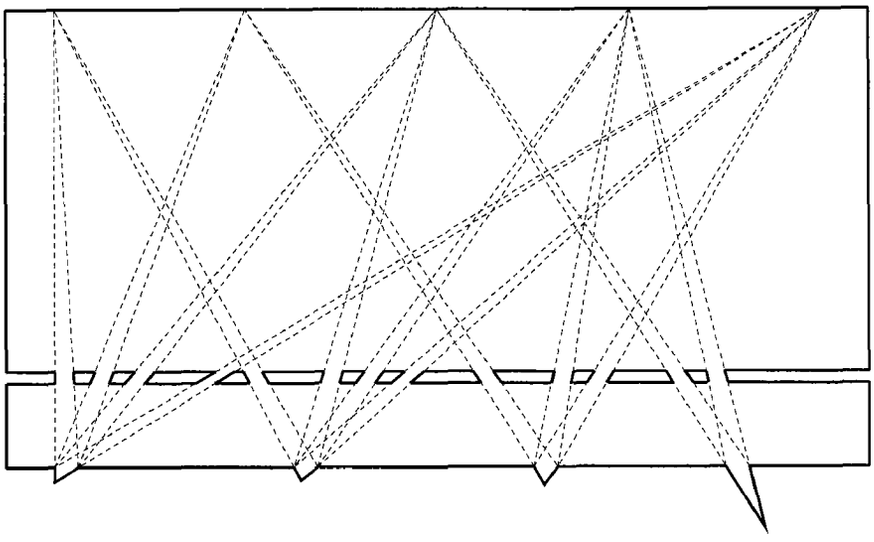
Diss. ETH No. 13683 2000

(In-)Approximability of Visibility Problems on Polygons and Terrains



Stephan Eidenbenz

Seite Leer /
Blank leaf

Diss. ETH No. 13683

(In-)Approximability of Visibility Problems on Polygons and Terrains

A dissertation submitted to the

SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH

for the degree of

Doctor of Technical Sciences

presented by

Stephan Johannes Eidenbenz
Dipl. Informatik-Ingenieur ETH Zürich
born August 25, 1973 in Zürich, ZH

accepted on the recommendation of

Prof. Dr. Peter Widmayer, Examiner
Prof. Dr. Dr. Jürgen Richter-Gebert, Co-Examiner

2000

Seite Leer /
Blank leaf

Acknowledgements

I would like to express my sincere gratitude to my advisor, Prof. Dr. Peter Widmayer, for supporting me and for being a true inspiration throughout my PhD studies. I am grateful to Prof. Dr. Dr. Jürgen Richter-Gebert for being my co-examiner. I want to thank the members of our research group, who have shared a small part of their lives with me, i.e., Mark Cieliebak, Barbara Heller, Philipp Kramer, Zsuzsa Lipták, Dr. Gabriele Neyer, Dr. Renato Pajarola, Peter Remmele, Dr. Thomas Roos, Konrad Schlude, Christoph Stamm, Roland Ulber, and Dr. Roger Wattenhofer, for all the fruitful discussions we have had. From outside our research group, I thank Christoph Ambühl for sharing many common interests.

I am very grateful to my parents and my family for helping me reach my goals. Most of all, I thank my fiancée Seraina for always supporting me by giving so much love. Finally, I would like to thank God for showing me that the beauty of His creation is sometimes hidden in the hardness of things.

Contents

Abstract

Visibility problems appear in a variety of applicative backgrounds. While the traditional "art gallery" problem, which consists of guarding a given floor plan of an art gallery by a minimum number of guards seems to be mainly of theoretical interest, a variation of this problem, where a 2.5 dimensional terrain rather than a two dimensional polygonal floor plan needs to be guarded, is of practical significance in the planning of wireless communication networks, where minimizing the numbers of antennas reduces overall costs.

In a visibility problem, we are given an input polygon, which may or may not contain holes, or a 2.5 dimensional triangulated terrain. We say that two points in the polygon or on or above the terrain see each other, if the straight line segment connecting the two points does not intersect the exterior of the polygon or the space below the terrain.

A first category of visibility problems is the problem of "guarding". We need to find a minimum number of guard positions, such that these guards collectively see the whole polygon or the terrain. Since this problem is NP-hard in most variations, the search for approximability as well as inapproximability results may prove helpful. We show that for input polygons with holes, no polynomial time approximation algorithm can achieve an approximation ratio that is logarithmic in the number of polygon vertices. This result is tight up to constant factors, since there exists a polynomial time algorithm that achieves a logarithmic approximation ratio. We obtain this result by construction a gap-preserving reduction from the MINIMUM SET COVER problem. This inapproximability result carries over to the problem of guarding terrains. We also propose an approximation algorithm for guarding terrains that achieves a logarithmic approximation ratio. The situation is less clear for input polygons without holes: we are only able to show APX-hardness (i.e. there exists a constant ϵ, such that no approximation algorithm can achieve an approximation ratio of $1+\epsilon$). We modify and analyze an already known reduction to obtain this result. Thus, for poly-

gon without holes, it is still open, whether there exists an approximation algorithm with a constant approximation ratio.

A second category of visibility problems is "hiding". In a hiding problem, we need to find a maximum number of positions in the input polygon or on the input terrain, such that no two of these positions see each other. We show that for input polygons with holes, as well as for terrains, no approximation algorithm can achieve an approximation ratio of n^ϵ for some constant $\epsilon > 0$. This result is obtained by a gap-preserving reduction from MAXIMUM CLIQUE. For input polygons without holes, we again can only show APX-hardness; thus a gap remains in this case.

A third category of visibility problems is "convex covering", where we need to cover an input polygon with a minimum number of (possibly overlapping) convex polygons that are inside the input polygon. We propose a polynomial time algorithm for this problem that achieves a logarithmic approximation ratio. The key ingredient in this algorithm is a successful discretization of the plane. As for inapproximability, we show APX-hardness; thus a gap remains.

In a fourth category of visibility problems, we only consider the corresponding visibility graphs as input structures. We show how to geometrically interpret some classic graph-theoretic problems, and show that some of our results for general visibility problems carry over. In particular, we show that for visibility graphs of polygons with holes, the clique problem is almost as hard to approximate as in general graphs, while it is known that clique can be solved in polynomial time if the poygons contain no holes.

Zusammenfassung

Sichtbarkeitsprobleme haben verschiedene praktische Anwendungen. Während das klassische "Kunstgallerie"-Problem, bei dem der Grundriss einer Kunstgallerie mit einer minimalen Anzahl Wächter bewacht werden soll, mehr von theoretischem Interesse ist, hat ein verwandtes Problem, bei dem ein 2.5 dimensionales Terrain anstelle einer Kunstgallerie überwacht werden soll, eine praktische Bedeutung bei der Planung von Mobilfunknetzwerken, wo möglichst wenige Antennen ein möglichst grosses Gebiet abdecken, oder eben überwachen sollen.

Ein Sichtbarkeitsproblem bekommt als Eingabe entweder ein Polygon mit oder ohne Löcher oder ein 2.5 dimensionales trianguliertes Terrain. Zwei Punkte im Polygon oder auf oder oberhalb des Terrains sehen sich gegenseitig, falls das Liniensegment, das die zwei Punkte verbindet, nicht das Äussere des Polygons bzw. den Raum unterhalb des Terrains schneidet.

Eine erste Gruppe von Sichtbarkeitsproblemen sind Überwachungsprobleme. Gefragt wird dabei immer nach einer minimalen Anzahl von Wächtern, die zusammen das ganze Polygon oder Terrain sehen. Weil dieses Problem in den meisten Varianten NP-schwer ist, ist es sinnvoll, Approximierbarkeits- und Nichtapproximierbarkeitsresultate für diese Probleme zu finden. Für Eingabepolygone mit Löchern zeigen wir, dass kein polynomieller Approximationsalgorithmus eine Approximationsgüte erreichen kann, welche logarithmisch in der Anzahl Polygonecken ist. Dieses Resultat ist optimal bis auf konstante Faktoren, da ein polynomieller Approximationsalgorithmus existiert, der eine logarithmische Approximationsgüte erreicht. Wir erhalten dieses Resultat durch eine lückenerhaltende Reduktion vom MINIMUM SET COVER Problem. Dieses Nichtapproximierbarkeitsresultat lässt sich leicht auf den Fall übertragen, in dem ein 2.5 dimensionales Terrain überwacht werden soll. Für Eingabepolygone ohne Löcher haben wir noch kein optimales Resultat: wir zeigen zwar, dass das entsprechende Problem APX-schwer ist (d.h., dass eine Konstante $\epsilon > 0$ existiert, so dass kein polynomieller Approximationsalgorithmus eine Approximationsgüte von $1 + \epsilon$

erreichen kann), es ist jedoch kein Approximationsalgorithmus bekannt, der eine konstante Approximationsgüte erreicht. Der beste Approximationsalgorithmus erreicht eine logarithmische Approximationsgüte.

Eine zweite Gruppe von Sichtbarkeitsproblemen nennen wir "Verstecken". In einem Versteckproblem müssen wir eine maximale Anzahl von Positionen in einem Polygon oder einem Terrain finden, an denen wir Gegenstände oder Personen verstecken können, so dass keine zwei dieser Positionen sich gegenseitig sehen. Wir zeigen mittels einer lückenerhaltenden Reduktion von MAXIMUM CLIQUE, dass für Eingabepolygone mit Löchern oder Terrains kein polynomieller Approximationsalgorithmus eine Approximationsgüte von n^ϵ erreichen kann, für ein bestimmtes $\epsilon > 0$. Wiederum können wir für Eingabepolygone ohne Löcher nur ein schwächeres Resultat zeigen, nämlich APX-Schwere.

Eine dritte Gruppe von Sichtbarkeitsproblemen nennen wir "konvexes Überdecken". Hier soll ein gegebenes Polygon mit einer minimalen Anzahl von konvexen Polygonen, die alle innerhalb des gegebenen Polygons liegen, überdeckt werden. Wir geben einen Approximationsalgorthmus, der eine logarithmische Approximationsgüte für dieses Problem erreicht. Der Schlüssel dieses Algorithmus liegt im "Diskretisieren" der Ebene, ohne dabei zuviel zu verlieren. Durch Modifizieren und genaues Analysieren einer bereits bekannten Reduktion zeigen wir ausserdem, dass dieses Problem APX-schwer ist.

In einer vierten Gruppe von Sichtbarkeitsproblemen betrachten wir nur die Sichtbarkeitsgraphen als Eingabestrukturen und nicht die ganzen Polygone. Wir zeigen, wie einige klassische graphentheoretische Probleme auf Sichtbarkeitsgraphen geometrisch interpretiert werden können. Wir zeigen ausserdem, dass etliche unserer Resultate für allgemeine Sichtbarkeitsprobleme auch für diesen eingeschränkten Bereich gelten. Als weiteres Resultat zeigen wir, dass das MAXIMUM CLIQUE Problem auf Sichtbarkeitsgraphen von Polygonen mit Löchern fast so schwer ist wie auf allgemeinen Graphen. Dieses Resultat ist vor allem deshalb interessant, weil das entsprechende Problem für Polygone ohne Löcher in polynomieller Zeit lösbar ist.

Chapter 1

Introduction

1.1 Motivation

At the dawning of the information age, modern societies face tremendous challenges of social and technological nature. Communication and information are increasingly considered to be the crucial economic resources of the future. Increasing demand for communication has led to intensive research and, subconsequently, to the development of communication technology that seemed beyond imagination only years ago. Undoubtedly, the most promising approach in new communication technology is that of wireless communication.

In this sector, deregulation – a political instrument – has its share of responsibility for the mushrooming of new telecommunication companies and the deployment of a large number of wireless networks. Despite the current boom in wireless communication, basic economic intuition lets us anticipate that only the "fittest" and economically soundest of these companies will survive in the long run. Setting up, maintaining, and running wireless communication networks generates huge costs, but also prospects for huge revenues. Reducing costs may thus be a way of staying in business that is even more effective than in other markets.

Of course, the quality of a network largely depends on the quality of its "nodes", which are transmission stations or antennas in our case. The cost of a network, however, largely depends on the number of nodes. Typically, a network for wireless communication consists of transmission stations (antennas) that receive and send signals. The set of antennas needs to cover a specific geographic region in its entirety.

Setting up an antenna incurs:

- the cost of the acquisition of the site, i.e., the rent or purchase of the soil
- the cost of the antenna itself
- protests of the local population due to fears of overexposure to radiation

Thus, putting up antennas is very costly, and hence telecommunication companies aim at placing a minimum number of antennas that cover a given region.[1] Since the traditional way of erecting antenna towers on the ground suffers from a number of obvious disadvantages, a novel approach is to put antennas up in the air: Balloons float at a certain fixed height and are held in geo-stationary position.

This setting (as well as the traditional setting with antenna towers on the ground) poses a lot of *engineering* problems:

- Developing suitable transmitter technology
- Developing a system to keep the balloon in geo-stationary position

While these problems are certainly interesting, an important *combinatorial* problem that has a huge cost-cutting potential arises, too:

- Finding the positions of a minimum number of balloons such that communication between any two points in a given region is possible

Communication between two points is possible, if a mobile transmitter (such as a cellular phone) at each point can communicate with an antenna in the network. Thus, in our abstract problem, each point in the region must be covered by at least one antenna. Communication between antennas and mobile transmitters is by means of electromagnetic wave propagation at high frequencies. Current frequencies are 900 MHz and 1800 MHz and the trend points towards frequencies even higher in the GHz-range. A straight *line-of-sight* approach models reality with sufficient precision in these frequency ranges, since the effects of reflection and refraction become negligible. Thus, we require each point in the region to be *visible* from at least one antenna in the network.

Since visibility between two points in a region is given or blocked by the topology of the region to be covered, the geometric model used to model the

[1] This is of course an abstraction of the real-world problem of setting up an antenna network. The real-world problem is more complex. For example, depending on the type of network (mobile phones, paging, military, etc.), telecommunication companies have to deal with the limited ressource of frequencies that are available to them – in addition to the covering problem.

region becomes important. In the geometric model, we call the region to be covered a *terrain*, which is described as a finite set of points in the plane together with a triangulation, and a height value associated with each point (this is also called a *triangulated irregular network (TIN)*, see e.g. [33]). Visibility on a terrain is defined on the basis of straight *lines-of-sight*: Two points above the terrain are mutually visible if their connecting straight line segment runs entirely above (or on) the terrain.

Thus, our problem has engineering, combinatorial, and *geometric* aspects. We will focus exclusively on the combinatorial and geometric aspects.

The terrain covering problem can be seen to belong to quite a large family of geometric covering and guarding problems that have been studied for more than two decades. Legend has it that during a conference in 1976, Victor Klee started the study by posing the following problem, which today is known as the *original art gallery problem*: How many guards are needed to see every point in the interior of an art gallery? In the abstract version of this problem, the input is a simple polygon in the plane, representing the floor plan of the art gallery, and visibility is of course limited to the interior of the polygon. The variations of this polygon guarding problem that have been investigated can be classified as to where the guards may be positioned (anywhere, or in any one of a few distinguished locations), what kind of guards are to be used (single points versus sets of points, such as line segments, and guards in stationary positions versus mobile guards), whether only the boundary or all of the interior of the polygon must be guarded, what the assumptions are on the input polygons (such as being simple or orthogonal). Many upper and lower bounds on the number of necessary guards are known for specific settings, while comparatively few papers study the computational complexity of finding good positions for guards, given a polygon. For more details, see any of several surveys on the general topic of art galleries [39, 44, 48].

While guarding a polygon or a terrain has many straight-forward applications, the "opposite" problem of *hiding* a maximum number of objects from each other in a given polygon or terrain is intellectually appealing as well with a less prominent applicative background. However, when we let the problem instance be a terrain, we obtain the following background, which is the practical motivation for the theoretical study of our problem: A real estate agency owns a large, uninhabited piece of land in a beautiful area. The agency plans to sell the land in individual pieces to people who would like to have a cabin in the wilderness, which to them means that they do not see any signs of human civilization from their cabins. Specifically, they do not want to see any other cabins. The real estate agency, in order to maximize profit, wants to sell as many pieces of land as possible.

In an abstract version of the problem we are given a terrain which represents the uninhabited piece of land that the real estate agency owns. The problem now consists of finding a maximum number of lots (of comparatively small size) in the terrain, upon which three-dimensional bounding boxes can be positioned that represent the cabins such that no two points of two different bounding boxes see each other. Since the bounding boxes that represent the cabins are small compared to the overall size and elevation changes in the terrain (assume that we have a mountainous terrain), we may consider these bounding boxes to be zero-dimensional, i.e. to be points on the terrain. This problem has other potential applications in animated computer-games, where a player needs to find and collect or destroy as many objects as possible. Not seeing the next object while collecting an object makes the game more interesting.

Next to guarding and hiding, there are other problems that deal with visibility in a graph. One of the best studied of these problems is the problem of covering a given polygon by a minimum number of (possibly overlapping) convex polygons inside the given polygon.

Since most visibility problems cannot be solved optimally unless $NP = P$, it is also interesting to consider simplified versions of these problems, where the problem is reduced to the visibility graph, which contains a vertex for each polygon vertex and an edge between two vertices, if the corresponding polygon vertices see each other. In this setting, most classic visibility problem turn into graph problems. To illustrate this, consider the hiding problem, which turns into the independet set problem. The class of visibility graphs has so far defied characterization. As a different aproach to learn more about this graph class, we study how different graph problems behave in visibility graphs.

In the remainder of this chapter, we first give in Sect. 1.2 the precise definitions of all objects and problems that will be used throughout the thesis. We then make a detour to give an introduction into approximability classes as defined by [3] in Sect. 1.3, which is necessary in order to understand the previous results given in Sect. 1.4 and – even more – to understand the results of this thesis, which are summarized in Sect. 1.5

1.2 Definitions

1.2.1 Polygons, Terrains, and Visibility

Definition 1.2.1 *A* polygonal chain T *is an ordered sequence of points* $p_1, \ldots, p_n$, $n \geq 3$ *in the plane, called the* vertices *of* T, *together with the set of line segments joining* p_i *to* p_{i+1}, $i = 1, \ldots n-1$, *called the* edges *of* T.

A polygonal chain is called closed *if $p_1 = p_n$; otherwise it is called* open. *For a closed polygonal chain, we sometimes refrain from repeating the first vertex p_1, and we end the chain with p_{n-1}.*

Definition 1.2.2 *A polygonal chain is called* simple *if the only intersections of edges are those at common endpoints of consecutive edges. A simple, closed polygonal chain T divides the plane into two regions, the* interior *and the* exterior *of T, where the exterior is the unbounded region and the interior is the bounded region (it does not contain a line or even a halfline).*

Definition 1.2.3 *The interior of a simple, closed polygonal chain T, together with T, is called a* simple polygon without holes. *Its* boundary *δT is just T. For simplicity, the polygon is denoted by T as well.*

Definition 1.2.4 *A* polygon *is the union of a finite number of simple polygons. A polygon T is called* connected, *if any two points of T can be joined by a polygonal chain that belongs to T. A connected polygon T is called* simply connected *if every polygonal chain between two boundary points that does not pass through any other boundary point divides T. A connected polygon that is not simply connected is called a* simple polygon with holes.

Note that a simple polygon with holes T can be represented by a finite number k of polygonal chains $T_1, \ldots, T_k$ that represent its boundary, where T_1 is the *outer* boundary of the polygon, and the T_i, for $i = 2, \ldots, k$ are the boundaries of the *holes*. For this representation to work, we require that $T_i \subseteq T_1$ for $i = 2, \ldots, k$ and that $T_i \cap T_j = \emptyset$ for $i \neq j$ and $i, j = 2, \ldots, k$. The interior of T is the set difference between T_1 and the interiors of $T_2, \ldots, T_k$ viewed as simple polygons without holes. Note that the boundaries of $T_2, \ldots, T_k$ belong to T.

Since we only deal with connected polygons in the following, we will use the term polygon for a connected polygon, with or without holes.

Among the multitude of notions for *visibility* between two points in a polygon, we will use the following:

Definition 1.2.5 *Let T be a polygon, and let a and b be points belonging to T. Points A and B are mutually* visible *with respect to T, if the straight line segment connecting a and b belongs to T. We also say that a and b* see *each other, that a is visible from b, and that a sees b. For a set Q and a set S of points of T, we say that Q is visible from S if for each point $q \in Q$ there is a point $s \in S$ that sees q.*

Note that visibility is symmetric for single points, but not for sets of points: While a set Q may be visible from a set S of points with respect

to a polygon T, it may not be true that S is visible from Q. We therefore call Q the set of guarded points and S the set of guard points (or simply guards).

We will define visibility problems not only on polygons as input structures, but also on *terrains*.

Definition 1.2.6 *A* terrain *T is a two-dimensional surface in three-dimensional space, represented as a finite set of vertices in the plane, together with a triangulation of their planar convex hull, and a height value associated with each vertex. By a linear interpolation in between vertices, this representation defines a bivariate, continuous function. The corresponding surface in space is also called the 2.5-dimensional terrain. A terrain divides three-dimensional space into two subspaces, i.e., a space* above *and a space* below *the terrain, in the obvious way.*

For simplicity, we will describe the terrain problems in the Cartesian $x-y-z$-space, where the z-value denotes the height of terrain points.

Definition 1.2.7 *Let T be a terrain, and let a and b be two points in space above or on T. Point a is* visible *from point b with respect to T if the straight line segment connecting a and b is entirely on or above T.*

For antennas, this definition does not model all aspects of electromagnetic wave propagation exactly, since e.g. the signal of an antenna gets weaker as it propagates, and the signal is reflected on a rocky wall. However, for the practical problem that motivates this study, the straight *line of sight* (LOS) approach provides a satisfactory approximation of reality.

Definition 1.2.8 *A* 1.5 DIMENSIONAL TERRAIN *T is a one-dimensional surface in the two-dimensional plane, represented by a set of vertices on the x-axis together with a height value for each vertex.*

Visibility on 1.5 dimensional terrains is defined accordingly as for 2.5 dimensional terrains.

1.2.2 Visibility Problems

We now define the problems we are studying.

Definition 1.2.9 *Let T be a simple polygon without holes. The problem* MINIMUM VERTEX GUARD WITHOUT HOLES *is the problem of finding a minimum subset S of (the set of) vertices of T such that the boundary and the interior of T is visible from S. The vertices in S are called* vertex guards.

Note that as usual, a minimum subset of a set denotes a subset of smallest cardinality among all candidate subsets.

Definition 1.2.10 *Let T be a simple polygon without holes. The problem* MINIMUM EDGE GUARD WITHOUT HOLES *is the problem of finding a minimum subset S of edges of T such that the boundary and the interior of T is visible from the points in S. The edges in S are called* edge guards.

Definition 1.2.11 *Let T be a simple polygon without holes. The problem* MINIMUM POINT GUARD WITHOUT HOLES *is the problem of finding a minimum set S of points belonging to T such that the boundary and the interior of T is visible from S. The points in S are called* point guards.

These problems can also be defined such that only the boundary (rather than the boundary and the interior) of the input polygon T must be visible from at least one guard. We call the corresponding problems MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITHOUT HOLES, MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITHOUT HOLES, and MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES,

Finally, we can define these problems for input polygons with (instead of without) holes. This yields the problems MINIMUM VERTEX GUARD WITH HOLES, MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITH HOLES, MINIMUM EDGE GUARD WITH HOLES, MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITH HOLES, MINIMUM POINT GUARD WITH HOLES, and MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES.

For a terrain, let us consider the following problems:

Definition 1.2.12 *Let T be a terrain. The problem* MINIMUM VERTEX GUARD ON TERRAIN *is the problem of finding a minimum subset S of vertices of T such that T is visible from S.*

Definition 1.2.13 *Let T be a terrain. The problem* MINIMUM POINT GUARD ON TERRAIN *is the problem of finding a minimum set S of points on T such that T is visible from S.*

Definition 1.2.14 *Let T be a terrain, and let h be a height value, such that the plane $z = h$ lies entirely above (or partially on) T. The problem* MINIMUM FIXED HEIGHT GUARD ON TERRAIN *is the problem of finding a minimum set S of points in space at height h such that T is visible from S.*

Of course, we can define this problem on a 1.5 dimensional terrain and thus get the problem MINIMUM FIXED HEIGHT GUARD ON 1.5 D TERRAIN.

We now define three additional terrain guarding problems. The reason is that our inapproximability results for guarding terrains will be formulated for terrain problem versions with the additional restriction that each triangle in the triangulation of T must be visible from a single point in the guard set S; that is, guards are not allowed to cooperatively see a triangle in T's triangulation, contrary to the problem versions above. We denote these problem version by MINIMUM VERTEX GUARD ON TERRAIN WITH TRIANGLE RESTRICTION, MINIMUM POINT GUARD ON TERRAIN WITH TRIANGLE RESTRICTION, MINIMUM FIXED HEIGHT GUARD ON TERRAIN WITH TRIANGLE RESTRICTION

We now define the hiding problems.

Definition 1.2.15 *The problem* MAXIMUM HIDDEN SET ON TERRAIN *asks for a set S of maximum cardinality of points on a given terrain T, such that no two points in S see each other.*

In a variant of the problem, we introduce the additional restriction that these points on the terrain must be vertices of the terrain.

