m]$, at least one literal in clause j is set to True in the satisfying assignment. Hence, if ℓ_1, ℓ_2, ℓ_3 are the lines corresponding to the literals in the j^{th} clause, at least one of ℓ_1, ℓ_2, ℓ_3 will not have been used in the variable phase. Take such a line in the slab from $x = x'_j$ to $x = x'_j + 1$; if there is more than one line available, take the one that is first lexicographically.

Finally, the cleanup phase lasts for $m(n-1)$ slabs, and there are exactly $m(n-1)$ unused lines. Further, at the start of the cleanup phase, we are forced to start at or above $y = y'_m - 2 < 0$, and all the unused lines (which consist of lines from pos_{ij} and neg_{ij}) lie strictly above $y = 0$. We can greedily traverse the entire cleanup phase starting with the unused line that ends the lowest at the end of the current slab, and continuing until we reach the end of the last slab. Since all of the pos_{ij},neg_{ij} lie below $y = 240m^4n^2$ and the END is at $y = 240m^4n^2$, we have a valid monotone matching.

Hence we have reduced 3-Sat to Uniform Monotone Matching, proving the theorem. □

A (trivial) reduction from Uniform Monotone Matching to Monotone Matching implies the following:

Corollary 16. Monotone Matching *is NP-complete.*

By the discussion in Section 2.2, the above also proves Theorem 2.

5 Conclusion

In this paper, we provided an affirmative answer to the question posed by Demaine and O'Rourke [6], by proving that floodlight illumination is NP-hard. The proof was based on a reduction from 3-SAT to the equivalent problem of monotone matching. The key building block for the reduction was the *forcing gadget*, which was used to simulate the behavior of individual variables and clauses of a 3-CNF formula using several variable and clause gadgets. Our proof also shows that floodlight illumination is NP-complete in the uniform and tight case. (It remains open whether floodlight illumination, in its generality, is in NP or not.) Finally, we showed that computing finite-region approximations for the problem is NP-hard, while finite-angle approximations can be computed in polynomial time.

Acknowledgements

The authors would like thank Neva Cherniavsky for introducing the problem, Gidon Shavit and William Pentney for helpful discussions, and the anonymous reviewers for helpful comments.

References

[1] J. Abello, V. Estivill-Castro, T. C. Shermer, and J. Urrutia. Illumination of orthogonal polygons with orthogonal floodlights. *International Journal of Computational Geometry and Applications*, 8(1):25–38, 1998.

[2] J. Bagga, L. Gewali, and D. Glasser. The complexity of illumination polygons by α-flood-lights. In *Proceedings of the 8th Canadian Conference on Computational Geometry*, pages 337–342, Ottawa, Canada, Aug. 1996.

[3] P. Bose, L. Guibas, A. Lubiw, M. Overmars, D. Souvaine, and J. Urrutia. The floodlight problem. *International Journal of Computational Geometry and Applications*, 7(1/2):153–163, 1997.

[4] S. A. Cook. The complexity of theorem proving procedures. *Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[5] J. Czyzowicz, E. Rivera-Campo, and J. Urrutia. Optimal floodlight illumination of stages. In *Proceedings of the 5th Canadian Conference on Computational Geometry*, pages 393–398, Waterloo, Canada, Aug. 1993.

[6] E. Demaine and J. O'Rourke. Open problems from CCCG 2001. In *Proceedings of the 13th Canadian Conference on Computational Geometry*, Waterloo, Canada, Aug. 2001.

[7] V. Estivill-Castro, J. O'Rourke, J. Urrutia, and D. Xu. Illumination of polygons with vertex lights. *Information Processing Letters*, 56(1):9–13, 1995.

[8] V. Estivill-Castro and J. Urrutia. Optimal floodlight illumination of orthogonal art galleries. In *Proceedings of the 6th Canadian Conference on Computational Geometry*, pages 81–86, Saskatchewan, Canada, Aug. 1994.

[9] V. Estivill-Castro and J. Urrutia. Two-floodlight illumination of convex polygons. In *Proceedings of the 4th International Workshop on Algorithms and Data Structures*, volume 955 of *LNCS*, pages 62–73, Kingston, Canada, Aug. 1995. Springer Verlag.

[10] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[11] D. Y. Grigor'ev. Complexity of deciding Tarski algebra. *Journal of Symbolic Computation*, 5:65–108, 1998.

[12] J. O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, 1987.

[13] T. C. Shermer. Recent results in art galleries. *Proceedings of the IEEE*, 80(9):1384–1399, 1992.

[14] B. Speckmann and C. D. Tóth. Allocating vertex π-guards in simple polygons via pseudo-triangulations. *Discrete & Computational Geometry*, 33(2):345–364, 2005.

[15] W. L. Steiger and I. Streinu. Illumination by floodlights. *Computational Geometry*, 10:57–70, 1998.

[16] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 1948.

[17] C. D. Tóth. Illuminating polygons with vertex π-floodlights. In *Proceedings, Part I, of the International Conference on Computational Science (ICCS)*, volume 2073 of *LNCS*, pages 772–784, San Francisco, CA, May 2001. Springer Verlag.