Definition 1.2.16 *The problem* MAXIMUM HIDDEN VERTEX SET ON TERRAIN *asks for a set S of maximum cardinality of vertices of a given terrain T, such that no two vertices in S see each other.*

In a more abstract variant of the same problem, we are given a simple polygon with or without holes instead of a terrain. This yields the following four problems.

Definition 1.2.17 *The problem* MAXIMUM HIDDEN SET ON POLYGON WITH(OUT) HOLES *asks for a set S of maximum cardinality of points in the interior or on the boundary of a given polygon P, such that no two points in S see each other.*

Definition 1.2.18 *The problem* MAXIMUM HIDDEN VERTEX SET ON POLYGON WITH(OUT) HOLES *asks for a set S of maximum cardinality of vertices of a given polygon P, such that no two vertices in S see each other.*

Another problem that we will study with respect to its approximation properties is MINIMUM CONVEX COVER.

Definition 1.2.19 *Let T be a simple polygon with or without holes. The problem* MINIMUM CONVEX COVER *is the problem of covering a given polygon T with a minimum number of (possibly overlapping) convex polygons that lie in T.*

1.2.3 Other Problems

We prove our inapproximability results by proposing reductions from the following problems.

Definition 1.2.20 *Let Φ be a boolean formula given in conjunctive normal form, with each clause consisting of at most 3 (2) literals and with each variable appearing in at most 5 clauses. The problem* MAXIMUM 5-OCCURRENCE-3 (2)-SATISFIABILITY *consists of finding a truth assignment for the variables of Φ that satisfies as many clauses as possible.*

Definition 1.2.21 *Let $E = \{e_1, \ldots, e_n\}$ be a finite set (called universe) of elements, and let $S = \{s_1, \ldots, s_m\}$ be a collection of subsets of E, i.e., $s_j \subseteq E$ for $1 \leq j \leq m$. The problem* SET COVER *is the problem of finding a minimum subset $S' \subseteq S$ such that every element $e_i \in E$, $1 \leq i \leq n$ belongs to at least one subset in S'. For ease of discussion, let the elements in E and the subsets in S have an arbitrary, but fixed order, denoted by the index.*

Definition 1.2.22 *Let $G = (V, E)$ be an undirected graph with n vertices $V = \{v_1, \ldots, v_n\}$ and edges E. The problem* DOMINATING SET *consists of finding a minimum set S' of vertices such that each vertex $v_i \in V$ has at least one neighboring vertex in S', i.e., for each vertex $v_i \in V$, there exists a vertex $v_j \in S'$ with $(v_i, v_j) \in E$.*

1.3 Approximation Classes and Gap-Preserving Reductions

1.3.1 Approximation Algorithms and Approximation Ratios

In this section, we give an introductory overview of the notions of approximability, approximation ratio, gap-preserving reduction, and approximation classes. These are all well-known concepts that can be found in any textbook on approximation algorithms, in particular in [31] and [3].

Let Π be a combinatorial optimization problem, such as MAXIMUM CLIQUE, and let I be an instance of Π; this would be a graph $G = (V, E)$ for MAXIMUM CLIQUE. Let $\Sigma(I)$ be the set of all feasible solutions of I; in our example of MAXIMUM CLIQUE, $\Sigma(I)$ would be the set of all subsets SOL of the vertex set V, where the elements (vertices) in SOL form a clique. Let $OPT(I)$ denote the size of an optimum solution of I, i.e. $OPT(I) = \max_{SOL \in \Sigma(I)} |SOL|$, if Π is a maximization problem,

and $OPT(I) = \min_{SOL \in \Sigma(I)} |SOL|$, if Π is a minimization problem, where $|SOL|$ denotes the cardinality of the set SOL.

Let A be an approximation algorithm for problem Π. By definition, an approximation algorithm runs in time polynomial in its input. The output of algorithm A on input of instance I is a feasible solution $SOL(A(I))$, which is an element of $\Sigma(I)$.

An approximation algorithm A for problem Π is said to achieve an approximation ratio of $R_A(I)$, if for all instances I of problem Π:

$$R_A(I) \geq \max\{\frac{OPT(I)}{|SOL(A(I))|}, \frac{|SOL(A(I))|}{OPT(I)}\}.$$

Note that the approximation ratio is thus defined to always be a number greater or equal to one, no matter whether the optimization problem Π is a maximization or minimization problem.

We say that an optimization problem Π cannot be approximated with an approximation ratio of $R(I)$, if for every (polynomial time) approximation algorithm A for Π, there exists an instance I of Π with:

$$R(I) < R_A(I).$$

In this case, we speak of an inapproximability result. Of course, such inapproximability results are always under the assumption that $NP \neq P$. In some cases, the strongest inapproximability results only hold under somewhat stronger assumptions (such as $coR \neq NP$ or $NP \neq TIME(n^{O(\log \log n)})$) that are generally assumed to be true.[2]

1.3.2 Promise Problems

Promise problems are used to create a connection between optimization problems and decision problems. In a promise problem, we are given an instance I of a maximization problem Π, and we are promised (i.e., some oracle that we trust says) that the size $OPT(I)$ of the optimum solution of I is either at least $U(I)$ or strictly smaller than $L(I)$, but no value between $L(I)$ and $U(I)$, i.e.:

$$OPT(I) \geq U(I)$$

or

$$OPT(I) < L(I)$$

The two bounds $U(I)$ and $L(I)$ depend on instance I, and obviously, $L(I) < U(I)$. The promise problem consists of determining, which of the two cases

[2] This notion is also called "quasi NP-hardness". Alternatively, the notation $\tilde{P} \neq NP$ (read "soft P does not equal NP") is also used in the literature [3].

is true. Thus, a promise problem is given as a pair of functions $(L(I), U(I))$. The definition can be extended canonically to minimization problems.

Let us consider an example. Let Π be MAXIMUM CLIQUE, let instance I be a graph $G = (V, E)$ with $n := |V|$, let k be an integer with $k \leq n$, and let $\epsilon > 0$ be a small constant. Then, we let $U(I) := k$ and $L(I) := \frac{k}{n^{1/2-\epsilon}}$. Thus, we have the promise problem of MAXIMUM CLIQUE, where we are promised that either

$$OPT(I) \geq k$$

or

$$OPT(I) < \frac{k}{n^{1/2-\epsilon}}.$$

Now, the following holds:

Theorem 1.3.1 [29] *The promise problem of* MAXIMUM CLIQUE *with parameters* $U(I) := k$ *and* $L(I) := \frac{k}{n^{1/2-\epsilon}}$ *is NP-hard to decide.*

The connection of promise problems to inapproximability result is now evident: Since no polynomial time algorithm can decide the promise problem with parameters $U(I) := k$ and $L(I) := \frac{k}{n^{1/2-\epsilon}}$, no polynomial time approximation algorithm can guarantee an approximation ratio $R(I)$ of

$$R(I) = \frac{U(I)}{L(I)} = \frac{k}{\frac{k}{n^{1/2-\epsilon}}} = n^{1/2-\epsilon},$$

because such an approximation algorithm could be used to decide the promise problem.

Let us state two additional NP-hardness results for promise problems.

Theorem 1.3.2 [24] *The promise problem of* MINIMUM SET COVER *with parameters* $U(I) := k(1-\epsilon)\ln n$ *and* $L(I) := k$ *for any* $\epsilon > 0$, *where* n *denotes the number of elements in instance* I, *and where* k *depends on* I, *is* NP*-hard to decide.*

Theorem 1.3.3 [3] *The promise problem of* MAXIMUM SATISFIABLITY *with parameters* $U(I) := m$ *and* $L(I) := (1-\epsilon)m$ *for some* $\epsilon > 0$, *where* m *denotes the number of clauses in instance* I, *is* NP*-hard to decide.*

Theorems 1.3.1, 1.3.2, and 1.3.3 are proved by constructing special Probabilistically Checkable Proof Systems (PCP systems) for an NP-complete (decision) problem. All possible random computation paths of such a PCP system are encoded into an instance of MAXIMUM CLIQUE, MINIMUM SET COVER, or MAXIMUM SATISFIABILITY, respectively. The gaps, i.e., the differences between the upper bounds $U(I)$ and the lower bounds $L(I)$, are

basically obtained by counting the number of computation paths of the PCP system that lead to an affirmative answer to the input decision problem. For additional information on the very complex issue of PCP systems and in particular on the PCP theorem, which has made these results possible, see [2] and [3].

1.3.3 Gap-Preserving Reductions

A gap-preserving reduction (as proposed in [3]) is a polynomial time algorithm f that transforms an instance I of a promise problem Π with upper bound $U(I)$ and lower bound $L(I)$ into an instance $I' := f(I)$ of another promise problem Π' with upper bound $U'(I')$ and lower bound $L'(I')$. Thus, the gap between the upper and lower bound is "preserved". Note, however, that the functions $U(I)$ and $L(I)$ may be very different from $U'(I')$ and $L'(I')$: the gap may be shrunk or expanded by the reduction, but there always remains a gap.

We can illustrate the effect of a gap-preserving reduction by two implications that need to be satisfied by the reduction:

$$OPT(I) \geq U(I) \implies OPT(I') \geq U'(I')$$

$$OPT(I) < L(I) \implies OPT(I') < L'(I')$$

If the promise problem Π with parameters $U(I)$ and $L(I)$ is NP-hard to decide, then the promise problem Π' with parameters $U'(I')$ and $L'(I')$ is NP-hard to decide as well.

To see this, assume by contradiction that there exists a polynomial time algorithm that decides the promise problem Π'. This would directly allow us to decide the promise problem Π: Suppose w.l.o.g that our algorithm tells us that $OPT(I') < F'(I')$ holds. This implies that $OPT(I') \geq U'(I')$ is false, which implies that $OPT(I) \geq U(I)$ is false, too. This, however, implies $OPT(I) < L(I)$. So we have decided the promise problem Π.

Thus the existence of such a gap-preserving reduction implies that problem Π' cannot be approximated with an approximation $R'(I')$ of

$$R'(I') = \frac{U'(I')}{L'(I')}.$$

1.3.4 Approximation Classes

Optimization problems can be classified with respect to the approximation ratios that can be achieved for them by polynomial approximation algorithms. This classification is of course somewhat arbitrary and may reflect more the limitations of our current proof methods than reality.

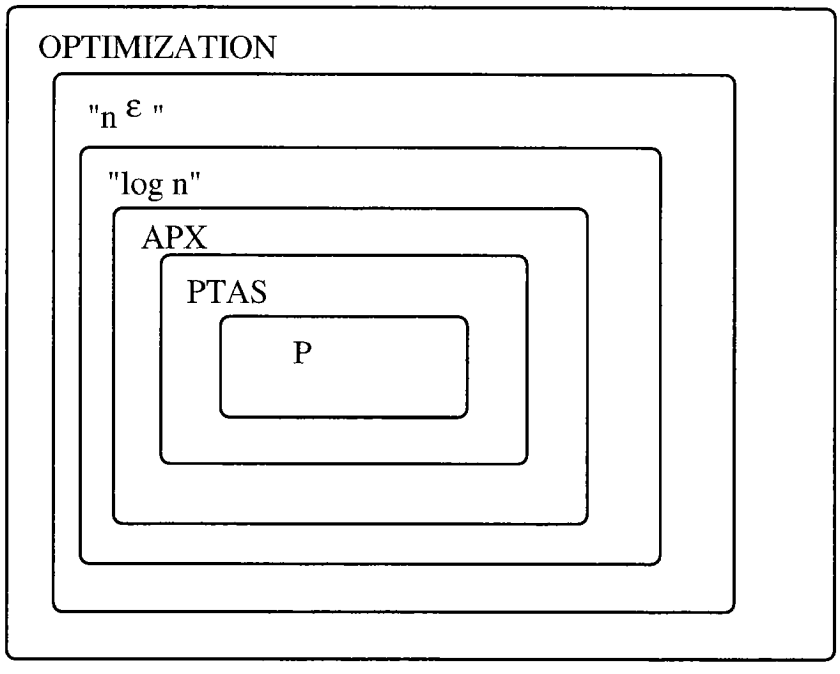


Figure 1.1: Approximation Classes

Nevertheless, a generally accepted classification is illustrated in Fig. 1.1: *OPTIMIZATION* denotes the class of all combinatorial optimization problems.

We define the class "n^ϵ" as a subclass of *OPTIMIZATION*. An optimization problem lies in "n^ϵ", if there exists an $\epsilon > 0$ such that the problem can be approximated by a polynomial time algorithm with an approximation ratio of $O(n^\epsilon)$, where n is the size of the problem instance. We call an optimization problem "n^ϵ"-hard, if there exists an $\epsilon > 0$ such that no polynomial time approximation algorithm can achieve an approximation ratio of $R(I)$, where $R(I) \in \Theta(n^\epsilon)$. An optimization problem is "n^ϵ"-complete, if it is in "n^ϵ" and "n^ϵ"-hard. An example of an "n^ϵ"-complete problem is MAXIMUM CLIQUE.

An optimization problem lies in the class "$\log n$" (which is a subclass of "n^ϵ"), if there exists a polynomial approximation algorithm that achieves an approximation ratio of $O(\log n)$, where n denotes the size of the problem instance. We call an optimization problem "$\log n$"-hard, if no polynomial time approximation algorithm can achieve an approximation ratio of $R(I)$,

where $R(I) \in \Theta(\log n)$. An optimization problem is "$\log n$"-complete, if it is in "$\log n$" and "$\log n$"-hard. An example of a "$\log n$"-complete problem is MINIMUM SET COVER.

An optimization problem lies in the class APX (which is a subclass of "$\log n$"), if there exists a polynomial approximation algorithm that achieves an approximation ratio of $1 + \epsilon$ for some $\epsilon > 0$. We call an optimization problem APX-hard, if there exists an $\epsilon > 0$ such that no polynomial time approximation algorithm can achieve an approximation ratio of $1 + \epsilon$. An optimization problem is APX-complete, if it is in APX and APX-hard. An example of an APX-complete problem is MAXIMUM SATISFIABILITY.

An optimization problem lies in the class $PTAS$ (which is a subclass of APX), if there exists a polynomial time approximation algorithm that achieves an approximation ratio of $1+\epsilon$ for any constant $\epsilon > 0$. The running time of the algorithm may be exponential in $\frac{1}{\epsilon}$.

Finally, a problem lies in the class P, if it can be solved in polynomial time.[3]

An important subclass of APX is called $MAXSNP$, which is defined as the class of all optimization problems that are L-reducible to MAXIMUM SATISFIABILITY. See [41] for an introduction to $MAXSNP$ and the notion of an L-reduction. An optimization problem is $MAXSNP$-complete, if all problems in $MAXSNP$ can be L-reduced to it. Most variations of satisfiability including MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY and MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY are $MAXSNP$-complete [41]. This means that there exist L-reductions from MAXIMUM SATISFIABILITY to these problems. Since every L-reduction is also a gap-preserving reduction, the corresponding promise problems of MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY and MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY are NP-hard to decide.

A major motivation for introducing such a classification of approximability is the desire to rate the quality of an approximation algorithm and of inapproximability results. For our purposes, we will call an inapproximability result or an approximation algorithm for an optimization problem tight, optimum, or matching, if it lets us classify the problem to be complete for one of the approximation classes.

Note that we could strengthen the notion of optimality of results in such a way, that also low-order terms must match. Research on such strongly optimum result is primarily done on important basic problems. MAXIMUM CLIQUE, for example, is inapproximable with an approximation ratio of $n^{1-\epsilon}$ for any $\epsilon > 0$ [29], while a trivial algorithm achieves a matching approxi-

[3] The class "n^{ϵ}" is equivalent to class IV as introduced in [3]. Class "$\log n$" is equivalent to class II and APX corresponds to class I.

mation ratio of n. For APX-complete problems, stronger optimality means finding a constant $\epsilon > 0$ such that an approximation algorithm achieves an approximation ratio of $1+\epsilon$, but no polynomial time algorithm can achieve an approximation ratio of $1+\epsilon-\gamma$ for any $\gamma > 0$. An example of an APX-complete problem, for which such strongly optimum results exist is MAXIMUM EXACT-3-SATISFIABILITY, which is a satisfiability variant with exactly three literals in each clause [28]. See the next section for additional strongly optimum results.

1.4 Previous Work

Traditionally, the art gallery community has studied upper and lower bounds of visibility problems with respect to descriptional complexity. The original *art gallery theorem* is also of this type. It says that $\lfloor \frac{n}{3} \rfloor$ guards always suffice and are sometimes necessary to guard a given polygon with n vertices. In this thesis, we are more interested in computational complexity results.

Guarding a given polygon is NP-hard in all versions as defined in 1.2, i.e., MINIMUM (BOUNDARY RESTRICTED) VERTEX/EDGE/POINT GUARD WITH(OUT) HOLES is NP-hard [35].

Approximation algorithms for MINIMUM VERTEX/EDGE GUARD WITH-(-OUT) HOLES, which achieve approximation ratios of $O(\log n)$, are also known [26]. These algorithms can be easily modified to work for the boundary restricted versions of these guarding problems as well.

Hiding in all versions, i.e., MAXIMUM HIDDEN (VERTEX) SET WITH(OUT) HOLES is NP-hard as well [45].

MINIMUM CONVEX COVER is NP-hard [12], too. Interestingly, the related problem of partitioning a given polygon without holes into a minimum number of (non-overlapping) convex polygons is solvable in polynomial time [9]. The problem becomes NP-hard only if the input polygons have holes [36]. The related problem MINIMUM RECTANGLE COVER of covering a given orthogonal polygon with a minimum number of rectangles can be approximated with a constant ratio for polygons without holes [25] and with an approximation ratio of $O(\sqrt{\log n})$ for polygons with holes [34].

The technique of using dynamic programming to find maximum convex structures has been used before to solve the problem of finding a maximum (with respect to the number of vertices) empty convex polygon, given a set of vertices in the plane [14]. We will use a similar approach to find approximation algorithms for MINIMUM CONVEX COVER.

There are several excellent surveys on visibility results [39, 44, 48].

There are various problems that deal with terrains. Quite often, these problems have applications in the field of telecommunications, namely in set-

ting up communications networks. There are some upper and lower bound results on the number of guards needed for several kinds of guards to collectively cover all of a given terrain [6]. Very few results on the computational complexity of terrain problems are known. The shortest watchtower (from where a terrain can be seen in its entirety) can be computed in time $O(n \log n)$ [49].

Most results in this thesis are inapproximability results, which are obtained from gap-preserving reductions [3] from classic NP-hard problems. Therefore, we summarize the (in-)approximability results for these problems as well.

MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY and MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY are APX-complete [41].

MINIMUM SET COVER and MINIMUM DOMINATING SET can be approximated in polynomial time with a ratio of $\ln n + 1$ by a simple greedy algorithm [10, 32], but cannot be approximated by any polynomial time algorithm with a ratio of $(1 - \epsilon) \ln n$, for any $\epsilon > 0$ unless

$$NP \subseteq TIME(n^{O(\log \log n)}),$$

where n is the number of elements for MINIMUM SET COVER and the number of vertices in the graph for MINIMUM DOMINATING SET [10, 24]. There are also logarithmic inapproximability results for these problems under the weaker assumption that $NP \neq P$ [4], i.e., an approximation ratio of $c \log n$ is hard to achieve for some constant $c > 0$.

MAXIMUM CLIQUE and MAXIMUM INDEPENDENT SET cannot be approximated with an approximation ratio of $n^{1-\epsilon}$ for any $\epsilon > 0$ unless $NP = coR$, and with an approximation ratio of $n^{\frac{1}{2}-\epsilon}$ for any ϵ unless $NP = P$ [29], where n is the number of vertices in the input graph.

1.5 Summary of Results

In Chapt. 2, we present algorithms, approximation algorithms, and inapproximability results for guarding problems.

We start with the apparently easiest problem: MINIMUM FIXED HEIGHT GUARDS ON 1.5 D TERRAIN. In Sect. 2.1, we propose a linear time algorithm to find the optimum solution of any MINIMUM FIXED HEIGHT GUARDS ON 1.5 D TERRAIN instance. Moreover, we characterize the solution space of all optimum solutions of an instance.[4] The algorithm proposed

[4] There actually is already a linear time algorithm known for this problem, even though it comes from a quite different background [37]; however, the characterization of the solution space is a new result.

constructs the solution incrementally and the characterization of the solution space is obtained by letting the algorithm run from left to right and from right to left.

In Sect. 2.2, we show that MINIMUM (BOUNDARY RESTRICTED) VERTEX/EDGE/POINT GUARD WITHOUT HOLES is APX-hard. To this end, we construct a reduction from MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY to MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES that is gap-preserving and thus establishes the APX-hardness of MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES. The same reduction with minor modifications can also be used to show APX-hardness for the other guarding problems for polygons without holes. The reduction is based on an already known reduction [35] that was originally used to prove NP-hardness for these problems. The results in Sect. 2.2 have been reported previously [22].

We continue with inapproximability results for guarding problems for polygons with holes in Sect. 2.3. We propose a gap-preserving reduction from RESTRICTED MINIMUM SET COVER to MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES that establishes a logarithmic inapproximability result. More precisely, we show that no polynomial time approximation algorithm can guarantee an approximation ratio of $\frac{1-\epsilon}{12} \ln n$ for any $\epsilon > 0$, unless $NP \subseteq TIME(n^{O(\log \log n)})$, where n is the number of vertices of the input polygon. Again, minor modifications of the construction of the reduction establish the same logarithmic inapproximability results for MINIMUM (BOUNDARY RESTRICTED) VERTEX/EDGE/POINT GUARD WITHOUT HOLES. The results in this section have been published previously [20].

Our inapproximability results for MINIMUM (BOUNDARY RESTRICTED) VERTEX/EDGE GUARD WITH HOLES are optimum upto a constant factor because there exist approximation algorithms with logarithmic approximation ratios for these problems [26]. Thus, these problems are all "$\log n$"-complete. The inapproximability results for MINIMUM (BOUNDARY RESTRICTED) VERTEX/EDGE GUARD WITHOUT HOLES are not tight: there still exists a gap between the best inapproximability result, which says that these problems are APX-hard, and the best approximation algorithms, which achieve logarithmic approximation ratios for these problems. The situation is even worse for MINIMUM (BOUNDARY RESTRICTED) POINT GUARD WITH(OUT) HOLES, where the best approximation algorithm achieves an approximation ratio of $\Theta(n)$.

In Sect. 2.4, we show how our inapproximability result for MINIMUM POINT GUARD WITH HOLES translates into logarithmic inapproximability results for MINIMUM FIXED HEIGHT/VERTEX/POINT GUARD ON TER-

RAIN (WITH TRIANGLE RESTRICTION) by constructing a very simple terrain from a polygon with holes. The results in this section have been reported previously [21].

We proceed by proposing approximation algorithms for MINIMUM FIXED HEIGHT/VERTEX GUARD ON TERRAIN (WITH TRIANGLE RESTRICTION) and for MINIMUM POINT GUARD ON TERRAIN WITH TRIANGLE RESTRICTION, which achieve logarithmic approximation ratios in Sect. 2.5. These approximation algorithms are thus optimal upto a constant factor. The results from Sect. 2.5 have been reported previously together with the results from Sect. 2.2, Sect. 2.3, and Sect. 2.4 [18].

Chapter 3 deals with the problem of hiding in polygons with and without holes and in terrains. We present inapproximability results that have been reported previously [19].

We prove the APX-hardness of MAXIMUM HIDDEN (VERTEX) SET WITHOUT HOLES in Sect. 3.1 by proposing a gap-preserving reduction from MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY to MAXIMUM HIDDEN SET WITHOUT HOLES, which also works for MAXIMUM HIDDEN VERTEX SET WITHOUT HOLES with minor modifications. The reduction that we propose is simpler than the one used originally to show NP-hardness [45]. However, this inapproximability result is far from tight, since the best approximation algorithm achieves only an approximation ratio of $\Theta(n)$.

In Sect. 3.2, we propose a gap-preserving reduction from MAXIMUM CLIQUE to MAXIMUM HIDDEN (VERTEX) SET WITH HOLES, which implies that MAXIMUM HIDDEN (VERTEX) SET WITH HOLES cannot be approximated with an approximation ratio of $\frac{n^{\frac{1}{6}-\gamma}}{4}$ for any $\gamma > 0$, unless $NP = P$, where n is the number of vertices in the input polygon. This inapproximability result clearly places these hiding problems on polygons with holes into the same inapproximability class as MAXIMUM CLIQUE, i.e. they are "n^{ϵ}"-complete.

We show in Sect. 3.3 that the inapproximability results for MAXIMUM HIDDEN VERTEX SET WITH HOLES carries over to MAXIMUM HIDDEN (VERTEX) SET ON TERRAIN.

In Chapt. 4, we attack the problem of covering a polygon with a minimum number of convex polygons. The results from this chapter have been reported previously [17].

After a short introduction in Sect. 4.1, we propose a polynomial time approximation algorithm for MINIMUM CONVEX COVER that achieves a logarithmic approximation ratio in Sect. 4.2. To get this result, we first show that any solution of an instance of MINIMUM CONVEX COVER can be transformed into a solution, where the vertices of the convex polygons that cover the input polygon, may only lie on a certain grid, in such a

way that the transformed solution contains at most three times as many convex polygons as the original solution. Our algorithm then uses dynamic programming to find the convex polygon with vertices only on the grid that contains a maximum number of basic units of the input polygon that are not yet covered by a convex polygon in the solution. If we apply this algorithm iteratively, we obtain the result.

Sect. 4.3 contains a proof for the APX-hardness of MINIMUM CONVEX COVER, which is by a gap-preserving reduction from MAXIMUM 5-OCC-3-SAT. The reduction relies on the reduction that was used to show NP-hardness of the problem [12].

Chapter 5 deals with visibility graphs. We show that a a variety of visibility problems have a graph-theoretic nature, when we restrict the input to visibility graphs. The problem of guarding, for example, corresponds to the problem of MINIMUM DOMINATING SET. We investigate several graph problems on visibility graphs with respect to differences between visibility graphs for polygons with and polygons without holes. The results of this chapter have been reported previously [16].

In Sect. 5.1, we give an introduction. In Sect. 5.2, we show that our inapproximability results for guarding polygons with and without holes carry over to the problem MINIMUM DOMINATING SET ON VISIBILITY GRAPHS. Hiding problems correspond to the MAXIMUM INDEPENDENT SET problem. In Sect. 5.3, we show that the inapproximability results for finding hidden sets in polygons with and without holes carry over to MAXIMUM INDEPENDENT SET ON VISIBILITY GRAPHS.

Finding maximum cliques in visibility graphs corresponds to finding largest (with respect to number of vertices) convex polygons in a given polygon. In Sect. 5.4, we show that the problem MAXIMUM CLIQUE ON VISIBILITY GRAPH WITHOUT HOLES can be solved in polynomial time using dynamic programming.[5] However, MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES cannot be approximated by any polynomial time algorithm with an approximation ratio of $\frac{n^{\frac{1}{6}-\gamma}}{4}$ for any $\gamma > 0$, unless $NP = P$, where n is the number of vertices in the input polygon. We prove this result by modifing the reduction from Sect. 3.2. Thus, we have found a classic graph problem (i.e.MAXIMUM CLIQUE) that is solvable in polynomial time for visibility graphs for polygons without holes and that is almost as hard to approximate as MAXIMUM CLIQUE on general graphs on visibility graphs for polygons with holes.

We use our algorithm for finding maximum cliques in visibility graphs for polygons without holes to build an approximation algorithm for MINIMUM

[5] There are algorithms known for finding the largest emtpy convex polygon in a point set that also solve this problem [14].

CLIQUE PARTITION ON VISIBILITY GRAPHS WITHOUT HOLES that achieves a logarithmic approximation ratio in Sect. 5.5. We also show that MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPHS WITHOUT HOLES is APX-hard and that MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPHS WITH HOLES cannot be approximated with an approximation ratio of $O(n^{\epsilon})$ for some constant $\epsilon > 0$. The results from Sect. 5.4 and Sect. 5.5 have been reported previously [16]. In Sect. 5.6, we show that not all visibility graphs for polygons with holes have Hamilton circles, while – trivially – all visibility graphs for polygons without holes have Hamilton circles.

Chapter 6 contains summarizing and concluding remarks.

Chapter 2

Guarding

2.1 A Linear Time Algorithm for MINIMUM FIXED HEIGHT GUARD ON 1.5D TERRAIN

2.1.1 Introduction

Before we attack the problems of guarding polygons or 2.5 dimensional terrains, we study the problem of guarding a 1.5 dimensional terrain with guards at fixed height above the terrain. This will turn out to be the only guarding problem among the ones we study that is solvable in polynomial time. The same problem in a very different setting has already been studied and a linear time algorithm has been proposed [37]. However, in Sect. 2.1.4, we present a characterization of the set of all optimum solutions, which yields a much deeper insight into the structure of the problem.

An instance I of MINIMUM FIXED HEIGHT GUARD ON 1.5D TERRAIN consists of a 1.5 dimensional terrain T given by its n vertices $\{v_1, \ldots, v_n\}$ in the x-y-plane that are ordered (w.l.o.g. ascendingly) according to their x-coordinates, and a height value h that is larger than the maximum y-value of all vertices. A solution $C = \{c_1, \cdots, c_k\}$ of I consists of points (guards) on the horizontal line $y = h$. In the algorithm that we are going to propose, pairs of points on the terrain T and on the horizontal line $y = h$ will be compared as to their x-coordinate. We therefore say a point p is to the right of a point p and write $p > p'$, if the x-coordinate of p is strictly greater than the x-coordinate of p'.

In Sect. 2.1.2, we propose a basic construction of the solution. We prove that this construction finds an optimum solution if it terminates. By refining the construction, we propose in Sect. 2.1.3 an algorithm that is easy

to implement and easy to analyze. It turns out that the algorithm runs in linear time and space and is therefore asymptotically optimal. In addition, the algorithm leads to a characterization of the set of all optimum solutions, presented in Sect. 2.1.4. This characterization should help in choosing among all possible optimum placements the best one with respect to other criteria that have not been formulated in the objective of the optimization problem.

2.1.2 An Incremental Construction

Before proposing an incremental construction of an optimum solution, let us discuss a few properties of the structure of the solution.

Lemma 2.1.1 *For any fixed point p on the terrain T, the set of all points c on the line $y = h$ that are visible from p forms a single interval $[l(p), u(p)]$.*

Proof: Let $l(p)$ ($u(p)$) be the leftmost (rightmost) point on the horizontal line $y = h$ that sees p. Assume by contradiction, that there is an point c on the line $y = h$ that does not see p in between the two points $l(p)$ and $u(p)$ that are both visible from p, i.e., with $l(p) < c < u(p)$. Since c is invisible from p, the line segment connecting c with p must intersect the terrain T at least once. Fix any such intersection point p', and with no loss of generality let $p' > p$. The terrain connects p' and p. Since $u(p)$ is to the right of c, the line segment from $u(p)$ to p will either cross the terrain at some point between p and p' or reach p from underneath the terrain. In both cases $u(p)$ is invisible from p, a contradiction. □

We call the interval $[l(p), u(p)]$ for a given point $p \in T$ the *visibility interval* of p. In our construction, we only need the right (or *upper*) end $u(p)$ of the visibility interval.

Let $U_{\min}(\emptyset)$ be the minimum of all $u(p)$. As a first step in our construction, it seems reasonable to position the first guard at $c_1 = U_{\min}(\emptyset)$. Given a partial solution C, i.e. a set of guards that does not cover all of T, let $U_{\min}(C)$ be the minimum of all upper ends $u(p)$ of the visibility intervals of all points p on the terrain T that are not visible from any guard in C. Our first proposal is to simply add the point $U_{\min}(C)$ as a guard to the solution C, thereby changing C, and to do so repeatedly until all points on the terrain T are visible from a guard in C. We call this construction the *incremental construction.*

Let c_i be the point added to C in the i-th iteration, and let k be the number of iterations. Obviously, k is equal to $|C|$. The incremental construction will always find the minimum $u(p)$ and add it to C, starting with

the empty set. Therefore, the elements in C will be ordered according to their x-coordinates: $c_1 < c_2 < \cdots < c_k$.

At this point it is not clear whether this incremental construction terminates. The argument that the number of points in T invisible from C decreases by definition of $U_{\min}(C)$ with every iteration does not imply termination, because the number of points in T is infinitely large, since T is defined over **IR**. The next section will show that only a finite number of points in T, actually only $O(n)$ of them, need to be considered in the construction. The next theorem analyzes the optimality of the solution produced by the incremental construction. In preparing to do so, we state the following elementary fact:

Lemma 2.1.2 *Let c, c' be guards on the line $y = h$ and let $p \in T$ with $p > c > c'$. Then, if p sees c', then p also sees c.*

Proof: Consider the visibility interval $[l(p), u(p)]$ of p. Since T is a terrain, we get $l(p) < p < u(p)$. If p sees c', then $c' > l(p)$. The lemma follows since $l(p) < c' < c < p < u(p)$. □

Theorem 2.1.3 *If the incremental construction terminates, it finds an optimum solution C.*

Proof: Assume there is a solution C' to an instance of the problem that needs fewer guards than the solution C produced by the incremental construction, i.e., $|C| > |C'|$. Let C and C' be ordered according to the x-coordinates of their elements. Let j be the index of the leftmost guard where the solutions disagree. Let $C_i = \{c_1, c_2, \cdots, c_i\}$ and $C'_i := \{c'_1, c'_2, \cdots, c'_i\}$ be the partial solution of the first i guards in C and C', respectively. Let M be the set of all points p in T with $u(p) = U_{\min}(C_{j-1})$ that are not visible from C_{j-1}, i.e. all points in T invisible from C_{j-1} with minimum upper end of the visibility interval. The points in M are exactly the ones that determine the value of $U_{\min}(C_{j-1})$. M contains at least one point, by definition of $U_{\min}(C_{j-1})$. M may contain more than one point, it may even contain a subsegment of T. To prove the theorem, we distinguish two cases.

Case 1: $c_j < c'_j$.

None of the points in M are visible from c'_j, since c'_j is to the right of the visibility intervals of all these points. Since $C_{j-1} = C'_{j-1}$ and since no guard from C_{j-1} sees any of the points in M, no guard from C'_{j-1} can see any of the points on M either. Therefore no guard in C'_j sees the points in M. Since C' is ordered, none of the guards in C' sees the points in M.

Therefore, C' is not a correct solution, a contradiction.

Case 2: $c_j > c'_j$.

We distinguish two subcases. The first case is where all points in M are visible from c'_j. With this assumption one can argue that C' cannot be better than C. By definition, the guards in C_j see all points on T with x-value smaller than $U_{\min}(C_{j-1})$. Therefore, to the left of c_j, the guards in C_j see at least as much as the guards in C'_j. To the right of c_j, on the other hand, the guards of C_j also see at least as much as the guards of C'_j, since the x-monotonicity of the terrain implies that every point to the right of c_j that is visible from c'_j is also visible from c_j (see lemma 2.1.2). Since the guards in C_j see at least all the points that the guards in C'_j see, c'_j might as well be replaced by c_j and thus, $|C'|$ cannot (by induction) be smaller than $|C|$.

The second subcase is where not all points in M are visible from c'_j. In this case one can argue exactly the same way as before, but here it is clear that the guards in C'_j see strictly fewer points than the guards in C_j. To see a corresponding point p in M, C' will have to have another guard no further to the right than the upper end of the visibility interval of p, which is $u(p) = U_{\min}(C_{j-1})$, and therefore, C' will have more guards than C in order to be a correct solution. □

2.1.3 A Linear Time Algorithm

It remains to be shown that the incremental construction can be implemented efficiently. To do so, we will show how to efficiently compute $U_{\min}(C_{j-1})$ and also how to determine whether the guards of C_j see all of the terrain T. This is done by showing that only a finite number of points in T need to be considered. This number will be $O(n)$, and the runtime of the algorithm will be $O(n)$ as well.

Recall that we need to compute $U_{\min}(C_{j-1})$ given a partial solution C_{j-1}, where $U_{\min}(C_{j-1})$ is defined to be the leftmost upper end of the visibility intervals on the horizontal line $y = h$ of all points on the terrain T that are invisible from all guards in the partial solution C_{j-1}. We now make a series of five observations that simplify the computation of $U_{\min}(C_{j-1})$, which leads to an efficient algorithm

A first simplification of the computation of $U_{\min}(C_{j-1})$ comes from the observation that the guards in C_j see all points on terrain T with x-value smaller than $U_{\min}(C_{j-1})$, because the upper end of the visibility interval of a point p on the terrain is always to the right of p. Since it is known that the guards in the partial solution C_{j-1} see all points in T to the left of c_{j-1},

it is sufficient to consider only those points on the terrain T that are to the right of c_{j-1} (and that are invisible from all guards in C_{j-1}).

A second simplification comes from Lemma 2.1.2: If a point on the terrain T to the right of c_j is invisible from c_j, then it is invisible from all other guards in C_j, because c_j is the rightmost guard in C_j.

A third, somewhat more involved observation will limit the points p on the terrain T, of which we have to compute the upper end of the visibility interval, to a number of points on T.

Lemma 2.1.4 *Let p, p' be two points on the segment of terrain T between the two vertices v_i and v_{i+1} with $v_i \leq p \leq p' < v_{i+1}$. Then:*

$$u(p) \geq u(p')$$

Proof: Let v_l be the vertex of terrain T that lies on the line segment that connects p with $u(p)$. There always must exist at least one such vertex by the definition of the visibility interval. If there are more than one such vertices just take the first.

The terrain T connects v_{i+1} with v_l. (Note that $l \geq i+1$.) Assume by contradiction that $u(p') > u(p)$. Consider the line segment from p' to $u(p')$. This line segment must intersect the terrain or reach point p' from underneath the terrain, since it is clearly to the right of the line segment from p to $u(p)$. In both cases, $u(p')$ is invisible from p', a contradiction. □

Lemma 2.1.5 *Let p, p' be two points on the segment of terrain T between the two vertices v_i and v_{i+1} with $v_i < p \leq p' \leq v_{i+1}$. Furthermore, let p and p' be to the right of guard c_{j-1}, i.e., $c_{j-1} \leq p \leq p'$. If guard c_{j-1} sees point p, then it must also see point p'.*

Proof: Assume by contradiction that point p' is invisible from c_{j-1}. Let p'' denote a point on the terrain blocking the view of p', i.e., an intersection point of the line segment from p' to c_{j-1} with the terrain T.

The terrain T, again, connects p with p''. Recall that c_{j-1} lies to the left of p. The line segment from p to c_{j-1} lies entirely to the left of the line segment from p' to c_{j-1}. Therefore, the line segment from p to c_{j-1} must intersect the terrain, which makes p invisible from c_{j-1}, a contradiction. □

Lemma 2.1.6 *If vertex v_i and a point p on the terrain T with $v_i < p \leq v_{i+1}$ are both visible from guard c_{j-1} that is to the right of v_i, then all points p' on terrain T with $v_i \leq p' \leq p$ are visible from c_{j-1}*

Proof: Assume by contradiction that there exists a point p' on the terrain with $v_i \leq p' \leq p$ that is invisible from c_{j-1}. Since v_i is visible, the line segment connecting p' with c_{j-1} must reach p' from underneath the terrain. This, however, implies that a line segment connecting any point on the terrain between vertices v_i and v_{i+1} to c_{j-1} reaches the point on the terrain from underneath. Therefore, point p is also invisible from c_{j-1}, a contradiction. □

Let v_i^- be a conceptual point on the terrain to the left of vertex v_i, but arbitrarily close to v_i, or the vertex itself if T describes a concave function at v_i. Lemma 2.1.4 implies that for each line segment of T to the right of the guard c_{j-1}, it is sufficient to compute the upper end of the visibility interval only of v_i^- or the rightmost point that is invisible from c_{j-1} on the line segment in T. There are four possibilities:

- If v_{i-1} and v_i are both visible from c_{j-1}, then Lemmas 2.1.5 and 2.1.6 say that all points on the line segment between v_{i-1} and v_i are visible from c_{j-1}, and therefore they do not need to be considered when computing the minimum upper end $U_{\min}(C_{j-1})$ of all visibility intervals of points not seen by any guard in C_{j-1}.

- If v_{i-1} and v_i are both invisible from c_{j-1}, then it is clear by Lemma 2.1.5 that all points on the line segment between v_{i-1} and v_i are invisible from c_{j-1}. Because of Lemma 2.1.4, however, it is sufficient to consider the upper end $u(v_i^-)$ of the visibility interval of v_i^- only.

- If v_{i-1} is visible and v_i is invisible from c_{j-1} then, because of Lemmas 2.1.4 and 2.1.5, it is sufficient to consider $u(v_i^-)$.

- If v_{i-1} is invisible and v_i is visible from c_{j-1}, then it is sufficient to consider the rightmost point on the segment that is invisible from c_{j-1}, according to Lemma 2.1.4. We call this point a *split-point.*

If v_{i-1} is invisible and v_i is visible from c_{j-1}, then there exists a split-point on the segment between the two points. Note that the line segment from a split-point to the guard c_{j-1} contains at least one vertex of the terrain T. The leftmost upper end of the visibility intervals $U_{\min}(C_{j-1})$ can be computed by computing the upper ends of the visibility intervals of all points v_i^- that are invisible from c_{j-1} and all split-points produced by c_{j-1}, such that these points are to the right of c_{j-1}.

For a fourth simplification of the computation of $U_{\min}(C_{j-1})$, let p be a point on the terrain T with $v_i < p \leq v_{i+1}$. Let $f(p)$ be the intersection point of the line $y = h$ and the line through vertex v_i. Figure 2.1 illustrates

$f(p)$. Since $f(p)$ will be an approximation for $u(p)$, we are only interested in $f(p)$, if it lies to the left of p, otherwise we let $f(p) := \infty$. With these definitions, obviously $u(p) \leq f(p)$. If this holds with equality, then $u(p)$ is very easy to compute.

Lemma 2.1.7 *For all points v_i^- on the terrain T for which $u(v_i^-) < f(v_i^-)$, the following holds:*

$$u(v_{i+1}^-) \leq u(v_i^-).$$

Proof: If $u(v_i^-) < f(v_i^-)$ and $u(v_i^-) \neq \infty$, the line segment connecting v_i^- and $u(p_i^-)$ must contain at least one vertex v_l of the terrain T. Otherwise, since this line segment obviously intersects T, it would intersect T in a non-vertex-point, and a guard at $u(v_i^-)$ would not be visible from v_i^-, contrary to the definition of $u(v_i^-)$. Let c' be the intersection point of the line going through v_{i+1}^- and v_l and the line $y = h$. Since $v_{i+1}^- \leq v_l$, $u(v_{i+1}^-) \leq c'$. Since $v_i^- \leq v_{i+1}^-$, $c' \leq u(v_i^-)$. The lemma follows. □

Because of Lemma 2.1.7, it is sufficient to approximate the upper end $u(v_i^-)$ with $f(v_i)$. For, if v_{i+1} is visible from c_{j-1}, then there must be a split-point s_l on the segment in T between v_i and v_{i+1}, for which the exact upper end $u(s_l)$ is computed, and because of Lemma 2.1.4 $u(s_l) \leq u(v_i)$. If v_{i+1} is invisible from c_{j-1}, then either $u(v_{i+1}^-) = f(v_{i+1})$ or the lemma can be applied to v_{i+1}.

The definition of $f(p)$ resolves the problem of having to deal with v_i^- instead of v_i and therefore we can compute $U_{\min}(C_{j-1})$ by computing the upper ends of the visibility intervals for the split-points produced by the guard c_{j-1} and the points $f(v_i)$ for all vertices v_i invisible from and to the right of guard c_{j-1}.

For a fifth and final simplification of the computation of $U_{\min}(C_{j-1})$, let $u_p^{p'}$ with $p, p' \in T$ and $p < p'$ be the intersection point of the line $y = h$ and the line containing both p and p'. Note that $u_p^{p'} \geq u(p)$ for all p'.

Lemma 2.1.8 *Let s_i be the i-th split-point to the right of c_{j-1} produced by c_{j-1}. Then for all i the following holds: If $u(s_i) < u(s_{i+1})$, then there exists a terrain vertex v_l with $v_l \leq s_{i+1}$ and $u_{s_i}^{v_l} = u(s_i)$.*

Proof: Figure 2.1 illustrates $u_{s_i}^{v_l}$. The same argument as used in the first part of the proof of Lemma 2.1.7 shows that the value of $u(s_i)$ is determined by a terrain vertex. Therefore, it is sufficient to consider points vertices only, when computing $u_{s_i}^{v_l}$. Let $v_{l'}$ be a terrain vertex on the line from s_i to $u(s_i)$. Let $v_{l''}$ be a terrain vertex on the line from s_{i+1} to $u(s_{i+d})$. Also let $u(s_i) < u(s_{i+1})$. Then $v_{l'} \leq s_{i+1}$, since otherwise $v_{l''} \leq v_{l'}$ (by the argument

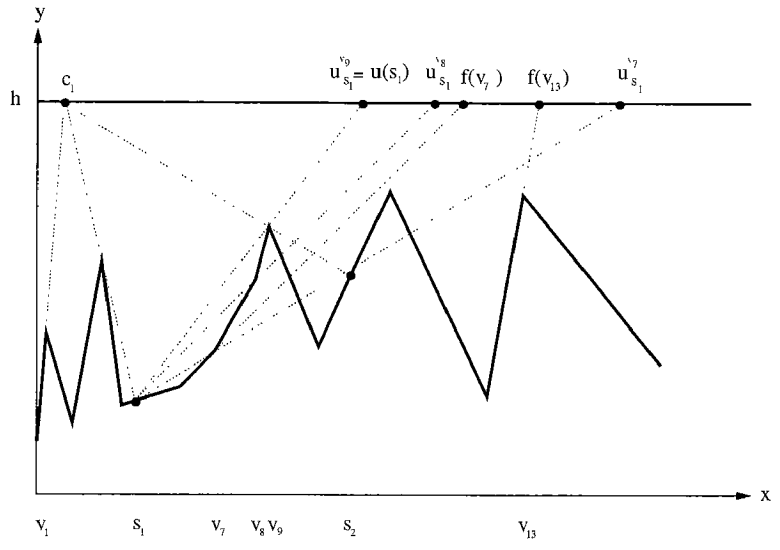


Figure 2.1: Illustration of $f(v_l)$ and $u_{s_i}^{v_l}$

in the second part of the proof of Lemma 2.1.7), and then $u(s_i) > u(s_{i+1})$. The point $v_{l'}$ satisfies the conditions of the point v_l in the lemma. □

Note how the values $u_{s_1}^{p_7} > u_{s_1}^{p_8} > u_{s_1}^{p_9}$ in Figure 2.1 continuously decrease and how the upper end of the approximation $u_{s_1}^{p_9} = u(p_9)$ is reached before the next split-point s_2 is reached.

The lemma says that if $u_{s_i}^{v_l}$ is computed for every terrain vertex v_l with $s_i < v_l < s_{i+1}$, then either $u(s_i)$ is obtained or $u(s_{i+1})$ is smaller than $u(s_i)$ and, therefore, it is sufficient to compute that value.

A combination of Lemma 2.1.7 and Lemma 2.1.8 leads to a simpler computation of $U_{\min}(C_{j-1})$, which summarizes all of our efforts thus far:

$$U_{\min}(C_{j-1}) = \min \left\{ \begin{array}{ll} u_{s_i}^{v_l}: & \text{with } s_i \le v_l \le s_{i+1},\ \forall i \\ f(v_i): & \text{with } v_i > c_{j-1} \text{ and } v_i \text{ invisible from } c_{j-1} \end{array} \right\},$$

where s_i and s_{i+1} is the i-th and $(i+1)$-st split-point produced by the guard c_{j-1} to the right of c_{j-1}, respectively.

This suggests a sweepline algorithm for the computation of $U_{\min}(C_{j-1})$ with stops at each terrain vertex v_i to the right of the guard c_{j-1}. It works as follows:

At each stop, the visibility of the vertex v_i from guard c_{j-1} is determined, which can be done in constant time by maintaining a pointer to the most

recent vertex that was visible while its successor was invisible.

If v_i is invisible, then $f(v_i)$ is computed in constant time, and $f(v_i)$ is compared with the minimum (approximate) upper end $U_{\min}$ found so far.

If the first split-point has already been found, then $u_{s_l}^{v_i}$ of the most recent split-point s_l found is computed and compared with $U_{\min}$. This is also done in constant time.

If v_i is visible but v_{i-1} (from the previous stop) was invisible, then the next split-point s_l is computed in constant time (by using the previously mentioned pointer).

At the first stop, where v_i is to the right of the minimum (approximate) upper end $U_{\min}$ found so far, the minimum has been found and the next guard c_j can be positioned at $U_{\min}$.

This leads to a sweepline algorithm for the computation of an optimum solution C:

First, $U_{\min}(\emptyset)$ is computed and the first guard is positioned at $U_{\min}(\emptyset)$. Then, starting at the first terrain vertex that is to the right of $U_{\min}(\emptyset)$, the position $U_{\min}(C_1)$ of the second guard is computed; this iteration continues until there are no more terrain vertices left.

The algorithm stops at every terrain vertex only once and takes only constant time at each stop. Let us summarize the results of this section.

Theorem 2.1.9 MINIMUM FIXED HEIGHT GUARD ON 1.5 D TERRAIN *can be solved in linear time.*

Of course, a linear time algorithm has an asymptotically optimum running time.

2.1.4 The Set of all Optimum Solutions

For placing communication satellites into the sky, it is certainly interesting to see what flexibility an optimum placement contains. What is the choice of satellite positions, while the number of satellites remains minimum? This choice can be exploited so as to take other factors into account that have not made their way into the objective function.

In particular, it is interesting to see the solution of the symmetric variant of the guarding algorithm that runs "from right to left" instead of "from left to right". That is, instead of computing the leftmost right end $U_{\min}(C_{i-1})$ of the visibility intervals of all points not yet covered by the partial solution C_{i-1}, for a given partial solution B_{i-1} the rightmost left end $L_{\max}(B_{i-1})$ of the visibility intervals of all points not yet covered by B_{i-1} is computed. The solution B obtained this way is also optimal, and it gives some insight into the set of all optimum solutions. C is the "right-most" optimum solution

and B is the "left-most" optimum solution, since the guards c_i (b_j) are positioned as far right (left) as possible. Let $k := |C| = |B|$, and let

$$\begin{aligned} C &= \{c_1 < c_2 < \cdots < c_k\} \\ B &= \{b_1 > b_2 > \cdots > b_k\}. \end{aligned}$$

Lemma 2.1.10 *In an optimum solution C produced by the guarding algorithm, all points on the terrain T with x-values between the x-values of two consecutive guards $c_i, c_{i+1} \in C$ are covered by those two guards c_i and c_{i+1}.*

Proof: This follows directly from Lemma 2.1.2: All points in T are covered by C, and if a guard c' to the right of a point $p \in T$ covers p, then a guard c to the right of p and closer to p than c' covers p as well. The same argument can be used accordingly for guards to the left of p. For all points on the terrain between the guards c_i and c_{i+1}, either c_i or c_{i+1} is the closest guard. □

Lemma 2.1.11 *Let c_i, c_{i+1} be two consecutive guards in a solution C produced by the guarding algorithm. If c_{i+1} is moved to the right, then not all points on the terrain T between the guards c_i and c_{i+1} are covered by those two guards.*

Proof: This follows from the definition of $U_{\min}(C_i)$. □

Lemma 2.1.12 *Consider the union of the two sets of guards C and B. Let the guards in this union be ordered ascendingly according to their x-values. Then this order is:*

$$b_k \leq c_1 \leq b_{k-1} \leq c_2 \leq \cdots \leq b_{k+1-i} \leq c_i \leq \cdots \leq b_1 \leq c_k$$

Proof: Let $b_{k+1-(i+1)}$ lie between c_i and c_{i+1}. According to Lemma 2.1.10, all points on the terrain between c_i and c_{i+1} are covered by those two guards. Since $b_{k+1-(i+1)}$ is to the left of c_{i+1}, all points in T between c_i and $b_{k+1-(i+1)}$ are covered by c_i and $b_{k+1-(i+1)}$. Since b_{k+1-i} is defined to be the point as far left as possible that together with $b_{k+1-(i+1)}$ still covers all points in T inbetween, b_{k+1-i} must be at least as far left as c_i. Therefore the order of the four points must be

$$b_{k+1-i} \leq c_i \leq b_{k+1-(i+1)} \leq c_{i+1}.$$

Consider c_k. Definitely, $b_1 \leq c_k$, since otherwise C would not cover the points that cause the positioning of b_1. Also, $c_{k-1} \leq b_1$, otherwise c_k would

not be necessary, since all points covered by c_k would also be covered by c_{k-1}. Therefore, the three guards on the very right of the terrain can be used as an anchor. The lemma follows by induction. □

To study the set of all optimum solutions, consider the set of guards containing all guards in C from c_1 up to c_j and all guards in B from b_1 up to $b_{k+1-j-1}$. This set contains k guards, and it is an optimum solution. To see this, observe that C_j covers all points to the left of c_j and $B_{k+1-j-1}$ covers all points to the right of $b_{k+1-j-1}$. The points between c_j and $b_{k+1-j-1}$ are covered, since $b_{k+1-j-1} \leq c_{j+1}$ (see Lemma 2.1.10).

Every $b_{k+1-i} \in B$ has a partner $c_i \in C$ which is the one element next to b_{k+1-i} on the right, for $i = 1, \ldots, k$. Partner guards form intervals $[b_{k+1-i}, c_i]$, $i = 1, \ldots, k$ that help to characterize the set of all optimum solutions.

Lemma 2.1.13 *Every optimum solution has exactly one guard in each interval* $[b_{k+1-i}, c_i]$, $i = 1, \ldots, k$.

Proof: For the sake of contradiction, let A be an optimum solution containing a guard a between the two non-partner guards c_{i-1} and b_{k+1-i}. Applying Lemma 2.1.11 inductively starting with c_1, it is easy to see that c_{i-1} is the rightmost point such that the partial solution C_{i-1} covers all points on the terrain T to the left of c_{i-1} using $i-1$ guards. Similarly, b_{k+1-i} is the leftmost point such that the partial solution B_{k+1-i} covers all points to the right of b_{k+1-i} using $k+1-i$ guards.

Because of Lemma 2.1.2, the partial solution of A with all guards to the left of a including a must cover all points on the terrain to the left of a in order to be a correct solution. This cannot be done with less than i guards. Similarly, the partial solution of A with all guards to the right of a including a must cover all points on the terrain to the right of a in order to be a correct solution. This cannot be done with less than $k+1-i+1$ guards. Therefore A must contain at least (considering that a is counted twice): $i + (k+1-i+1) - 1 = k+1$ guards, which is not optimal, a contradiction. □

Note that the converse of Lemma 2.1.13 is not true: There are of course placements of one guard per interval $[b_{k+1-i}, c_i]$ that do not cover the terrain. For a correct partial solution that covers one end of the terrain, however, a completion can be computed in linear time, as the following lemma shows. Here, a partial solution $A_i = \{a_1 < a_1 < \cdots < a_i\}$ is called a *correct partial solution* iff A_i covers all points on the terrain T to the left of a_i.

Lemma 2.1.14 *Let A_i be a correct partial solution, with guard $a_j \in A_i$ in the interval $[b_{k+1-j}, c_j]$. If the guarding algorithm is used accordingly to complete the solution A_i, the produced solution A is optimal.*

Proof: To find the guard a_{i+1}, the guarding algorithm computes $U_{\min}(A_i)$ and sets the x-value of a_{i+1} to $U_{\min}(A_i)$. Note that a_{i+1} lies in the interval $[b_{k+1-(i+1)}, c_{i+1}]$. To see this, consider that all points on T between b_{k+1-i} and $b_{k+1-(i+1)}$ are covered by these two guards (see Lemma 2.1.10). Since $b_{k+1-i} \leq a_i$, all points on T between a_i and $b_{k+1-(i+1)}$ are covered by these two guards (see Lemma 2.1.2). This means that $b^x_{k+1-(i+1)} \leq U_{\min}(A_i) = a^x_{i+1}$. On the other hand, the guards c_i and c_{i+1} cover all points from T in their interval and according to Lemma 2.1.11 c_{i+1} cannot move to the right. Since $a_i \leq c_i$, a_i covers fewer points on T in the interval $[c_i, c_{i+1}]$ (see Lemma 2.1.2). Therefore, $a^x_{i+1} = U_{\min}(A_i) \leq c^x_{i+1}$. This means that a correct partial solution A_{i+1} with each guard $a_j \in A_{i+1}$ in the interval $[b_{k+1-j}, c_j]$ is obtained. The same argument can be used inductively to obtain a complete optimum solution. □

Note that of course the situation is symmetric for a partial solution that is correct from the right end of the terrain. Figure 2.2 shows in an example partner guards; the intervals in which the guards have to be placed in any optimum solution are marked by a thick line.

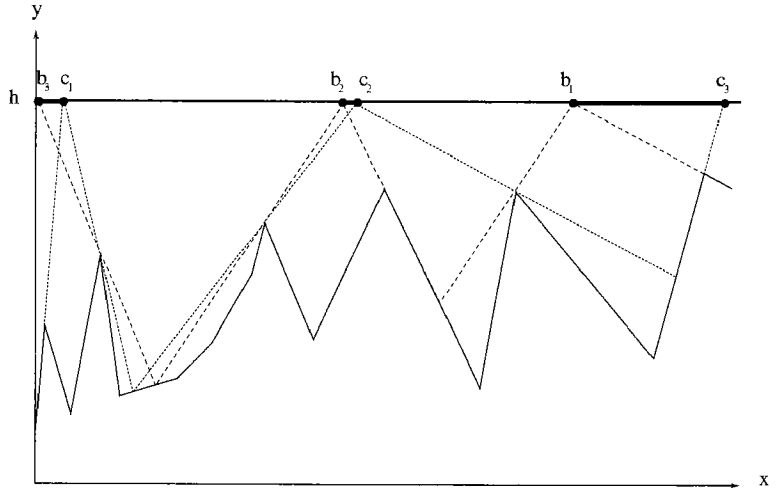


Figure 2.2: Partner intervals

2.2 Guarding Polygons without Holes

In this section, we propose a reduction from MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY to MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES, analyze it and show that it is gap-preserving. This implies that MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES is APX-hard. We also show that APX-hardness follows for the problems MINIMUM BOUNDARY RESTRICTED VERTEX/EDGE GUARD WITHOUT HOLES and MINIMUM VERTEX/EDGE/POINT GUARD WITHOUT HOLES.

2.2.1 Construction of the Reduction

We present the construction for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES. Suppose we are given an instance I of MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY. Let I consist of n variables $x_1, \cdots, x_n$ and of $m \leq \frac{5}{3}n$ clauses $c_1, \cdots, c_m$. Taking instance I as input, we construct a polygon T, which is an instance I' of MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES.

Overview

The polygon T contains six different kinds of basic units, called patterns. These are called *literal, clause, variable, ear, body,* and *spike patterns.* Each pattern is a polygon, which will be part of the final polygon T. We obtain the final polygon by taking the union of all patterns. Each pattern (except for the body pattern) contains a *distinguished tuple.*

Definition 2.2.1 *A* distinguished tuple (p_i, p_j, p_k, p_l) *of a pattern is formed by the four vertices* p_i, p_j, p_k, p_l *of the pattern with the following properties:*

- p_i *and* p_j *are neighboring vertices (i.e.,* $j = i + 1$ *or* $j = i - 1$*)*
- *Any guard that sees an arbitrarily small part of the edge from* p_i *to* p_j*, which includes vertex* p_i*, must lie inside or on the boundary of the pattern, even if we consider the edges of the pattern, which form the path between the vertices* p_k *and* p_l *(not taking the route that includes vertex* p_i*), to be transparent.*

As an example, consider the polygon in Fig. 2.4, in which (q_6, q_5, q_8, q_1) is a distinguished tuple. That fact that the edge from q_1 to q_8 is transparent is indicated through a dashed line. Polygon components will be composed by attaching them to each other at transparent edges; these edges will, hence, disappear in the composition, and their transparency indicates just this.

Definition 2.2.2 *Let (p_i, p_j, p_k, p_l) be a distinguished tuple of a pattern. Then, an arbitrarily small part of the edge from p_i to p_j starting at p_i and going towards p_j is called* distinguished arrow.

Distinguished tuples and distinguished arrows will help us define an algorithm that obtains a truth assignment for the variables of I, if it is given a solution (i.e. a set of guards) of I'.

The reduction works as follows: For each literal, we construct a literal pattern, each of which contains a vertex T^{lit} and a vertex F^{lit}, which corresponds to the truth value of the literal, if a guard sits there. Three literal patterns form a clause pattern in such a way that the clause pattern can only be guarded by a minimum number of guards, if at least one literal in the clause is true. We construct a variable pattern for each variable, which contains a vertex T^{var} and a vertex F^{var}, which corresponds to the truth value of the variable. Finally, spike patterns are used to connect variable and literal patterns in such a way that a minimum number of guards is only possible, if the truth values are assigned consistently.

We first introduce the literal, clause, variable, ear, and body patterns. We then show how these patterns are put together, and finally we define spike patterns.

Literal Pattern

Let $l_j(c_i)$, for $j = 1, \ldots, 3$ and $i = 1, \ldots, m$ denote the j-th literal of the i-th clause. Note that $l_j(c_i) = x_k$ or $l_j(c_i) = \neg x_k$ for some $k = 1, \ldots, n$. For each literal $l_j(c_i)$, we construct a literal pattern as shown in Fig. 2.3. The literal pattern is the polygon defined by the points $p_1(l_j(c_i)), \ldots, p_6(l_j(c_i))$, given in counterclockwise order as shown in Fig. 2.3. Whenever it is clear which literal we are talking about, we will denote vertex $p_k(l_j(c_i))$ simply by p_k, omitting the argument, as done in Fig. 2.3. The edge from p_6 to p_1 is not part of the final polygon, but serves as an interface to the outside of the literal pattern. We will lose this edge when we form the union of the literal pattern with a clause pattern. As before, the transparent edge from p_1 to p_6 is drawn as a dashed line. All other edges in the literal pattern are part of the final polygon. The points p_4, p_5, p_1 are collinear. Note that a guard at point p_1 or point p_5 sees all of the interior of the literal pattern. The final construction will be such that a guard at point p_1 implies that the literal is true and a guard at point p_5 implies that the literal is false. We, therefore, call point $p_1(l_j(c_i))$ simply $T^{lit}(l_j(c_i))$; similarly, $p_5(l_j(c_i))$ is called $F^{lit}(l_j(c_i))$. Note that (p_4, p_3, p_1, p_6) is a distinguished tuple. The distinguished arrow (p_4, p_3) is marked by an arrow in Fig. 2.3.

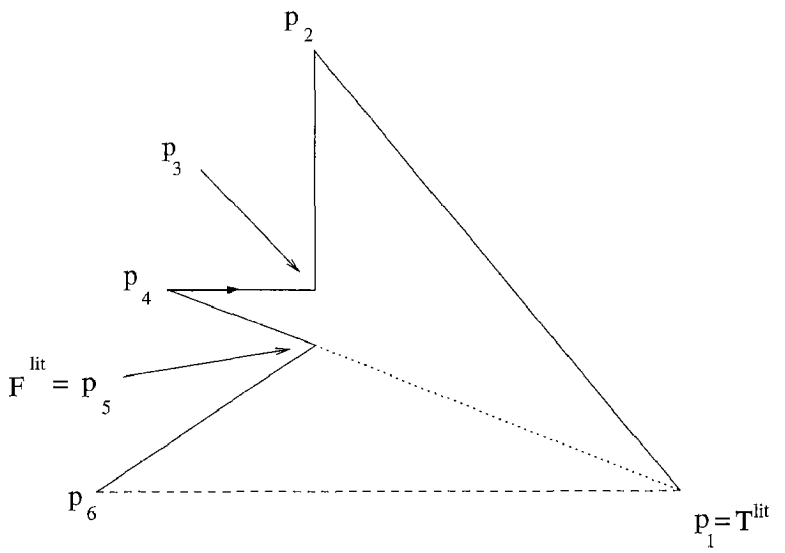


Figure 2.3: Literal Pattern

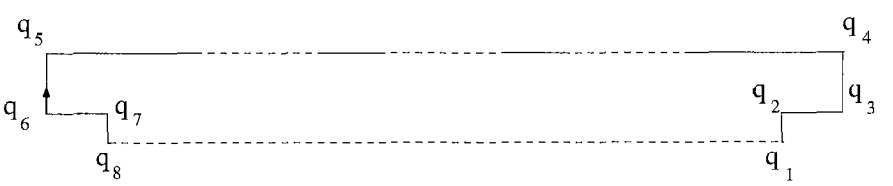


Figure 2.4: Clause Pattern

Clause Pattern

For every clause c_i, we construct a clause pattern as shown in Fig. 2.4. The clause pattern for c_i is the polygon defined by the vertices $q_1(c_i), \cdots, q_8(c_i)$. Vertices q_6, q_7, q_2 and q_3 are collinear. The tuple (q_6, q_5, q_1, q_8) is a distinguished tuple.

We form the union of the clause pattern of clause c_i and the three literal patterns $l_1(c_i)$, $l_2(c_i)$, and $l_1(c_i)$ as indicated in Fig. 2.5. Note that this is done in such a way that a guard at vertex T^{lit} of any of the three literals sees the distinguished arrow of the clause pattern, while a guard at a vertex F^{lit} of any literal pattern cannot see the distinguished arrow of the clause pattern.

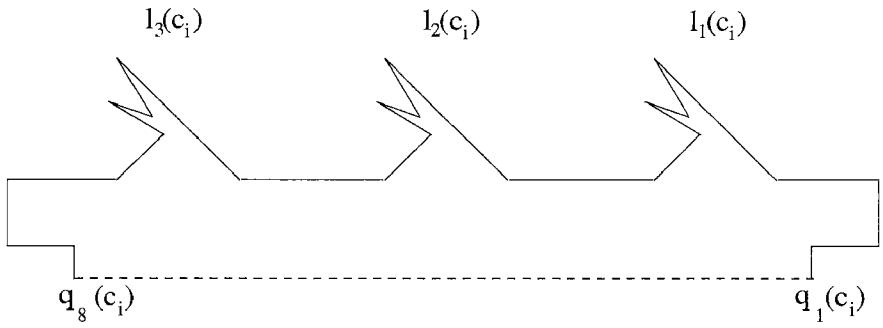


Figure 2.5: Union of clause pattern and literal patterns

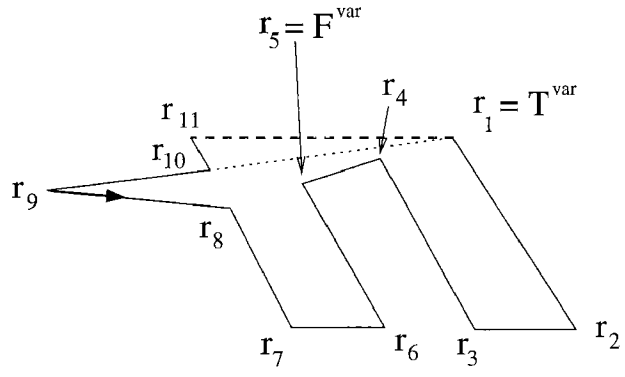


Figure 2.6: Variable Pattern

Variable Pattern

For every variable x_k, we construct a variable pattern as shown in Fig. 2.6. The variable pattern is the polygon defined by the vertices

$$r_1(x_k), \cdots, r_{11}(x_k)$$

. We call the polygon defined by the vertices r_1, r_2, r_3, r_4 the *TRUE leg* of the variable pattern and the polygon defined by r_5, r_6, r_7, r_{11} the *FALSE leg* of the variable pattern. The vertices r_9, r_{10}, r_1 are collinear, and so are vertices r_7, r_8, r_{10}, r_{11}, and also vertices r_1, r_4, r_5, r_8. The shape of the variable pattern can be changed slightly (as will be done in the final construction), as long as the collinearities are maintained. The tuple (r_9, r_8, r_1, r_{11}) is a distinguished tuple.

In the final polygon it will turn out that a guard sits at point r_1, if the

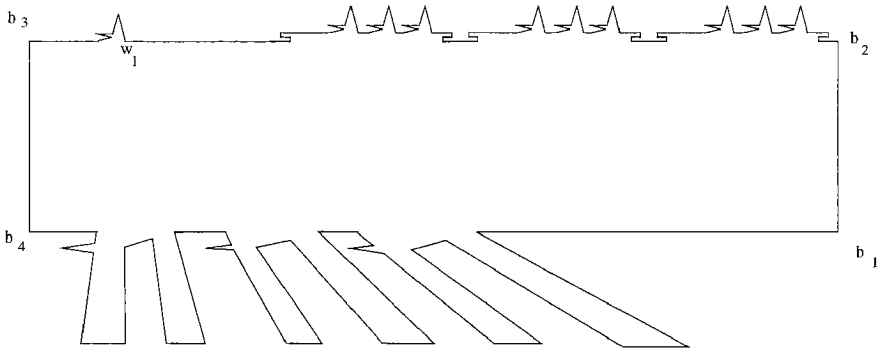


Figure 2.7: Putting the pieces together

variable is assigned the value true, and it sits at point r_5, if the variable is false. Therefore, we define: $T^{var}(x_k) := r_1(x_k)$ and $F^{var}(x_k) := r_5(x_k)$.

Ear Pattern

The ear pattern is necessary for technical reasons. Its use will become evident in the analysis of the reduction. An ear pattern is the same as a literal pattern. However, it is not associated with any literal. We use the same numbering as for the literal pattern and denote the vertices of the ear pattern by w_k for $k = 1, \ldots, 6$.

Body Pattern

The body pattern is a rectangle with vertices $b_1, \cdots, b_4$. These vertices are shown in Fig. 2.7.

Forming the Union of the Components

We put all pieces together as shown in Fig. 2.7. The legs of the variable patterns are such that a guard at point w_1 sees all the legs of the variable patterns. We call the polygon obtained at this stage T'.

Spike Patterns

A spike pattern s is a triangle shaped polygon with some additional vertices on the edges. In the final polygon, there will be one spike pattern for each vertex T^{lit} and F^{lit}, which are of slightly different types. Figure 2.8 (a) shows the type of spike patterns for vertices T^{lit}, which we call TRUE spike

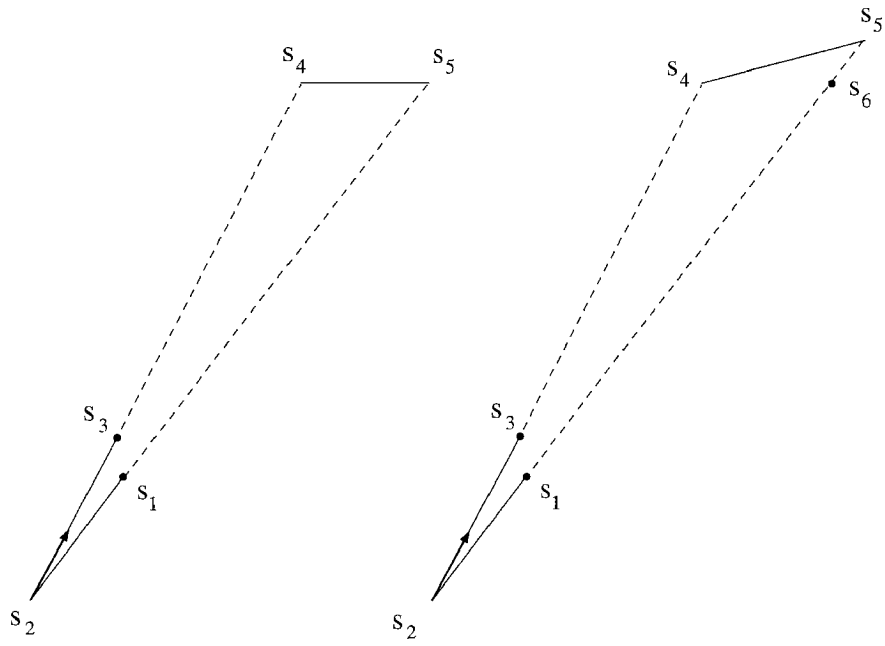


Figure 2.8: (a) TRUE Spike Pattern; (b) FALSE Spike Pattern

pattern; Figure 2.8 (b) shows the type of spike patterns for vertices F^{lit}, which we call FALSE spike pattern. The spike pattern s is the polygon with vertices $s_1, \cdots, s_5, (s_6)$. We have the following collinearities:

- Vertices s_2, s_3, s_4 are collinear.
- Vertices $s_2, s_1, (s_6), s_5$ are collinear.

The tuple (s_2, s_3, s_1, s_3) is a distinguished tuple, if we change the definition in such a way that the edge from s_4 to s_5 is not transparent, as indicated in Figs. 2.8 (a) and (b) and in such a way that the view of a guard that is located on the extension of the line from s_2 through s_4 is blocked by the vertex s_4.

Adding Spikes to the Construction

We form the union of the spikes with the polygon T' as follows: We construct for each literal $l_j(c_i)$ in each clause two spike patterns (one TRUE and one

FALSE spike pattern) as shown in Figs. 2.9 and 2.10.

Figure 2.9 is for the case, when literal $l_j(c_i)$ is positive (i.e. $l_j(c_i) = x_k$, for some k). In this case, we have a TRUE spike pattern

$$s(T^{lit}(l_j(c_i)), F^{var}(x_k)),$$

which connects vertex T^{lit} of the literal pattern $l_j(c_i)$ with vertex F^{var} of the variable pattern x_k, and a FALSE spike pattern

$$s(F^{lit}(l_j(c_i)), T^{var}(x_k)),$$

which connects vertex F^{lit} of the literal pattern $l_j(c_i)$ with vertex T^{var} of the variable pattern x_k.

Figure 2.10 is for the case, when literal $l_j(c_i)$ is negative (i.e. $l_j(c_i) = \neg x_k$, for some k). In this case, we have a TRUE spike pattern

$$s(T^{lit}(l_j(c_i)), T^{var}(x_k)),$$

which connects vertex T^{lit} of the literal pattern $l_j(c_i)$ with vertex T^{var} of the variable pattern x_k, and a FALSE spike pattern

$$s(F^{lit}(l_j(c_i)), F^{var}(x_k)),$$

which connects vertex F^{lit} of the literal pattern $l_j(c_i)$ with vertex F^{var} of the variable pattern x_k.

For each TRUE spike pattern s, we have the following, where p_6 and T^{lit} are vertices of the corresponding literal pattern:

- $s_4 = T^{lit}$
- s_4, s_5, and p_6 are collinear.

For each FALSE spike pattern s, we have the following, where p_6 and T^{lit} are vertices of the corresponding literal pattern:

- $s_5 = F^{lit}$
- $s_4 = p_6$
- s_4, s_6, and T^{lit} are collinear.

For each spike pattern $s(T^{lit}, F^{var})$ or $s(F^{lit}, F^{var})$ we have the following collinearities, where q_7, q_8 and F^{var} are vertices of the corresponding variable pattern:

- s_1, F^{var}, s_5 are collinear.

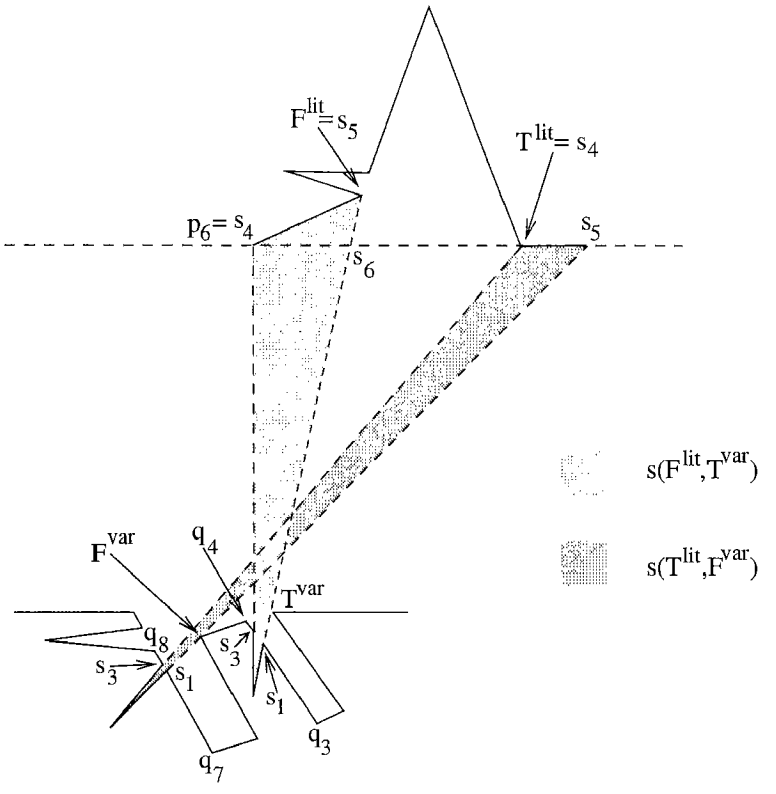


Figure 2.9: Spike patterns for positive literal

- q_7, s_1, s_3, q_8 are collinear.

For each spike pattern $s(T^{lit}, T^{var})$ or $s(F^{lit}, T^{var})$ we have the following collinearities, where q_3, q_4 and T^{var} are vertices of the corresponding variable pattern:

- s_1, T^{var}, s_5 are collinear.

- q_3, s_1, s_3, q_4 are collinear.

As a result, we obtain the polygon T, which is the instance I' of MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES.

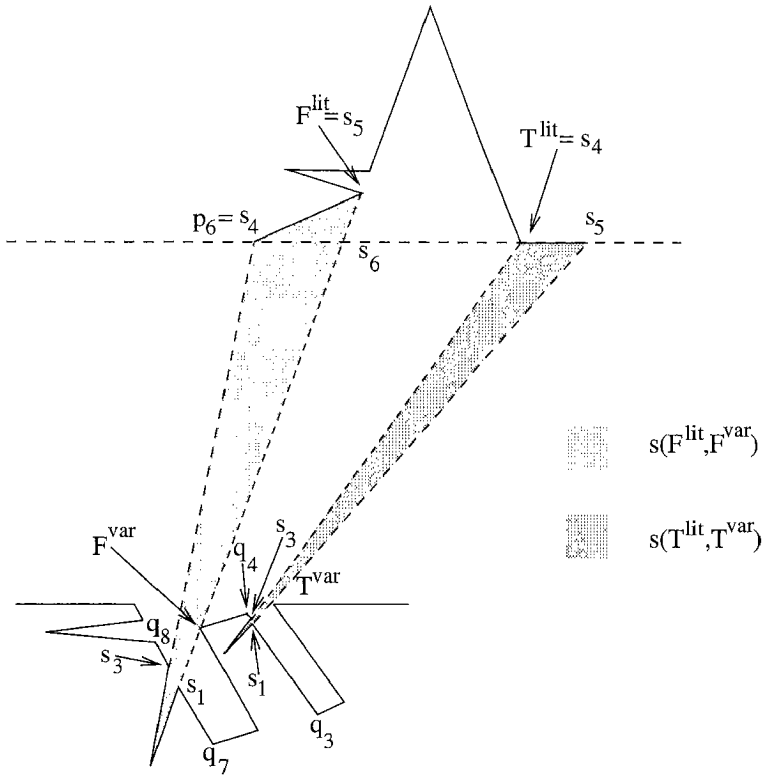


Figure 2.10: Spike patterns for negative literal

Feasibility of the Construction

Remember that, in order to see an arbitrarily small part of the edge from s_2 to s_3 including s_2 of each spike pattern s, a guard must lie in the interior or on the boundary of s, because (s_2, s_3) is a distinguished arrow.

In order to prove our inapproximability result for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES, we must ensure that the following holds:

Lemma 2.2.3 *Instance I' of* MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES *(i.e. the polygon T) can be constructed from the* MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY *instance I in such a way that no three spike patterns that connect literal patterns to three different legs (of the variable patterns) intersect in a common point.*

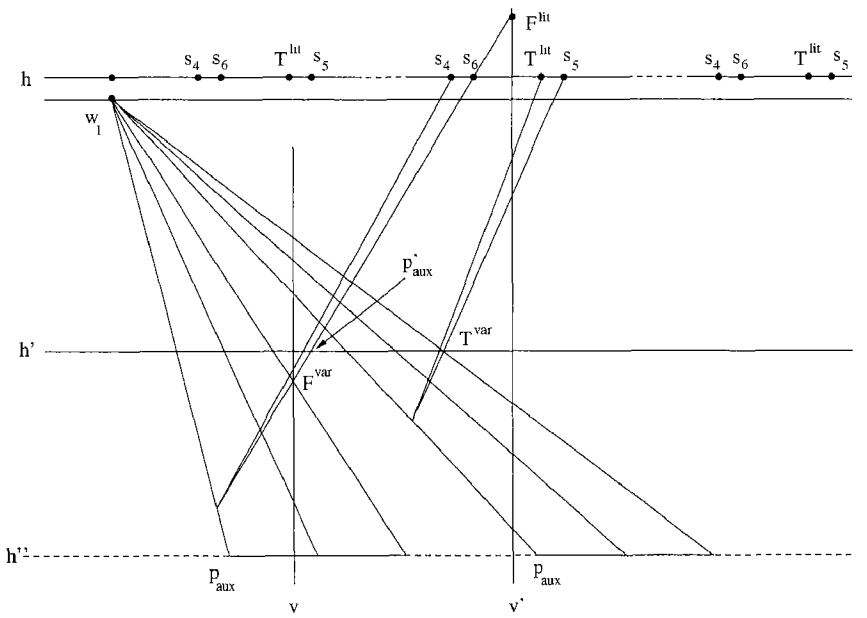


Figure 2.11: Detailed construction

Proof: We prove the lemma by giving a detailed description of how this can be achieved. An overview of the construction is given in Fig. 2.11.

We start with the first literal pattern $l_1(c_1)$. Fix vertex $s_5(T^{lit}(l_1(c_1)), \cdot)$ of the TRUE spike pattern on a horizontal line h. Then set vertex $T^{lit}(l_1(c_1))$ at a distance $a(I)$ to the left of $s_5(T^{lit}(l_1(c_1)), \cdot)$ on the horizontal line h. Fix vertex $s_6(F^{lit}(l_1(c_1)), \cdot)$ of the FALSE spike pattern at a constant distance to the left of $T^{lit}(l_1(c_1))$ on h, then set vertex $s_4(F^{lit}(l_1(c_1)), \cdot)$ at distance $a(I)$ to the left of $s_6(F^{lit}(l_1(c_1)), \cdot)$ on h. Then fix vertex $s_5(T^{lit}(l_2(c_1)), \cdot)$ of the TRUE spike pattern of the second literal pattern at constant distance to the left of $s_4(F^{lit}(l_1(c_1)), \cdot)$ on h and repeat the procedure for all literals.

Note that $a(I)$ depends on the instance (i.e. $a = a(I)$). Choose w_1 (of the ear pattern) at a constant distance to the left of point $s_5(F^{lit}(l_3(c_m)), \cdot)$ (of the leftmost literal) and at distance $a'(I)$ below the line h.

Assume that the variable patterns for the variables $x_1, \cdots, x_{k-1}$ have already been constructed, that the vertices T^{var} of all of these variable patterns lie an the same horizontal line h', which is at constant distance from h, and that the vertices r_2, r_3, r_6, r_7 all lie on the same horizontal line h'', which is at distance $a''(I)$ from h'. We show how to construct the

next variable pattern for variable x_k. We determine point $T^{var}(x_k)$ of the variable pattern as follows:

Determine the rightmost of the (at most five) literal patterns that is a literal of x_k. Let us assume that it is a negative literal $l_j(c_i)$. (The case, where it is a positive literal can be treated similarly.)

Set vertex $T^{var}(x_k)$ on the horizontal line h' to the left of all the variable patterns already constructed in such a way that there exist no areas, where two spike patterns connecting literals with two different legs of variable patterns, intersect, to the left or on the line from $T^{var}(x_k)$ to $s_5(T^{lit}(l_j(c_i)), T^{var}(x_k))$.

The intersection of the line h'' with the line from w_1 through T^{var} yields vertex r_2. Fix r_3 at constant distance to the left of r_2 on h'' and fix some (auxiliary) point p_{aux} at constant distance to the left of r_3 on h''. The intersection of the line from s_5 through T^{var} with the line from w_1 to p_{aux} yields vertex s_2; it yields vertex s_1 if intersected with the line from w_1 to r_3 . Intersecting the line from w_1 to p_{aux} with the line from T^{lit} to s_2 gives vertex s_3. Thus, we have constructed the TRUE leg with the first spike pattern. Now, construct all remaining spike patterns for the leg. Note that their vertices s_2 are strictly below vertex s_2 of the first spike pattern constructed. Also note that the distance $a(I)$ must be chosen small enough, such that no two spike patterns intersect to the left of the line from r_3 to w_1.

We construct the FALSE leg in a similar way, however, we need an auxiliary point p'_{aux}, which is set on the horizontal line h' in such a way that there exist no areas, where two spike patterns connecting literals with two different legs of variable patterns intersect, to the left or on the line from p'_{aux} to the vertex s_5 (or s_6) of the spike pattern that connects the rightmost literal pattern that represents a literal of the variable. Let v be a vertical line at some constant distance to the left of p'_{aux}. Vertex F^{var} is the intersection point of v with either the line from s_5 (or s_6) through p'_{aux} or the line from T^{var} through vertex s_3 of the top-most spike pattern in the TRUE leg, whichever is closer to the horizontal line h'. The remaining vertices of the leg and the spike patterns are then constructed as in the TRUE leg.

The variable pattern can now be completed by just observing the required collinearities.

Once we have constructed all variable patterns, we need to construct the literal patterns. For each literal, proceed as follows:

Let v' be a vertical line at some constant distance to the right of vertex s_6 of the FALSE spike pattern of the literal. Vertex F^{lit} is the intersection of v' with the line from s_1 through s_6 of the FALSE literal pattern. Construct

the remaining vertices of the literal pattern straight-forward by observing the required collinearities.

We complete the clause pattern in a straight-forward manner observing all collinearities and the requirement that a guard at a vertex F^{lit} of some literal pattern may not see the distinguished arrow of the corresponding clause pattern. Vertices q_1 and q_8 of each clause pattern are on the same horizontal line as w_1, which is at distance $a'(I)$ from line h. Note that $a'(I)$, therefore, must be chosen small enough such that the polygonal chains from q_1 to q_4 and from q_5 to q_8 do not intersect any spike patterns.

We complete the construction as indicated in Fig. 2.7.

An analysis reveals that the coordinates of all points can be computed in polynomial time; some coordinates require a polynomial number of bits. The analysis is similar to the analysis for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES, which is given in full detail in Sect. 2.3. Therefore, the construction is polynomial in the size of the input.

□

2.2.2 Transformation of a Feasible Solution

We describe how to obtain an assignment of the variables of the satisfiability instance (i.e. a solution of I), given a feasible solution of the corresponding MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES instance I'. We move guards in such a way that the set of distinguished arrows that a guard sees changes in only one of two ways: The first possibility is that the set remains the same or contains some additional distinguished arrows. The second possibility is that some distinguished arrows are removed from the set, but then it is ensured that some other guards see the distinguished arrows that were removed. The guards are moved as follows, where we have illustrated some of these movements in Fig. 2.12:

- Determine which guard is inside the ear pattern $w_1, \cdots, w_6$ (see Fig. 2.7) and move this guard to w_1 as indicated at ① in Fig. 2.12.
- For each literal pattern, determine which guard sees the distinguished arrow (such a guard must be inside the literal pattern). If this guard is at vertex F^{lit}, then leave it there, otherwise, move it to vertex T^{lit} as indicated at ② in Fig. 2.12.
- If there is a guard at both vertices T^{lit} and F^{lit} of the literal pattern, move the guard at F^{lit} along the edge of the FALSE spike pattern towards vertex s_2 to the vertex F^{var} or T^{var} of the corresponding

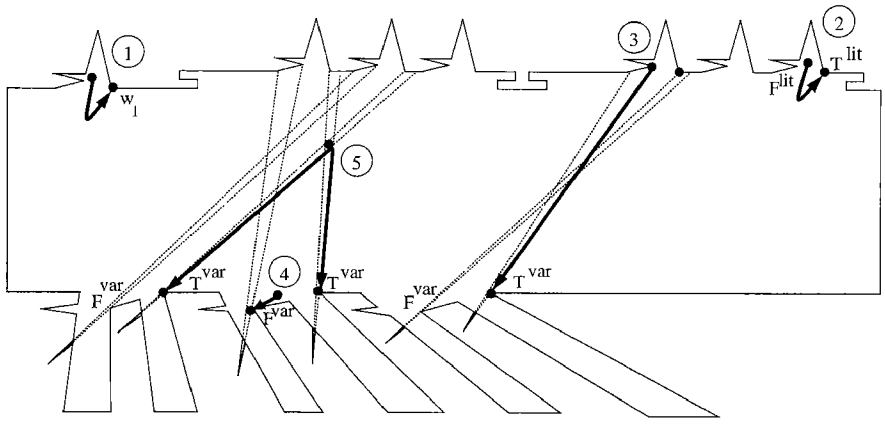


Figure 2.12: Different types of guard moves

variable pattern as indicated at ③ in Fig. 2.12; note that the guard at T^{lit} does not move.

- If there is more than one guard at some vertex T^{lit} (F^{lit}), move all but one guard along the edge of the TRUE (FALSE) spike pattern towards vertex s_2 to the vertex F^{var} or T^{var} of the corresponding variable pattern.

- For each clause pattern, move any guard that sees the distinguished arrow of the clause pattern and lies in a TRUE spike pattern to the vertex T^{lit} that is on the boundary of the spike pattern.

- For each clause pattern, consider a guard g that sees the distinguished arrow of the clause pattern and lies in a FALSE spike pattern. If there already is a guard g' at vertex F^{lit} of the spike pattern, then move the guard g to vertex T^{lit}, otherwise, move the guard g to vertex F^{lit}.

- For each variable pattern, move the guard that sees the distinguished arrow of the variable pattern to vertex T^{var}, if it also lies in a spike pattern that contains vertex T^{var}, and move it to point F^{var}, otherwise, as indicated at ④ in Fig. 2.12.

- Move all guards that lie in a spike pattern but do not see the distinguished arrows of any literal or clause pattern to vertex T^{var} or F^{var} that is contained in the spike pattern.

- If a guard sees the distinguished arrows of two spike patterns that connect literals to two different legs of variable patterns, add a guard and move one guard each to the two vertices T^{var} or F^{var} of the variable patterns that lie on the boundary of the two spike patterns as indicated at ⑤ in Fig. 2.12. (Note that because of Lemma 2.2.3, no guard can see the distinguished arrows of three spike patterns that belong to three different legs.)
- Guards that do not see any distinguished arrows are moved to any point T^{var} or F^{var} of any variable pattern, if there is no guard there already.

This procedure is iterated until all guards are at their final position. The solution obtained after moving and adding guards as indicated is still feasible. To see this, note that after this procedure there is exactly one guard in each literal pattern at either point F^{lit} or T^{lit}. In each clause pattern c_i there is at least one guard at either $T^{lit}(l_1(c_i))$, $T^{lit}(l_2(c_i))$, or $T^{lit}(l_3(c_i))$. Therefore, all literal and clause patterns are guarded. The remaining polygon (except for parts of the spike patterns) is guarded by the guard at point w_1 of the ear. Finally, the spike patterns are guarded, since all guards that saw the distinguished arrow of a spike pattern have been moved only within the spike pattern. Where such a guard saw two distinguished arrows of two spike patterns, we have added a guard.

We are now ready to set the truth values of the variables. For each variable pattern x_k, if there is a guard at point $F^{var}(x_k)$ and no guard at point $T^{var}(x_k)$, let x_k be false. If there is a guard at point $T^{var}(x_k)$ and no guard at point $F^{var}(x_k)$, let x_k be true. If there is a guard at both $T^{var}(x_k)$ and $F^{var}(x_k)$, then set x_k in such a way that a majority of the literals of x_k become true.

2.2.3 Analysis of the Reduction

We first prove two lemmas that will help us prove the APX-hardness of MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES.

Lemma 2.2.4 *If an instance of* MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY *with n variables and $m \leq \frac{5}{3}n$ clauses is satisfiable (i.e., all m clauses are satisfied), then there exists a feasible solution of the corresponding instance of* MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES *with $3m + n + 1$ guards.*

Proof: Fix any truth assignment of the variables that satisfies the MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY instance. Place one guard at w_1.

For each variable x_k, place a guard at point $F^{var}(x_k)$ of the variable pattern if x_k is false. Place a guard at $T^{var}(x_k)$, if x_k is true. For each literal $l_j(c_i)$ in each clause, place a guard at point $T^{lit}(l_j(c_i))$ of the literal pattern, if the literal is true. Place a guard at point $F^{lit}(l_j(c_i))$, if the literal is false. This solution is feasible and consists of $3m + n + 1$ guards.

□

Lemma 2.2.5 *If there exists an $\epsilon > 0$ and a feasible solution of the* MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES *instance I' with $3m+n+1+\epsilon m$ guards, then there exists an assignment of the variables of the corresponding* MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY *instance I that satisfies at least $m(1-4\epsilon)$ clauses.*

Proof: In the feasible solution of MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES, there must be at least one guard inside each literal pattern. There also must be at least one guard inside each variable pattern. Finally, one additional guard is needed at w_1.

Now, move the guards according to the transformation given in Section 2.2.2. At most ϵm guards see the distinguished pairs of two spike patterns belonging to different legs, since $3m$ guards are inside literal patterns, n guards are inside variable patterns and one guard is at w_1. Therefore, we have at most ϵm additional guards.

We now set the truth values of the variables according to the transformation. For at least $n - 2\epsilon m$ variable patterns, there is only one guard at either $T^{var}(x_k)$ or $F^{var}(x_k)$. For at most $2\epsilon m$ variable patterns, there is a guard at both points $T^{var}(x_k)$ and $F^{var}(x_k)$. When we set the truth value of each of these $2\epsilon m$ variables, at most 2 clauses will be unsatisfied for each variable. Therefore, we have at most $4\epsilon m$ unsatisfied clauses.

□

Now, consider the promise problem of MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY, where we are given an instance of MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY, and we are promised that the instance is either satisfiable or at most $m(1-4\epsilon)$ clauses are satisfiable by any assignment of the variables. This problem is NP-hard for small enough values of ϵ (see Sect. 1.3).

By Lemma 2.2.4 and by the contraposition of Lemma 2.2.5, we obtain the following theorem.

Theorem 2.2.6 *Let I be an instance of the promise problem of* MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY, *let n be the number of variables in I*

and let $m \leq \frac{5}{3}n$ *be the number of clauses in* I. *Let* $OPT(I)$ *denote the maximum number of satisfiable clauses (for any assignment). Furthermore, let* I' *be the corresponding instance of* MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES *and let* $OPT(I')$ *denote the minimum number of guards needed to cover* I'. *Then, the following hold:*

- *If* $OPT(I) = m$, *then* $OPT(I') \leq 3m + n + 1$.
- *If* $OPT(I) \leq m(1 - 4\epsilon)$, *then* $OPT(I') \geq 3m + n + 1 + \epsilon m$.

Theorem 2.2.6 shows that our reduction is gap-preserving (see [3]). It shows that the promise problem of MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES with parameters $3m + n + 1$ and $3m + n + 1 + \epsilon m$ is NP-hard. Note that $m \geq \frac{n}{3}$, since each variable appears as a literal at least once. Therefore, unless $NP = P$, no polynomial time approximation algorithm for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES can achieve an approximation ratio of:

$$\frac{3m + n + 1 + \epsilon m}{3m + n + 1} = 1 + \frac{\epsilon}{3 + \frac{n+1}{m}} \geq 1 + \frac{\epsilon}{3 + \frac{3(n+1)}{n}} \geq 1 + \frac{\epsilon}{7}$$

Thus, we have the following result:

Theorem 2.2.7 MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES *is APX-hard.*

Our proof works as well for MINIMUM POINT GUARD WITHOUT HOLES. To see this note that, if we are given a solution of the satisfiability instance and set the guards as indicated in Lemma 2.2.4, the guards see all of the interior and the boundary of the polygon. If we are given a solution of the MINIMUM POINT GUARD WITHOUT HOLES instance and perform the transformation as given in Sect. 2.2.2, the guards still see all of the interior and the boundary of the polygon. Therefore, we have:

Theorem 2.2.8 MINIMUM POINT GUARD WITHOUT HOLES *is APX-hard.*

2.2.4 Inapproximability Results for MINIMUM (BOUNDARY RESTRICTED) VERTEX/EDGE GUARD WITHOUT HOLES

The proof for the APX-hardness of MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITHOUT HOLES can be copied from the corresponding proof for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT

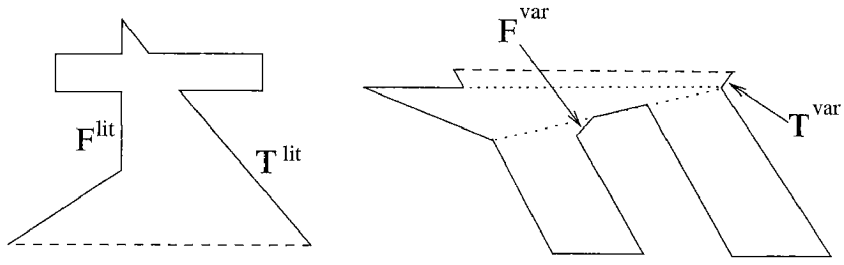


Figure 2.13: (a): Literal Pattern for MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITHOUT HOLES; (b): Variable Pattern for MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITHOUT HOLES

HOLES. Actually, we do not even need the property in the constructed polygon that no three spike pattern of three different legs intersect (see Lemma 2.2.3). Since there are no guards added in the transformation for MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITHOUT HOLES, we would get a slightly bigger constant for the inapproximability of MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITHOUT HOLES than the constant for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES, if we were interested in giving explicit constants.

Thus, we obtain:

Theorem 2.2.9 MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITHOUT HOLES *is APX-hard.*

Our proof carries over to MINIMUM VERTEX GUARD WITHOUT HOLES using the same arguments as for MINIMUM POINT GUARD WITHOUT HOLES. Therefore, we have:

Theorem 2.2.10 MINIMUM VERTEX GUARD WITHOUT HOLES *is APX-hard.*

The proof for the APX-hardness of MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITHOUT HOLES follows the lines of the corresponding proof for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES with some modifications. The literal pattern and the variable pattern are slightly different as shown in Figure 2.13. Note that F^{lit}, T^{lit}, F^{var}, and T^{var} are edges in the literal pattern.

The ideas of the proof for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES can now be applied here. Any solution of the MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITHOUT HOLES instance contains at least $3m + n + 1$ guards. If we are given a solution of the MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITHOUT HOLES instance we can adopt the transformation procedure described in Sect. 2.2.2. Therefore, we obtain:

Theorem 2.2.11 MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITHOUT HOLES *is APX-hard.*

The proof works as well for MINIMUM EDGE GUARD WITHOUT HOLES. Therefore, we have:

Theorem 2.2.12 MINIMUM EDGE GUARD WITHOUT HOLES *is APX-hard.*

2.3 Guarding Polygons with Holes

In this section, we propose a reduction from MINIMUM SET COVER to MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES, analyze it and show that it is gap-preserving. We also show that our result carries over to the problems MINIMUM BOUNDARY RESTRICTED VERTEX/EDGE GUARD WITH HOLES and MINIMUM VERTEX/EDGE/POINT GUARD WITH HOLES.

2.3.1 Construction of the Reduction

As a first step towards our inapproximability result for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES, we show how to construct an instance of MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES for every instance of MINIMUM SET COVER. The construction contains a triangle-shaped pattern, called spike, for each element of the MINIMUM SET COVER instance. All spikes lie on the lower segment of a large rectangle, which is "cut" into an upper and a lower part by a barrier that contains trapezoidal holes, through which a guard in the upper part, which corresponds to a set in the MINIMUM SET COVER instance, can see the spikes in the lower part, which correspond exactly to those elements that are in the set.

We construct a polygon in the $x - y$-plane; Figure 2.14 shows this construction. For each set $s_i, i = 1, \ldots, m$, place on the horizontal line $y = y_0$ the point $((i-1)d', y_0)$. This places a sequence of points from left to right, one point per set s_i for $i = 1, \ldots, m$, with d' a constant distance between

two adjacent points. For ease of description, call the i-th point s_i. For each element $e_i \in E$, place on the horizontal line $y = 0$ two points $(D_i, 0)$ and $(D_i', 0)$, with $D_i' = D_i + d$ for a positive constant d and $D_1 \geq 0$. Arrange the points from left to right for $i = 1, \ldots, n$, with distances $d_i = D_{i+1} - D_i'$ to be defined later. Call the points also D_i and D_i', for $i = 1, \ldots, n$.

For every element e_i, draw a line g through s_j and D_i, where s_j is the first set of which e_i is a member. Also draw a line g' through s_l and D_i', where s_l is the last set of which e_i is a member [1]. For simplicity, let e_i also denote the intersection point of g and g'. Then draw line segments from every s_k that has e_i as a member to D_i and to D_i'.

Two lines connecting points D_i and D_i' with points s_j form a cone-like feature; the area between these two lines will therefore be called a *cone*. Call the triangle $D_i e_i D_i'$ a *spike*. The point e_i of each spike plays a special role and is therefore called the *distinguished point* of the spike.

We have only constructed one part of the polygon thus far: Among all the lines described, only the spikes and the line segments of the horizontal line $y = 0$ that are between adjacent spikes are part of the polygon boundary, all other lines merely help in the construction.

In our construction the guards of an optimum solution will have to be placed at or near the points s_j, therefore we need to make sure that a guard at s_j only sees the distinguished point e_i, if the element e_i is a member of the set s_j. This is achieved by introducing a "barrier"-line at $y = b$, see Fig. 2.14. Only line segments on the horizontal line $y = b$ that are outside the cones are part of the polygon boundary. We draw another barrier-line with distance b' from the first barrier at $y = b + b'$. Define holes of the polygon by connecting endpoints of line segments of the two barrier lines that belong to the same cone-defining line. We call the area between the two lines at $y = b$ and $y = b + b'$ (including all holes) the *barrier*. Thus, the barrier contains a small part of all cones.

As a next step in the construction of the polygon, draw a vertical line segment at $x = -d''$, where d'' is a positive constant, from $y = 0$ to $y = y_0$. This line segment is part of the polygon boundary except for the segment between the two barrier lines.

Choose the coordinates (to be shown later) such that the rightmost spike is farther right than the rightmost set, i.e. $D_n' > s_m$ (for reasons of space, we violated this condition in Fig. 2.14), and draw another vertical line segment from $y = 0$ to $y = y_0$ at $x = D_n' + d''$, again taking a detour at the barrier. The boundary lines of the polygon defined so far are shown as solid lines in Fig. 2.14. It is important to note that the cones, drawn as dashed lines in the figures, are not part of the polygon boundary.

[1] We assume w. l. o. g. that each element is a member of at least two sets.

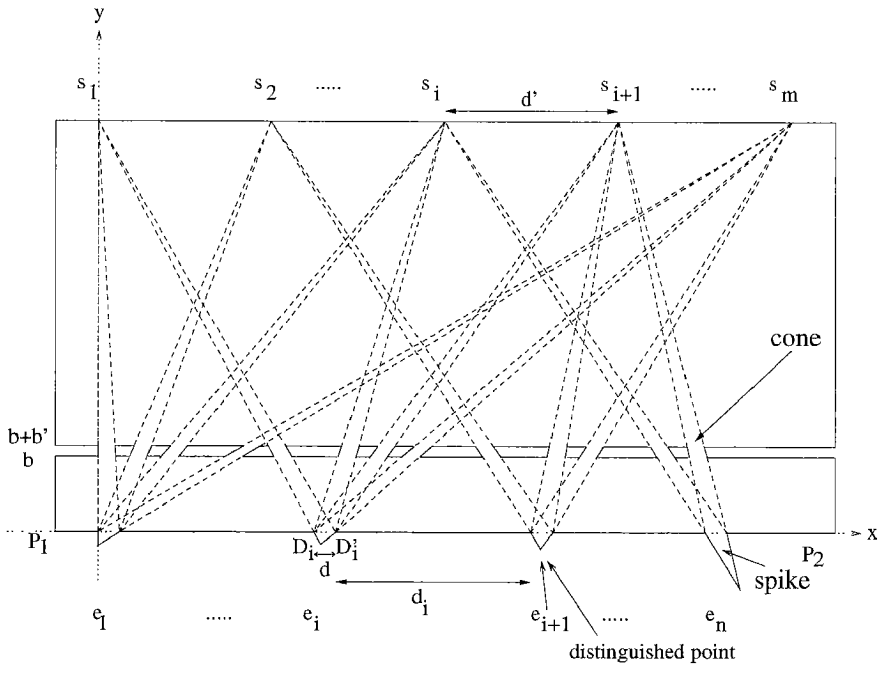


Figure 2.14: Basic construction

The thickness b' of the barrier is defined such that all segments of all holes except for those on the line $y = b + b'$ can be seen from two guards at $P_1 = (-d'', 0)$ and $P_2 = (D'_n + d'', 0)$. To achieve this, the thickness b' is determined by intersecting (for each pair of adjacent holes) a line from P_1 through the lower right corner (point G_1 in Fig. 2.15) of the left hole (of the pair of adjacent holes) with a line from P_2 through the lower left corner (point G_2) of the right hole as shown in Fig. 2.15. Now, the barrier line $y = b + b'$ is defined to go through the lowest of all these intersection points (point y_1 in Fig. 2.15). (They are indeed all at the same height, by arguments with similar triangles.)

We set the parameters of the reduction as follows: Let d' and y_0 be arbitrary positive constants. Let d and b be positive constants as well, where $d = \frac{d'}{4}$ and $b = \frac{5}{12}y_0$. We let $b' = \frac{\frac{35}{12^2}y_0}{-4^{l-1}m^{l-1}+2\sum_{i=0}^{l-1}4^i m^i + 2\frac{d''}{d} - \frac{19}{12}}$ and $D_l = -4^{l-1}m^{l-1}d - d + 2d\sum_{i=0}^{l-1}4^i m^i$ for $l = 1, \ldots, n$.

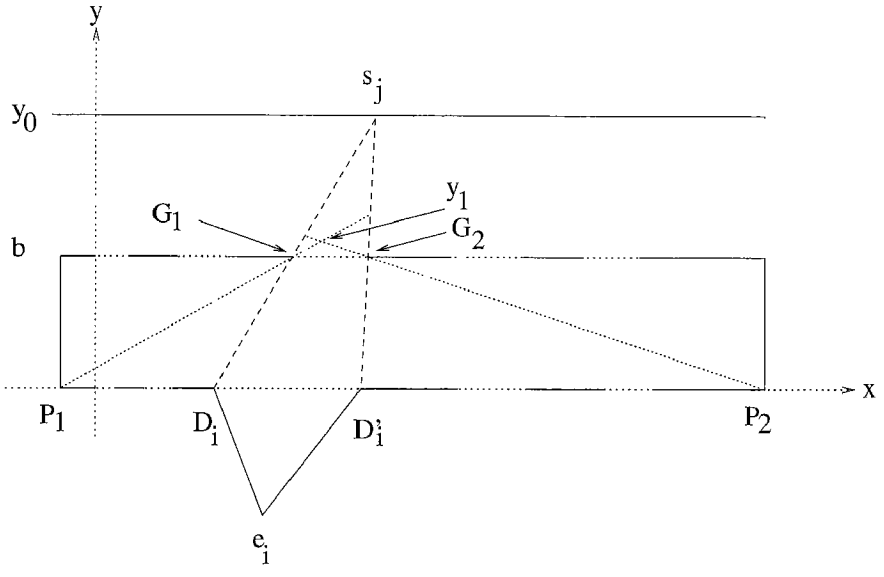


Figure 2.15: Thickness of the barrier

2.3.2 Properties of the Reduction

In order for the reduction to work, it is necessary that at no point a guard sees three or more distinguished points e_i unless there is a set s_j or a pair of sets s_j, s_l that (together) contain(s) all of the corresponding elements.

A guard that is placed at some point with y-value between 0 and $b + b'$, i.e., between the barrier and the spikes, sees at most one such distinguished point, provided the barrier is placed such that no cones of two different elements intersect in the area below the barrier and in the barrier.

In order to ensure that a guard that is placed at some point with y-value between $b + b'$ and y_0 does not see three or more distinguished points unless there is a set s_j or a pair of sets s_j, s_l that (together) contain(s) all of the corresponding elements, we introduce the notion of *extended cones* as shown in Fig. 2.16. The extended cone is the area in the rectangle $D_i, D_i', s_j + a, s_j - a$. Point $s_j - a$ is defined as the intersection point of the line $y = y_0$ with the line from D_i' through the lower right corner of the left of the two holes which contain a part of the cone from set s_j and element e_i. Point $s_j + a$ is defined accordingly. It will be easy to see that points $s_j - a$ and $s_j + a$ are both at a constant distance a from point s_j (see proof of Lemma 2.3.1).

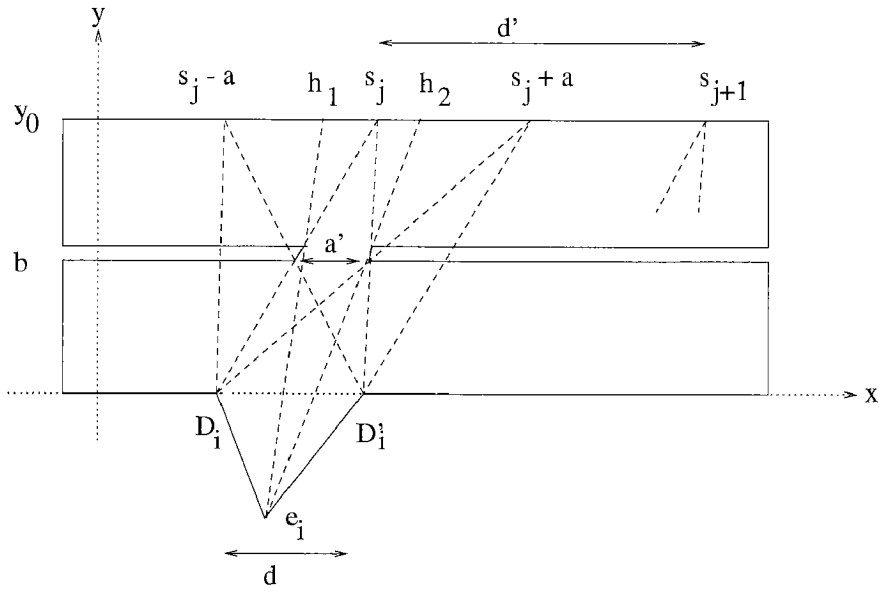


Figure 2.16: Extended cones

For a guard between the two horizontal lines $y = b + b'$ and $y = y_0$, in order to see the distinguished point e_i, it must lie in the area of the triangle defined by the points h_1, h_2 and e_i (or, of course, in the corresponding triangle of any other point $s_{j'}$ with $e_i \in s_{j'}$). In order to keep the analysis simple, we will argue with the extended cones rather than the triangles. If no three extended cones from three different elements and three different sets intersect in this area, then it is ensured that there exists a pair of setpoints such that each distinguished point that a guard in this area sees can also be seen from at least one of the setpoints of the pair. (It is, of course also possible that a single setpoint sees all the distinguished points that a guard in this area sees).

A guard that is placed at some point with y-value less than 0, sees at most one distinguished point, if it is ensured that no two spikes intersect.

Thus, we need to prove the following:

- No three extended cones from different elements and sets intersect.
- The barrier is such that all intersections of cones from the same element e_i are below b (to ensure that the view of the points s_j is blocked appropriately) and such that all intersections of cones from different

elements are above $b + b'$ and such that all of the barrier except for the line segments at $y = b + b'$ can be seen from at least one of two guards at P_1 and P_2.

- No two spikes intersect.

No Three Extended Cones from Different Elements and Sets Intersect

Lemma 2.3.1 *For $e_l \in s_{l'}$, let:*

$$D_l \geq \max\left(\frac{s_{i'} - s_{l'}}{s_{i'} - s_{j'} - 2a}(D_j + d - D_i) + D_i + d\right)$$

where the maximum is taken over all $e_i \in s_{i'}$ and $e_j \in s_{j'}$, for which $i < j < l$ and $l' < j' < i'$ holds. Then the three extended cones from e_l to $s_{l'}$, from e_i to $s_{i'}$ and from e_j to $s_{j'}$, with $i < j < l$ do not have a common intersection point.

Proof: Assume that the positions of the elements, i.e., the values D_v, have been set for all $v < l$ such that no three extended cones (connecting three different sets with three different elements) intersect. We show how to set D_l such that no three extended cones intersect; see Fig. 2.17. Let S be an intersection point with maximum y-value among the two extended cones connecting the elements e_i and e_j with the (different) points $s_{j'}$ and $s_{i'}$.

In order to ensure that our construction is feasible, S must lie in the area between y_0 and the barrier. Let S_y be the y-value of S. Then, $S_y < y_0$. To see this, note that this is equivalent to saying that $s_{j'} + a < s_{i'} - a$ (see Figs. 2.16 and 2.17), which is a weaker condition than $s_{j'} + a < s_{j'+1} - a$. Now, $s_{j'} + a < s_{j'+1} - a$ is equivalent to $2a < d'$. We express a as a function of y_0, b and d using the similarity of triangles. Note that $\frac{a'}{d} = \frac{y_0 - b}{y_0}$ and $\frac{b}{y_0} = \frac{a'}{a}$. Thus, we get $a = \frac{1-b}{b}d$. Using this result in $2a < d'$, we obtain:

$$b > \frac{2}{\frac{d'}{d} + 2} y_0,$$

which is equivalent to $b > \frac{1}{3}y_0$, since $d = \frac{d'}{4}$. This inequality for b is satisfied, since $b = \frac{5}{12}y_0 > \frac{1}{3}y_0$.

For each set $s_{l'}$ of which e_l is a member, draw a line through S, determine where it intersects the line $y = 0$ and let $D^S_{l,l'}$ be the x-value of this intersection point. Let $D^S_l = \max_{l'} D^S_{l,l'}$ be the maximum x-value of

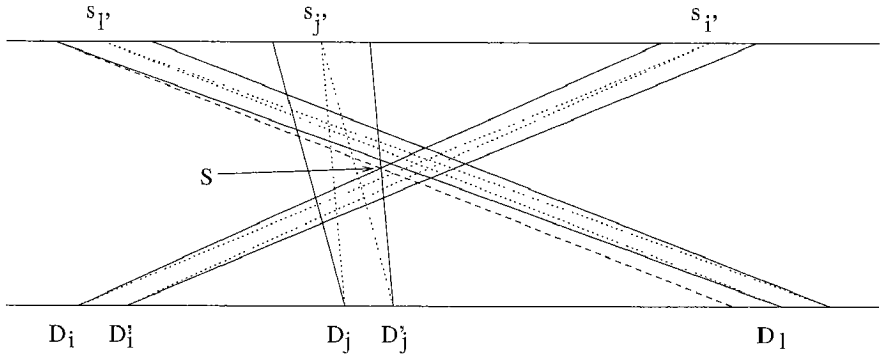


Figure 2.17: Intersection of three extended cones

all intersection points defined this way. For any pair of extended cones in "inverse position" to the left of e_l, with which an extended cone at e_l forms a "triple inversion", compute the corresponding D_l^S and let $D_l^{\max}$ be the maximum D_l^S. Finally, we let $D_l = D_l^{\max} + d$ to ensure that no three extended cones have one common intersection point at some point S. Figure 2.17 shows the situation for an intersection and explains the notation.

The point S is the intersection point of the lines g_1 from $s_{i'} - a$ to D_i and g_2 from $s_{j'} + a$ to D'_j.

These two lines can be expressed with parameter $t \in \mathbb{R}$:

$$g_1 \; : \; (1-t)\begin{pmatrix} s_{i'}-a \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_i \\ 0 \end{pmatrix}$$

$$g_2 \; : \; (1-t)\begin{pmatrix} s_{j'}+a \\ y_0 \end{pmatrix} + t\begin{pmatrix} D'_j \\ 0 \end{pmatrix}$$

The intersection is characterized by parameters t_1 and t_2 for g_1 and g_2:

$$\begin{aligned} (1-t_1)y_0 &= (1-t_2)y_0 \\ (1-t_1)(s_{i'}-a) + t_1 D_i &= (1-t_2)(s_{j'}+a) + t_2 D'_j \end{aligned}$$

The first equation leads to $t_1 = t_2$ and one obtains for t_1:

$$t_1 = \frac{s_{i'} - s_{j'} - 2a}{D'_j - D_i + s_{i'} - s_{j'} - 2a}$$

We express S as:

$$S = \begin{pmatrix} (1-t_1)(s_{i'}-a) + t_1 D_i \\ y_0(1-t_1) \end{pmatrix}$$

Let g_3 be the line from $s_{l'} - a$ to S with $t \in \mathbb{R}$ as parameter:

$$g_3 \quad : \quad (1-t)\begin{pmatrix} s_{l'}-a \\ y_0 \end{pmatrix} + t\begin{pmatrix} (1-t_1)(s_{i'}-a)+t_1 D_i \\ y_0(1-t_1) \end{pmatrix}$$

The intersection of g_3 and $y = 0$ is characterized by parameter t_3 :

$$\begin{aligned} (1-t_3)y_0 + t_3(1-t_1)y_0 &= 0 \\ t_3 &= \frac{1}{t_1} \end{aligned}$$

We let $D^S_{l,l'}$ be the corresponding x-value:

$$\begin{aligned} D^S_{l,l'} &= (1-\frac{1}{t_1})(s_{l'}-a) + \frac{1}{t_1}(1-t_1)(s_{i'}-a) + \frac{1}{t_1}t_1 D_i \\ &= (1-\frac{1}{t_1})(s_{l'}-s_{i'}) + D_i \\ &= \frac{s_{i'}-s_{l'}}{s_{i'}-s_{j'}-2a}(D'_j - D_i) + D_i \\ &= \frac{s_{i'}-s_{l'}}{s_{i'}-s_{j'}-2a}(D_j + d - D_i) + D_i \end{aligned}$$

The lemma follows.

□

Lemma 2.3.1 implies for all $j < l$:

$$\begin{aligned} \max(\frac{s_{i'}-s_{l'}}{s_{i'}-s_{j'}-2a}(D_j + d - D_i) + D_i + d) &\leq \max(\frac{md'}{d'-2a}(D_j + d) + d) \\ &\leq 4m(D_{l-1} + d) + d, \end{aligned}$$

where we have used $a = \frac{1-b}{b}d = \frac{7}{5}d$ and $d' = 4d$ in the last step. Now, let $D_l = 4m(D_{l-1} + d) + d$. It is easy to see that this is consistent with our definition of D_l, since:

$$\begin{aligned} -4^{l-1}m^{l-1}d - \quad d \quad + 2d\sum_{i=0}^{l-1} 4^i m^i = \\ 4m((-4^{l-2}m^{l-2}d - d + 2d\sum_{i=0}^{l-2} 4^i m^i) + d) + d \end{aligned}$$

The Barrier is in Good Position

Lemma 2.3.2 *Any two cones that belong to the same element e_i intersect only at points with y-values at most $y_0 \frac{d}{d+d'}$.*

Proof: Let e_i be a member of s_j and s_l and $s_j < s_l$. The intersection point of the lines g_1 from s_j to D_i' and g_2 from s_l to D_i is the point in the intersection area of the two cones that has the largest y-value. Let this value be y_c.

These two lines can be expressed with parameter $t \in \mathbb{R}$:

$$g_1 \; : \; (1-t)\begin{pmatrix} s_j \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_i' \\ 0 \end{pmatrix}$$

$$g_2 \; : \; (1-t)\begin{pmatrix} s_l \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_i \\ 0 \end{pmatrix}$$

The intersection is characterized by parameters t_1 and t_2 for g_1 and g_2:

$$\begin{aligned} (1-t_1)y_0 &= (1-t_2)y_0 \\ (1-t_1)s_j + tD_i' &= (1-t_2)s_l + tD_i \end{aligned}$$

The first equation leads to $t_1 = t_2$ and one obtains for t_1:

$$t_1 = \frac{s_l - s_j}{D_i' - D_i + s_l - s_j}$$

Since $D_i' - D_i = d$ and since $s_l - s_j \geq d'$, we get:

$$\begin{aligned} y_c &= y_0 \frac{d}{d + s_l - s_j} \\ &\leq y_0 \frac{d}{d + d'} \end{aligned}$$

□

Lemma 2.3.3 *Any two cones that belong to elements e_i, e_j, respectively, with $i < j$, intersect only at points with y-values at least $y_0 \frac{d_i}{d_i + m d'}$.*

Proof: Let e_i be a member of $s_{i'}$ and let e_j be a member of $s_{j'}$, also let $D_i < D_j$ and $s_{j'} < s_{i'}$. Exactly then, the corresponding two cones intersect.

The intersection point of the lines g_1 from $s_{j'}$ to D_j and g_2 from $s_{i'}$ to D_i' is the point in the intersection area of the two cones with minimum y-value. Let this value be y_c.

These two lines can be expressed with parameter $t \in \mathbf{IR}$:

$$g_1 \;:\; (1-t)\begin{pmatrix} s_{j'} \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_j \\ 0 \end{pmatrix}$$

$$g_2 \;:\; (1-t)\begin{pmatrix} s_{i'} \\ y_0 \end{pmatrix} + t\begin{pmatrix} D'_i \\ 0 \end{pmatrix}$$

The intersection is characterized by parameters t_1 and t_2 for g_1 and g_2:

$$\begin{aligned} (1-t_1)y_0 &= (1-t_2)y_0 \\ (1-t_1)s_{j'} + tD_j &= (1-t_2)s_{i'} + t_2 D'_i \end{aligned}$$

The first equation leads to $t_1 = t_2$ and one obtains for t_1:

$$t_1 = \frac{s_{i'} - s_{j'}}{D_j - D'_i + s_{i'} - s_{j'}}$$

Since $D_j - D'_i \geq d_i$ and since $s_{i'} - s_{j'} \leq md'$, we get:

$$\begin{aligned} y_c &= y_0 \frac{D_j - D'_i}{D_j - D'_i + s_{i'} - s_{j'}} \\ &\geq y_0 \frac{d_i}{d_i + md'} \end{aligned}$$

□

Lemma 2.3.4 *Let*

$$b' = \frac{bd(y_0 - b)}{y_0(p_2 - p_1) - d(y_0 - b)},$$

where p_1 and p_2 are the x-values of the points P_1 and P_2. Then all of the barrier including the segments of the cones except for the segments at $y = b + b'$ can be seen from the two guards at P_1 and P_2.

Proof: Let $e_i \in s_j$ and let G_1 and G_2 be the two points where this cone intersects with the barrier line $y = b$ (see Figure 2.15). We need to find an expression for y_1, which is the y-value of the intersection point of the two lines from P_1 to G_1 and from P_2 to G_2.

We find an expression for the point G_1 by calculating the intersection of the lines from s_j to D_i and $y = b$ and obtain:

$$G_1 = \begin{pmatrix} \frac{b}{y_0}(s_j - D_i) + D_i \\ b \end{pmatrix}$$

We find an expression for the point G_2 by calculating the intersection of the lines from s_j to $D_i + d$ and $y = b$ and obtain:

$$G_2 = \begin{pmatrix} \frac{b}{y_0}(s_j - D_i - d) + D_i + d \\ b \end{pmatrix}$$

Now, we find the intersection point of the lines from P_1 to G_1 and from P_2 to G_2:

$$(1-t_1)\begin{pmatrix} p_1 \\ 0 \end{pmatrix} + t_1 \begin{pmatrix} \frac{b}{y_0}(s_j - D_i) + D_i \\ b \end{pmatrix} = (1-t_2)\begin{pmatrix} p_2 \\ 0 \end{pmatrix} + t_2 \begin{pmatrix} \frac{b}{y_0}(s_j - D_i - d) + D_i + d \\ b \end{pmatrix}$$

Again, $t_1 = t_2$ and we obtain:

$$t_1 = \frac{p_1 - p_2}{d - \frac{bd}{y_0} + p_1 - p_2}$$

Therefore:

$$y_1 = bt_1$$

y_1 does not depend on D_i, therefore we let $b' = y_1 - b = b(t_1 - 1)$:

$$b' = \frac{bd(y_0 - b)}{y_0(p_2 - p_1) - d(y_0 - b)}$$

□

If we substitute $b = \frac{5}{12}y_0$ and $p_2 - p_1 = -4^{n-1}m^{n-1}d - d + 2d\sum_{i=0}^{n-1} 4^i m^i + d'' - (-d'') = -4^{n-1}m^{n-1}d - d + 2d\sum_{i=0}^{n-1} 4^i m^i + 2d''$ in the equation for b', we obtain:

$$b' = \frac{\frac{35}{12^2}y_0}{-4^{n-1}m^{n-1}2\sum_{i=0}^{n-1} 4^i m^i + 2\frac{d''}{d} - \frac{19}{12}}$$

A simple calculation shows that $b' < \frac{y_0}{12}$, if $m \geq 2$ and $n \geq 2$, which must be the case since there were no intersections otherwise.

Because of $d = \frac{d'}{4}$ and because of Lemma 2.3.2, any two cones from the same element intersect only at points with y-value at most $\frac{1}{5}y_0$ which is less than b. Because of $d_i \geq md'$ for all d_i and because of Lemma 2.3.3, any two cones from different elements intersect only at points with y-value at least $\frac{1}{2}y_0$, which is at most $b + b'$.

Spikes of Two Elements do not Intersect

Lemma 2.3.5 *The spikes of any two elements do not intersect.*

Proof: Let s_i be the first and let s_j be the last set that e_l is a member of. Obviously, $s_i < s_j$. The intersection point of the lines g_1 from s_i through D_l and g_2 from s_j through D'_l is the point I_l. Let the x-value of this point be x_l. Note that $x_l > D_l$.

These two lines can be expressed with parameter $t \in \mathbb{R}$:

$$\begin{aligned} g_1 &: (1-t)\begin{pmatrix} s_i \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_l \\ 0 \end{pmatrix} \\ g_2 &: (1-t)\begin{pmatrix} s_j \\ y_0 \end{pmatrix} + t\begin{pmatrix} D'_l \\ 0 \end{pmatrix} \end{aligned}$$

The intersection is characterized by parameters t_1 and t_2 for g_1 and g_2:

$$\begin{aligned} (1-t_1)y_0 &= (1-t_2)y_0 \\ (1-t_1)s_i + t_1 D_l &= (1-t_2)s_j + t_2 D'_l \end{aligned}$$

The first equation leads to $t_1 = t_2$ and with $D'_l = D_l + d$ one obtains for t_1:

$$t_1 = \frac{s_i - s_j}{d + s_i - s_j}$$

Thus, we obtain:

$$\begin{aligned} x_c &= s_i \frac{d}{d + s_i - s_j} + D_l \frac{s_i - s_j}{d + s_i - s_j} \\ &\leq D_l \frac{s_i - s_j}{d + s_i - s_j} \\ &\leq D_l \frac{-d'}{d - d'} \\ &= D_l \frac{-4d}{d - 4d} \\ &= \frac{4}{3} D_l \end{aligned}$$

where the second but last step is due to $d = \frac{d'}{4}$. Since $D_{l+1} = 4m(D_l + d) + d$ and since we can assume that $m \geq 1$, the lemma follows.

□

2.3.3 Transformation of the Solution

Given a solution of the MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES-instance, i.e. the coordinates of r guards $g_1, \ldots, g_r$, proceed as follows to obtain a solution for the MINIMUM SET COVER-instance:

For each guard g_i, determine the set h_i of elements e_j of which the guard g_i sees the corresponding distinguished point e_j.

Since no three extended cones from three different elements and three different sets intersect in the area above $y = b + b'$ by our construction, there exists a pair of sets (s_k, s_l) for each guard g_i such that $h_i \subseteq s_k \cup s_l$. Determine such a pair of sets for each guard g_i and add the sets to the solution of the MINIMUM SET COVER-instance.

2.3.4 The Reduction is Polynomial

Note that d, d', y_0, h, b are all constants in our reduction. The values for b' and for all D_i are computable in polynomial time and can be expressed with $O(n \log m)$ bits.

Therefore, the construction of the polygon can be done in time polynomial in the size of the input MINIMUM SET COVER-instance, since it only produces a polynomial number of points that each can be computed in polynomial time and each take at most $O(n \log m)$ bits to be expressed.

It is obvious that the transformation of the solution runs in polynomial time, since it only involves determining whether two points see each other and finding pairs of sets for a polynomial number of guards. (Note that if the number of guards exceeds n, the solution is trivial.)

2.3.5 An Inapproximability Result for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES

In order to prove a strong inapproximability result, we need the following:

Definition 2.3.6 *The* RESTRICTED MINIMUM SET COVER *problem consist of all* MINIMUM SET COVER *instances that have the property that the number of sets m is less or equal to the number of elements n, i.e., $m \leq n$.*

Lemma 2.3.7 RESTRICTED SET COVER *cannot be approximated by any polynomial time algorithm with an approximation ratio of $(1 - \epsilon) \ln n$ for any $\epsilon > 0$, unless $NP \subseteq TIME(n^{O(\log \log n)})$.*

Proof: DOMINATING SET cannot be approximated with an approximation ratio of $(1-\epsilon) \ln n$ for any $\epsilon > 0$, unless $NP \subseteq TIME(n^{O(\log \log n)})$, where n is the number of vertices in the graph [10]. Consider the following reduction

from DOMINATING SET to RESTRICTED SET COVER: Given a graph $G = (V, E)$ with $n := |V|$, which is an instance of DOMINATING SET, we construct a RESTRICTED SET COVER instance by letting the vertices of G be elements and by forming a set for each vertex that contains the vertex itself as well as its neighbors. The RESTRICTED SET COVER instance thus obtained contains n elements and n sets. This is clearly a gap-preserving reduction, since each feasible solution of the RESTRICTED SET COVER instance directly corresponds to a feasible solution (of the same size) of the DOMINATING SET instance.

□

We will now consider the reduction to be from RESTRICTED MINIMUM SET COVER to MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES (rather than from MINIMUM SET COVER to MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES).

Lemma 2.3.8 *Consider the promise problem of* RESTRICTED MINIMUM SET COVER *(for any $\epsilon > 0$), where it is promised that the optimum solution OPT is either less or equal to c or greater than $c(1-\epsilon)\ln n$ with c, n and OPT depending on the instance I. This problem is NP-hard unless $NP \subseteq TIME(n^{O(\log\log n)})$ (see the notion of quasi-NP-hardness in [3]). Then, we have for the optimum value OPT' of the corresponding* MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES*-instance I', that OPT' is either less or equal to $c+2$ or greater than $\frac{c+2}{12}(1-\epsilon)\ln|I'|$. More formally:*

$$OPT \leq c \implies OPT' \leq c+2 \tag{2.1}$$

$$OPT > c(1-\epsilon)\ln n \implies OPT' > \frac{c+2}{12}(1-\epsilon)\ln|I'| \tag{2.2}$$

Proof: The implication in (2.1) is trivial, since, given a solution of the RESTRICTED MINIMUM SET COVER-instance I of size c, we position a guard at each point s_j in the corresponding MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES-instance I', if the set s_j is in the solution of I, and we position two additional guards at points P_1 and P_2 in I', which see the barrier from below.

We prove the contraposition of (2.2), i.e.:

$$OPT' \leq \frac{c+2}{12}(1-\epsilon)\ln|I'| \implies OPT \leq c(1-\epsilon)\ln n$$

Observe that, if we are given a solution of I' with k guards, we can obtain a solution of I with at most $2k$ sets by performing the procedure described

in Section 2.3.3. Therefore:

$$\begin{aligned} OPT &\leq 2\frac{c+2}{12}(1-\epsilon)\ln|I'| && (2.3)\\ &\leq 2\frac{c+2}{12}(1-\epsilon)\ln n^3 && (2.4)\\ &\leq 2\cdot 3\frac{2c}{12}(1-\epsilon)\ln n && (2.5)\\ &\leq c(1-\epsilon)\ln n && (2.6) \end{aligned}$$

where we used $|I'| \leq n^3$ to get (2.4), which is true because the polygon of I' consists of n spikes and less than $nm \leq n^2$ holes (see definition of RESTRICTED MINIMUM SET COVER). Therefore, the polygon consists of less than $k(n^2+n)$ points, where k is a small constant. Therefore, $|I'| \leq n^3$ for n large enough. We used $2c \geq c+2$ to get to (2.5).

□

Lemma 2.3.8 completes the proof of Theorem 2.3.9.

Theorem 2.3.9 MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES *cannot be approximated by a polynomial time algorithm with an approximation ratio of* $\frac{1-\epsilon}{12}\ln n$ *for any* $\epsilon > 0$, *where* n *is the number of the polygon vertices, unless* $NP \subseteq TIME(n^{O(\log\log n)})$.

2.3.6 Inapproximability Results for MINIMUM BOUNDARY RESTRICTED VERTEX/EDGE GUARD WITH HOLES

A slight modification of the polygon as indicated in Fig. 2.18, where $b'' = y_0 + b'$, allows us to prove the corresponding theorems for MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITH HOLES and MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITH HOLES.

Theorem 2.3.10 MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITH HOLES *cannot be approximated by a polynomial time algorithm with an approximation ratio of* $\frac{1-\epsilon}{12}\ln n$ *for any* $\epsilon > 0$, *where* n *is the number of polygon vertices, unless* $NP \subseteq TIME(n^{O(\log\log n)})$.

Proof: The proof is almost identical to the proof for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES, except that instead of two additional guards at P_1 and P_2 we have a third additional guard at P_3 (see Fig. 2.18). This additional guard means that we need to replace $c+2$ by $c+3$ in the proof of Lemma 2.3.8. In addition, we get a slightly stronger condition, namely $2c \geq c+3$, to obtain the inequality at (2.5).

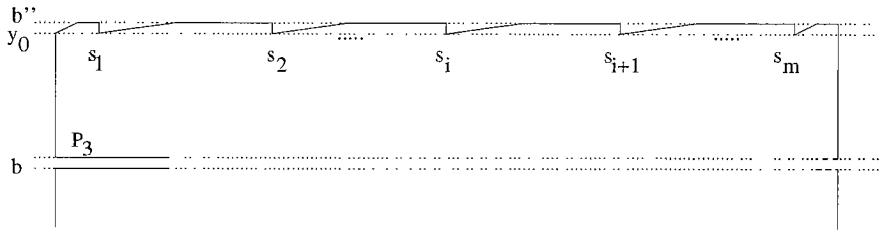


Figure 2.18: Polygon for MINIMUM BOUNDARY RESTRICTED VERTEX/EDGE GUARD WITH HOLES

□

Theorem 2.3.11 MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITH HOLES *cannot be approximated by a polynomial time algorithm with an approximation ratio of* $\frac{1-\epsilon}{12}\ln n$ *for any* $\epsilon > 0$, *where* n *is the number of polygon vertices, unless* $NP \subseteq TIME(n^{O(\log\log n)})$.

Proof: The proof is almost identical to the proof for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES with the additional information from the proof of Theorem 2.3.10. Note that in the case of MINIMUM BOUNDARY RESTRICTED EDGE GUARD WITH HOLES all guards are edges. The proofs carry over effortlessly.

□

2.3.7 Inapproximability Results for MINIMUM VERTEX /EDGE/POINT GUARD WITH HOLES

Theorem 2.3.12 MINIMUM VERTEX/EDGE/POINT GUARD WITH HOLES *cannot be approximated by a polynomial time algorithm with an approximation ratio of* $\frac{1-\epsilon}{12}\ln n$ *for any* $\epsilon > 0$, *where* n *is the number of polygon vertices, unless* $NP \subseteq TIME(n^{O(\log\log n)})$.

Proof: All proofs, which lead to Theorems 2.3.9, 2.3.10 and 2.3.11 carry over. Note that for a lemma corresponding to Lemma 2.3.8, which is the most crucial part in the proof, we can still, virtually without change, prove Eq. 2.1 and Eq. 2.2.

□

2.3.8 Inapproximability Results under the Assumption that $NP \neq P$

All our logarithmic inapproximablity results can actually also be proved under the weaker assumption that $NP \neq P$, however, with a slightly smaller factor than $1 - \epsilon$, since MINIMUM DOMINATING SET and therefore also RESTRICTED MINIMUM SET COVER cannot be approximated with an approximation ratio of $c \ln n$ for some $c > 0$ [4, 10, 43]. As the constant c is still being improved in current research (it will probably ultimately reach $1 - \epsilon$), we have chosen to present our results under the stronger assumption that $NP \neq TIME(n^{O(\log \log n)})$, which allows us to use the stronger inapproximability results for MINIMUM SET COVER and MINIMUM RESTRICTED SET COVER [10, 24].

2.4 Guarding 2.5 Dimensional Terrains

We prove inapproximability results for several terrain guarding problems by proposing reductions from RESTRICTED SET COVER, all of which are based on the reduction proposed for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES.

Theorem 2.4.1 MINIMUM POINT GUARD ON TERRAIN *cannot be approximated by a polynomial time algorithm with an approximation ratio of* $\frac{1-\epsilon}{12} \ln n$ *for any* $\epsilon > 0$, *where* n *is the number of terrain vertices, unless* $NP \subseteq TIME(n^{O(\log \log n)})$.

Proof: We reduce RESTRICTED SET COVER to MINIMUM POINT GUARD ON TERRAIN. In a first step, we construct the same polygon with holes as constructed in the corresponding reduction for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES. We then triangulate this polygon arbitrarily, and construct a terrain, by letting the interior of the polygon have height 0 and the exterior (including the holes in the barrier) have height h', for a positive constant h'. To make the terrain finite, we cut off the exterior of the polygon with a generous bounding box that is triangulated as well. Point Guard on The terrain we obtain has vertical walls, which is for reasons of simplicity only. The terrain can easily be modified to have steep, but not vertical walls.

The proof then carries on just as for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITH HOLES. The setpoints s_j are assumed to be at height h', as are points P_1 and P_2; the distinguished points e_i, however are at height 0.

□

Theorem 2.4.2 MINIMUM VERTEX GUARD ON TERRAIN *cannot be approximated by a polynomial time algorithm with an approximation ratio of* $\frac{1-\epsilon}{12} \ln n$ *for any* $\epsilon > 0$, *where* n *is the number of terrain vertices, unless* $NP \subseteq TIME(n^{O(\log \log n)})$.

Proof: Adopt the proof of Theorem 2.4.1 for MINIMUM POINT GUARD ON TERRAIN with the modifications of the constructed polygon as indicated in Theorem 2.3.10 for MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITH HOLES.

□

Theorem 2.4.3 MINIMUM FIXED HEIGHT GUARD ON TERRAIN *cannot be approximated by a polynomial time algorithm with an approximation ratio of* $\frac{1-\epsilon}{12} \ln n$ *for any* $\epsilon > 0$, *where* n *is the number of terrain vertices, unless* $NP \subseteq TIME(n^{O(\log \log n)})$.

Proof: Proceed as in the proof for Theorem 2.4.1, and let h, the fixed height, where the guards can be placed, be equal to h', the height of the exterior of the polygon.

□

Theorem 2.4.4 *The problems* MINIMUM VERTEX/POINT/FIXED HEIGHT GUARD ON TERRAIN WITH TRIANGLE RESTRICTION *cannot be approximated by a polynomial time algorithm with an approximation ratio of* $\frac{1-\epsilon}{12} \ln n$ *for any* $\epsilon > 0$, *where* n *is the number of terrain vertices, unless* $NP \subseteq TIME(n^{O(\log \log n)})$.

Proof: Proceed as in the proofs for Theorems 2.4.2, 2.4.2 and 2.4.3. However, triangulate the polygon in such a way that each spike is triangulated into a single triangle. Note that the spikes take over the role of the distinguished points. A solution for RESTRICTED MINIMUM SET COVER with k sets can still be easily transformed into a solution of the terrain guarding instance with k guards. Furthermore, a solution of the terrain guarding instance with k guards can still be transformed into a solution of the RESTRICTED MINIMUM SET COVER instance with at most $2k$ sets, because three different cones from three different elements and three different sets do not intersect in the area above the barrier, because this is a weaker condition than the corresponding condition with extended cones.

□

These logarithmic inapproximablity results can also be proved under the weaker assumption that $NP \neq P$ with a slightly smaller factor than $1 - \epsilon$, as mentioned in Sect. 2.3.8

2.5 Approximation Algorithms

2.5.1 Introduction

Despite the success of approximation algorithms in other fields, there has not been much research on approximation algorithms for visibility problems. There are only very few results.

2.5.2 Approximating Polygon Guarding Problems

It is known that the four problems MINIMUM VERTEX/EDGE GUARD WITH(OUT) HOLES are approximable with a ratio of $O(\log n)$, where n is the number of polygon vertices [26]. For the sake of completeness we briefly outline these algorithms.

The approximation algorithms partition the interior of the input polygon into "basic" convex components that are either completely visible or invisible from any vertex- or edge-guard. These basic convex components are obtained by drawing lines through all pairs of vertices of the polygon. Each of the $O(n^2)$ lines intersects at most $O(n^2)$ lines, which gives a total of $O(n^4)$ intersection points. Each intersection point is the vertex of a convex component that is minimum (in the convex component) with respect to the y-axis (where the y-axis is arbitrary but fixed). Therefore, we have $O(n^4)$ basic convex components.

The problem is then transformed into an instance of MINIMUM SET COVER with an element for each convex component and a set for each polygon vertex (or polygon edge). The sets contain as elements exactly the convex components that are visible from the vertex.

MINIMUM SET COVER can be approximated with an approximation ratio that is logarithmic in the number of elements of the MINIMUM SET COVER instance. The greedy algorithm that achieves this ratio consists of recursively adding to the solution the set that contains a maximum number of elements not yet contained in the solution obtained thus far [32]. Since we have a polynomial number of elements, the approximation ratio that the greedy algorithm achieves remains logarithmic.

These algorithms can easily be modified to work for MINIMUM BOUNDARY RESTRICTED VERTEX/EDGE GUARD WITH(OUT) HOLES by simply considering only those basic components that contain a part of a line segment of the boundary of the input polygon.

The situation is far worse for point guard problems. No sophisticated approximation algorithms are known for MINIMUM (BOUNDARY RESTRICTED) POINT GUARD WITH(OUT) HOLES, except for an even more restricted version of MINIMUM POINT GUARD WITHOUT HOLES [1]. In fact, it is not even known, if the corresponding decision problems are in NP.

A trivial approximation algorithm for MINIMUM (BOUNDARY RESTRICTED) POINT GUARD WITH(OUT) HOLES simply returns all n vertices of the polygon as a (feasible) solution. This algorithm achieves an approximation ratio of n, because there is at least one guard needed in any feasible solution. Note that this ratio might be improved slightly for MINIMUM POINT GUARD WITHOUT HOLES by applying an algorithm that places $\lfloor \frac{n}{3} \rfloor$ guards that together see all of the interior of the polygon (see [48] for details); this could be done similarly for MINIMUM POINT GUARD WITH HOLES with another algorithm (see [48] for details), but the approximation ratio remains $O(n)$.

A more sensible algorithm for point guard problems returns the (suboptimum) solution found for the corresponding vertex guard problem on the same input polygon. However, the approximation ratio remains $O(n)$.

Another approach for solving point guard problems works as follows: Lay a grid of polynomial density over the polygon and then compute the area of visibility (i.e. the part of the polygon that is visible from a guard at this point) for each grid point. This can be done in polynomial time, since it corresponds to constructing convex components. Here, however, we obtain the convex components by drawing lines from each grid point through each polygon vertex. We then obtain a MINIMUM SET COVER instance with sets for each grid point and elements for each convex component. Despite of all this extra effort, the approximation ratio remains $O(n)$, to the best of our knowledge.

2.5.3 Approximating Terrain Guarding Problems

We generalize the notion of basic convex components introduced in the previous section to terrains. This will help us obtain approximation algorithms for several terrain guarding problems. We obtain convex components by constructing planes through vertices and line segments of the terrain. More precisely, let v_i, v_j and v_k be vertices in a terrain T, where the vertices v_i and v_j are neighbors, i.e., the two vertices are connected by a line segment in the triangulation of the terrain. For each line segment v_i, v_j and each

vertex v_k on the terrain, we construct a plane that contains the line segment from v_i to v_j and the vertex v_k. Since there are only $O(n)$ line segments in the triangulation of T, this gives a total of $O(n^2)$ planes. These planes partition space into three dimensional *cells*, which in turn contain two dimensional *faces* that are defined by (one dimensional) *points*, which are the intersection points of three planes. We call this partition the *arrangement* of T. The arrangement consists of $O(n^6)$ intersection points, faces, and cells that can be computed in time $O(n^6)$.[2]

Lemma 2.5.1 *Let C be a cell in the arrangement of a terrain T. Then, every point in the interior of cell C (i.e. any point not on the faces or intersection points that also belong to C) sees exactly the same vertices of terrain T as any other point in the interior of C.*

Proof: Let a, b be two points in the interior of C and let v_i be a vertex on the terrain T. Assume by contradiction that b sees v_i, while a does not see v_i. Then, let a' be the intersection point of the terrain T and the line segment from point a to vertex v_i that is closest to point a. Since a does not see v_i, there always must be such a point a' that blocks the view. Let b' be the intersection point of the line segment from b to v_i and the plane defined by the triangle of the terrain, on which point a' lies. Since b sees v_i, b' cannot lie on the terrain, and in particular it cannot lie on the same triangle as a'. Consider the plane through vertex v_i and the line segment of the terrain that intersects with the line segment from a' to b'. This plane is a part of the arrangement, but it cuts cell C apart, as it separates points a and b from each other. Therefore, cell C is not a cell of the arrangement. □

Lemma 2.5.2 *Let C be a cell in the arrangement of a terrain T. Then, every point in the interior of cell C sees exactly the same line segments of the triangulation of terrain T completely as any other point in the interior of C.*

Proof: Let a, b be two points in the interior of C and let c be the line segment in the triangulation of T with vertices v_i and v_j as endpoints. Assume by contradiction that b sees c, while a does not see c completely. Since a and b are both in cell C we can assume by Lemma 2.5.1 that they both see the two vertices v_i and v_j. Consider the points on the line segment from point a to point b as we move from a towards b. Since a does not see

[2]These numbers are obtained easily using standard analysis of properties of arrangements that can be found in any textbook on computational geometry.

segment c completely, but b does, there must be a point p from where c is completely visible for the first time (as we move along the line segment from a to b). Remember that by Lemma 2.5.1 vertices v_i and v_j are always visible as we move. Therefore, by definition of point p, there must be a terrain vertex v_k (other that v_i and v_j) on the plane defined by v_i, v_j and p. Otherwise, p would not be the first point to completely see segment c, or not all points would see vertices v_i and v_j. The plane defined by v_i, v_j and p is equal to the plane defined by v_i, v_j and v_k. This plane, however, is part of the arrangment, since v_i and v_j are neighbors in the triangulation of terrain T; it separates points a and b, and therefore, C is not a cell of the arrangement. □

Lemma 2.5.3 *Let F be a face in the arrangement of a terrain T. Then, every point in the interior of face F (i.e. points not on the boundary of F) sees exactly the same vertices and the same complete line segments of the triangulation of terrain T as any other point in the interior of F.*

Proof: The proofs for the vertices and the line segments are the same as the proofs for Lemmas 2.5.1 and 2.5.2 respectively. □

Before we propose a first approximation algorithm, let us mention a few elementary facts:

- For each cell or face of the arrangement of a terrain T, the intersection points that are on the boundary of the cell or face see all the vertices and line segments in the triangulation of T that any other point in the interior of the cell or face sees. The intersection points may, however, see a few additional vertices and line segments.

- For any point p in space and any vertex v_i on terrain T, we determine if p sees v_i as follows: Compute all line segments in the triangulation of T that intersect the line from p to v_i in the orthogonal, two dimensional projection onto the $x-y$ plane. Then, check for each such segment whether the line from p to v_i is above or below the segment (with respect to the z-axis). Point p sees vertex v_i, exactly if each segment is below the line from p to v_i. This can be computed in time $O(n)$.

- For any point p in space and any line segment in the triangulation of T with endpoints v_i and v_j, we determine if p completely sees the line segment as follows: We first determine whether p sees the two vertices v_i and v_j. If this is affirmative, we check for each vertex on the terrain, which lies in the triangle v_i, v_j, p in its orthogonal projection onto the

$x-y$ plane, whether the vertex lies above or below the triangle v_i, v_j, p (with respect to the z-axis). Point p completely sees the segment from v_i to v_j, exactly if each such vertex lies below the triangle. This can be computed in time $O(n)$.

- We can determine if a point in space completely sees a triangle on the terrain by determining if it completely sees all three sides of the triangle. This takes $O(n)$ time.

We are now ready to prove approximation results.

Theorem 2.5.4 MINIMUM FIXED HEIGHT GUARD ON TERRAIN WITH TRIANGLE RESTRICTION *can be approximated by a polynomial time algorithm with a ratio of* $O(\log n)$*, where* n *is the number of terrain vertices.*

Proof: Consider the intersection of the arrangment of terrain T with the plane at $z = h$. This intersection is itself a two dimensional arrangement. Since we know from Lemmas 2.5.2 and 2.5.3 that all interior points in a cell or in a face see the same line segments in the triangulation of T and the boundary points may see a few additional line segments, it suffices to determine at each of the $O(n^6)$ intersection points (i.e. points in the two dimensional arrangement at height h, where two lines cross), which triangles it sees completely. This can be done in $O(n^6 \cdot n \cdot n) = O(n^8)$ time.

We can now interpret this information as a MINIMUM SET COVER instance, where each triangle in the terrain is an element and each intersection point defines a set, namely the set of triangles that it sees completely. This instance consists of $O(n)$ elements and $O(n^6)$ different sets. We now solve the MINIMUM SET COVER instance approximately, by applying the greedy algorithm for MINIMUM SET COVER. The greedy algorithm runs in time $O(n^8)$ as does the whole approximation algorithm. It achieves an approximation ratio of $O(\log n)$. □

Theorem 2.5.5 MINIMUM VERTEX GUARD ON TERRAIN WITH TRIANGLE RESTRICTION *can be approximated by a polynomial time algorithm with a ratio of* $O(\log n)$*, where* n *is the number of terrain vertices.*

Proof: We again build a MINIMUM SET COVER instance, where each triangle in the terrain is an element and each vertex defines a set, i.e., the set of triangles that it sees completely. In order to do this, we have to compute n sets, each of which takes time $O(n^2)$. This gives a total construction time of $O(n^3)$.

Solving the MINIMUM SET COVER instance by applying the greedy algorithm takes time $O(n^3)$ and achieves an approximation ratio of $O(\log n)$. □

Theorem 2.5.6 MINIMUM VERTEX GUARD ON TERRAIN *can be approximated by a polynomial time algorithm with a ratio of* $O(\log n)$*, where* n *is the number of terrain vertices.*

Proof: Consider the intersection of the arrangement of T with the terrain T itself. This intersection partitions all triangles of the terrain into two dimensional cells. Within such a cell, all points see the same set of vertices according to Lemma 2.5.3. The "inverse" holds as well: Any vertex in the terrain either sees such cell completely or not at all (except for points on the boundary of the cell). There are $O(n^6)$ such cells that can be computed in time $O(n^6)$. Note that we can determine in time $O(n)$, whether a vertex sees a cell, by testing whether it sees an interior point of the cell.[3]

We construct a MINIMUM SET COVER instance, where each cell in the terrain, which results from intersecting the arrangement of T with T itself, is an element and each vertex defines a set, i.e., the set of all cells that it sees completely. We have to compute n sets, each of which takes time $O(n^7)$. This gives a total construction time of $O(n^8)$.

We again solve the MINIMUM SET COVER instance by applying the greedy algorithm. This takes time $O(n^8)$ and achieves an approximation ratio of $O(\log n)$. □

Theorem 2.5.7 MINIMUM POINT GUARD ON TERRAIN WITH TRIANGLE RESTRICTION *can be approximated by a polynomial time algorithm with a ratio of* $O(\log n)$*, where* n *is the number of terrain vertices.*

Proof: Consider once again the intersection of the arrangement of T with the terrain T itself that partitions all triangles of the terrain into two dimensional cells. Within such a cell, all points see the same set of line segments of the triangulation of T according to Lemma 2.5.3; they also all see the same set of triangles on the terrain. Therefore, it suffices to place point guards at intersection points (i.e. points in on the terrain, where two lines of the arrangement cross).

We construct a MINIMUM SET COVER instance, where each triangle in the triangulation of the terrain is an element and each intersection point

[3] We can find an interior point by drawing two arbitrary diagonals and taking the intersection or – in the case of a triangle – by computing its center of gravity.

on the terrain defines a set, namely the set of all triangles that it sees completely. We have to compute $O(n^6)$ sets, each of which takes time $O(n^2)$. This gives a total construction time of $O(n^8)$.

We again solve the MINIMUM SET COVER instance by applying the greedy algorithm. This takes time $O(n^8)$ and achieves an approximation ratio of $O(\log n)$. □

In all the approximation algorithms proposed for terrain guarding problems in the proofs of the previous lemmas, we have focused on the polynomiality of these algorithms. The running times are truly upper bounds, since there exist algorithms, particularly for computing the horizon from a point in the terrain [13], that might be adopted to work for our problems as well with somewhat improved running times.

No sophisticated approximation algorithms are known for MINIMUM POINT GUARD ON TERRAIN and MINIMUM FIXED HEIGHT GUARD ON TERRAIN. These problems seem to defy all attempts to somehow discretize the space of all possible guard positions. Again, we can come up with several approaches to find good solutions for these two problems:

A trivial approximation algorithm for MINIMUM POINT GUARD ON TERRAIN simply puts a guard at each vertex of the terrain, thus achieving an approximation ratio of n, since at least one guard is always needed.

A trivial approximation algorithm for MINIMUM FIXED HEIGHT GUARD ON TERRAIN with approximation ratio n places a guard above each vertex of the terrain at height h. Again, the approximation ratios could be improved by a constant factor, since there exists an algorithm [6] that always places $\lfloor \frac{3n}{5} \rfloor$ (vertex-)guards on a terrain that together see all of the terrain. For MINIMUM FIXED HEIGHT GUARD ON TERRAIN, we could also reduce the approximation ratio to $\frac{n}{2}$ by determining whether height h is large enough such that the whole terrain can be seen from one single guard at some point at height h. The position of such a guard can be computed in linear time using linear programming (mentioned in [49] as the problem of computing the *lowest watchtower*). An approximation algorithm for MINIMUM FIXED HEIGHT GUARD ON TERRAIN could return the position of such a guard and if no such guard exists, it would proceed as in the trivial algorithm. However, the approximation ratios remain $O(n)$ for all problems.

A better approach, even if the approximation ratio remains $O(n)$, would be to solve the corresponding problems with triangle restriction, for which we have given approximation algorithms above.

Finally, the last approach from Sect. 2.5.2, which consists of laying a grid over the polygon, can be used here as well: We just lay a regular polynomial density grid onto the plane $z = h$ or the terrain itself and construct a MIN-

IMUM SET COVER instance for the MINIMUM FIXED HEIGHT GUARD ON TERRAIN problem or the MINIMUM POINT GUARD ON TERRAIN problem, respectively.

Chapter 3

Hiding

3.1 Hiding in Polygons without Holes

3.1.1 Introduction

We propose a gap-preserving reduction from MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY to MAXIMUM HIDDEN SET ON POLYGON WITHOUT HOLES, which allows us to prove the APX-hardness of MAXIMUM HIDDEN SET ON POLYGON WITHOUT HOLES. The same reduction will also work for MAXIMUM HIDDEN VERTEX SET ON POLYGON WITHOUT HOLES with minor modifications.

3.1.2 Construction of the Reduction

Suppose we are given an instance I of MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY, which consists of n variables $x_0, \ldots, x_{n-1}$ and m clauses $c_0, \ldots, c_{m-1}$. We construct a polygon without holes, i.e. an instance I' of MAXIMUM HIDDEN SET ON POLYGON WITHOUT HOLES, which consists of clause patterns and variable patterns, as shown schematically in Fig. 3.1. The construction looks somewhat similar to the construction from Sect. 2.2, where we reduced MINIMUM 5-OCCURRENCE-3-SATISFIABILITY to MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES; in fact, the main building blocks are alike: we have a variable pattern for each variable that consists of a TRUE- and a FALSE-leg. We have a clause pattern for each clause that is a simple zig-zag line forming 3 dents. The variable patterns contain a small dent, which we will call "cone", for each occurrence of the variable in the input satisfiability formula. These cones "connect" the variable patterns with the clause patterns.

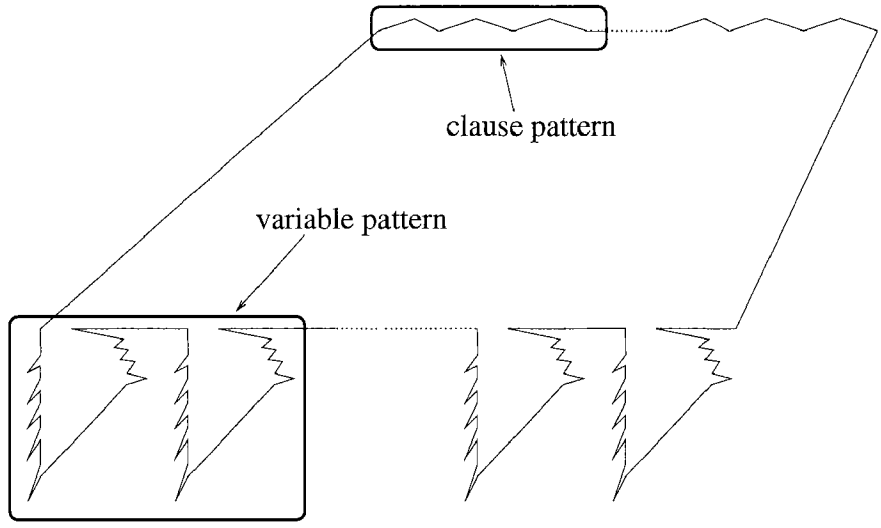


Figure 3.1: Schematic construction

We construct a variable pattern for each variable x_i as indicated in Fig. 3.2. Each variable pattern consists of a TRUE- and a FALSE-leg. Each leg has on its left boundary a maximum of five triangle-shaped dents with vertices f_k for $k = 1, \ldots, 5$. Each of these dents represents the lower part of a cone that connects the variable pattern to a clause pattern, in which the variable appears as a literal, as indicated in Fig. 3.3. Since these dents are triangle-shaped, we call the whole dent the *triangle of* f_k. All these triangles are attached to a single line as indicated in Fig. 3.2 on their right side, i.e.the "right" line segments of each triangle in a leg are collinear.

Each variable pattern contains at its right side exactly four dents with vertices v_1, v_2, v_3, and w. The construction is such that a point that is hiding in the triangle of any f_k sees the triangles of v_1, v_2, and v_3 completely, but it does not see the triangle of w. Therefore, we can hide in each leg a point at vertex w and additional points either in the triangles of f_k on the left side or in the triangles of v_k on the right side, but not in both. The idea is the following: if the variable is TRUE, then we hide points in the TRUE-leg of the variable pattern at vertices f_k, for $k = 1, \ldots, 5$ and at vertex w. In the FALSE leg, we hide points at vertices v_1, v_2, v_3, and w.

For each clause c_i we construct a clause pattern as indicated in Fig. 3.3. A clause pattern contsists of three dents, where the left and the right dent represent the literals of the clause. The middle dent represents the

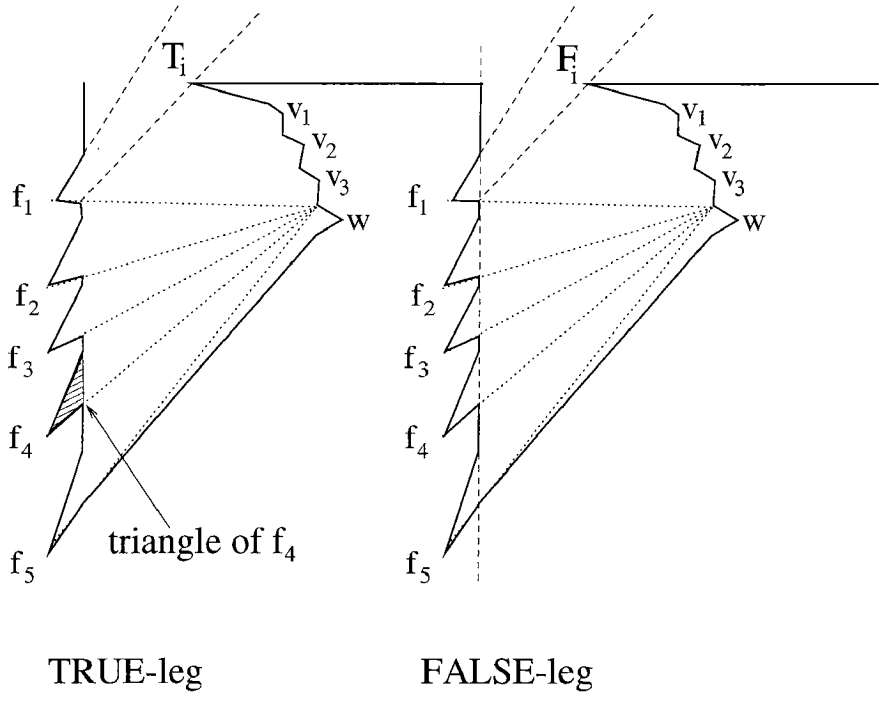


Figure 3.2: Variable pattern

truth value of the clause. The construction is such that we can hide three points in the clausepattern (i.e. one in each dent), exactly if the clause is satisfied. We can hide only two points (one in the left and one the right dent), otherwise. To achieve this we connect the variable patterns to the clause patterns with cones as illustrated in Fig. 3.3 for two variables x_i and x_j and a clause $(x_i, \neg x_j)$. This works accordingly for other types of clauses.

3.1.3 Analysis of the Reduction

We will show that this reduction is gap-preserving, i.e., it maps an NP-hard promise problem of MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY to an NP-hard promise problem of MAXIMUM HIDDEN SET ON POLYGON WITHOUT HOLES. The reduction has the following properties:

Lemma 3.1.1 *If there exists a truth assignment S to the variables of the* MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY *instance I that satisfies at least $(1-\epsilon)m$ clauses, then there exists a solution S' of the* MAXIMUM

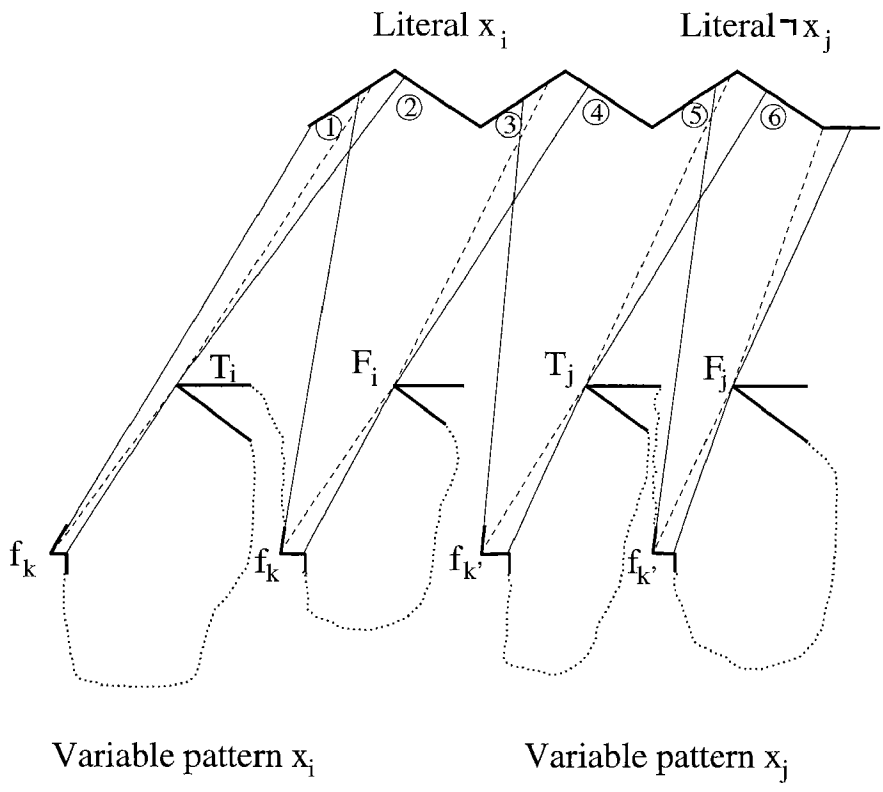


Figure 3.3: Clause pattern with cones

HIDDEN SET WITHOUT HOLES *instance* I' *with* $|S'| \geq 10n+2m+(1-\epsilon)m$.

Proof: If variable x_i is TRUE in S, then we let the vertices $f_1, \ldots, f_5$ and w of the TRUE-leg of x_i, as well as the vertices v_1, v_2, v_3 and w of the FALSE-leg of x_i be in the solution S'; vice-versa if x_i is FALSE in S. This gives us $10n$ points in S'.

The remaining points for S' are in the clause patterns. Figure 3.3 shows the clause pattern for a clause $(x_i, \neg x_j)$[1], together with the cones that link the clause pattern to the corresponding variable patterns. Remember that these cones are not part of the polygon boundary. To understand Fig. 3.3, assume x_i is assigned the value FALSE and x_j is assigned the value TRUE, i.e., the clause $(x_i, \neg x_j)$ is not satisfied. Then there is a point in the solution that sits at vertex f_k (for some k) in the FALSE-leg of x_i and a point that

[1] The proofs work accordingly for other types of clauses, such as (x_i, x_j).

sits at vertex f'_k (for some k') in the TRUE-leg of x_j. In this case, we can have only *two* additional points in the solution S' at points ①, ⑥. In the remaining three cases, where the variables x_i and x_j are assigned truth values such that the clause is satisfied, we can have *three* additional points in S' at ① – ⑥: If x_i and x_j are both FALSE in the solution S, then we hide points in the clause pattern at point ①, ④, and ⑤. If x_i and x_j are both TRUE, then we hide points in the clause pattern at point ②, ③, and ⑥. If x_i is TRUE and x_j is FALSE, then we can hide points in the clause pattern at point ② and ⑤, and one additional point at either ③ or ④.

Therefore, we have 2 points from all unsatisfied clauses and 3 points from all satisfied clauses, i.e. $2\epsilon m + 3(1-\epsilon)m$ points that are hidden in the clause patterns. Thus, $|S'| \geq 10n + 2m + (1-\epsilon)m$, as claimed. □

Lemma 3.1.2 *If there exists a solution S' of I' with $|S'| \geq 10n+3m-(\epsilon+\gamma)m$, then there exists a variable assignment S of I that satisfies at least $(1-\epsilon-\gamma)m$ clauses.*

Proof: For any solution S', we can assume that in each leg of each variable pattern, all points in S' are either in the triangles of vertices $f_1, \ldots, f_5$ and w, or in the triangles of vertices v_1, v_2, v_3, and w. To see this, note that there can be at most one point in each leg outside the triangles. This point completely sees at least one triangle, and we can move the point to the triangle, which results again in a feasible solution. Furthermore, any point in any triangle of $f_1, \ldots, f_5$ sees the triangles of v_1, v_2, v_3 completely (and vice-versa).[2]

As a first step in transforming the solution S', we move the points that are hiding in the triangles to the vertices f_k, v_1, v_2, v_3, and w respectively. This transformation is obviously no problem for the triangles of v_1, v_2, v_3, and w. Moving the points in the triangles of vertices f_k will sligthly change the cones that they see as indicated in Fig. 3.3 through dashed lines. However, we can still position at least the same number of points in the clause patterns. Thus, this transformation yields a feasible hidden set of points that consists of at least as many points as the original solution.

We now describe how to transform the solution S' (with $|S'| \geq 10n + 3m - (\epsilon+\gamma)m$) in such a way that it remains feasible, that its size (i.e. the number of hidden points) does not decrease, and that we will be able to assign truth values to the variables. This is done with an enumeration of all possible cases, i.e. we show how to transform the solution if there is a point hidden in 3, 4, or 5 of the triangles of the points $f_1, \ldots, f_5$ in the

[2] We need the triangle of w to ensure that, if there are points hiding in the triangles of the f_k's, there can be no additional point hiding outside the triangles in the leg.

TRUE-leg and the FALSE-leg of a variable pattern. Our goal is that at the end, we have for each variable pattern the six points at $f_1, \ldots, f_5$, and w from one leg in the solution and the 4 points v_1, v_2, v_3, and w from the other leg. Thus, we can easily obtain a truth assignment for the variables by letting variable x_i be TRUE iff the six points at $f_1, \ldots, f_5$, and w from the TRUE-leg are in the solution.

For each variable x_i, we argue as follows:

- If there are 5 points hidden at vertices f_k of the TRUE leg of x_i and 3 points at vertices v_l of the FALSE leg (or vice-versa), then we are already at the position that we want to get to.

- If there are 5 points hidden at vertices f_k of the TRUE leg of x_i and 5,4,3,2, or 1 point(s) at vertices f_k of the FALSE leg, then we delete the points in the FALSE leg and set 3 hidden points at vertices v_l of the FALSE leg. This yields a better solution, since the difference of guards in the variable pattern is -2,-1,0,+1,+2 and we can position 5,4,3,2,1 additional guards in the dents of the clause patterns that correspond to literals of x_i. These additional guards could not have been placed in the original solution, since the whole area of the dents was seen by the points at vertices f_k of both the TRUE and the FALSE leg. This works accordingly, if we switch the FALSE and the TRUE leg in the argument.

- If there are 4 points hidden at vertices f_k of the TRUE leg of x_i and 4,3,2, or 1 point(s) at vertices f_k of the FALSE leg, then we delete the points in the FALSE leg and set 3 hidden points at vertices v_l of the FALSE leg. Moreover, we place an additional point at the one vertex f_k in the TRUE leg, where there is not already a point hiding. Again, this yields a better solution, since the difference of guards in the variable pattern is 0,+1,+2,+3 and we can position at least 3,2,1,0 additional guards in the dents of the clause patterns that correspond to literals of x_i, which was impossible before. Because of the additional guard at f_k in the TRUE leg, we might lose at most one hidden point in a middle dent of a clause pattern. Again, this works accordingly, if we switch the FALSE and the TRUE leg in the argument.

- If there are 3 points hidden at vertices f_k of the TRUE leg of x_i and 3,2, or 1 point(s) at vertices f_k of the FALSE leg, then we delete the points in the FALSE leg and set 3 hidden points at vertices v_l of the FALSE leg. Moreover, we place two additional points at the two vertices f_k and $f_{k'}$ in the TRUE leg, where there is not already a point hiding.

Again, this yields a better solution, since the difference of guards in the variable pattern is +2,+3,+4 and we can position at least 1,0,0 additional guards in the dents of the clause patterns that correspond to literals of x_i. Because of the additional guards TRUE leg, we lose at most two hidden points in middle dents of clause patterns. Again, this works accordingly, if we switch the FALSE and the TRUE leg in the argument. We can argue in the same manner, if we have only two or one point hidden at the vertices f_k of the TRUE leg.

- If we have 3 hiding points in the v_l vertices of both legs, then we delete these points in the FALSE leg and place 5 additional points at the vertices f_k of the FALSE leg, if variable x_i appears as a negative literal more (or equally) often as as a positive literal. Again, we argue symmetrically, if there are more positive than negative literals. This yields a better or equally good solution, since we have 2 additional guards in the variable pattern and we lose at most 2 guards in the middle dents of clause patterns as we falsify at most 2 clauses.

- If we have 4,3 points hidden at vertices f_k of the TRUE leg of x_i and 3 points at vertices v_l of the FALSE leg, then we place 1, 2 additional guard(s) at the one vertex, the two vertices f_k in the TRUE leg, where there is not already a point hidden. This yields an equally good solution, since we have 1,2 additional points in the variable pattern and at most 1,2 fewer points in the middle dents of clause patterns. Again, this works accordingly, if we switch the FALSE and the TRUE leg.

As a last step in the transformation, we add points at each vertex w in each leg, if there are no points hiding there already.

Thus, the transformed solution S' consists of at least $10n+3m-(\epsilon+\gamma)m$ points, $10n$ of which lie in the variable patterns. At most 3 points can lie in each clause pattern. If 3 points lie in a clause pattern, then this clause is satisfied. Therefore, if 2 points lie in each clause pattern, there are still at least $(1-\epsilon-\gamma)m$ additional points in S'. These must lie in clause patterns as well. Therefore, at least $(1-\epsilon-\gamma)m$ clauses are satisfied. □

Lemmas 3.1.1 and 3.1.2 transform the promise problem of MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY into a promise problem of MAXIMUM HIDDEN SET WITHOUT HOLES. In the promise problem of MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY, we are promised that an optimum solution either satisfies at least $(1-\epsilon)m$ clauses or strictly less than $(1-\epsilon-\gamma)$

clauses for some constant $\epsilon, \gamma > 0$. For small enough values[3] of $\epsilon, \gamma > 0$, it is NP-hard to decide, which of the two cases is true. This follows from the fact that MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY is APX-complete. In the promise problem of MAXIMUM HIDDEN SET ON POLYGON WITHOUT HOLES, we are promised that either at least $10n + 3m - \epsilon m$ points can be hidden from each other, or strictly less than $10n + 3m - (\epsilon + \gamma)m$ points can be hidden from each other. Again, it is NP-hard to decide, what is true. Thus, MAXIMUM HIDDEN SET ON POLYGON WITHOUT HOLES cannot be approximated with an approximation ratio of:

$$\begin{aligned}
\frac{10n + 3m - \epsilon m}{10n + 3m - (\epsilon + \gamma)m} &= \frac{10n + 3m - \epsilon m - \gamma m}{10n + 3m - \epsilon m - \gamma m} + \frac{\gamma m}{10n + 3m - \epsilon m - \gamma m} \\
&= 1 + \frac{\gamma m}{10n + 3m - \epsilon m - \gamma m} \\
&\geq 1 + \frac{\gamma m}{m(33 - \epsilon - \gamma)} \\
&\geq 1 + \frac{\gamma}{33}
\end{aligned}$$

We have used that $m \geq n/3$. Thus:

Theorem 3.1.3 MAXIMUM HIDDEN SET ON POLYGON WITHOUT HOLES *is APX-hard.*

If we restrict the hidden set to consist of only vertices, we can use the same reduction and the same analysis with the modification that we introduce additional vertices in each clause pattern. More specifically, we replace each edge of all dents of the clause patterns by two, slightly convex edges that have their common endpoint right where the corresponding point ① – ⑥ from Fig. 3.3 is. Thus, the result carries over.

Theorem 3.1.4 MAXIMUM HIDDEN VERTEX SET ON POLYGON WITHOUT HOLES *is APX-hard.*

3.2 Hiding in Polygons with Holes

We can prove better hardness results for hiding in polygons with holes than for hiding in polygons without holes, as we propose a reduction from MAXIMUM CLIQUE that constructs a polygons with holes that very naturally corresponds to the graph.

[3] We need ϵ as a second parameter, since we can find out in polynomial time, whether all clauses of a MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY instance are satisfiable [41];only the optimization version of MAXIMUM 5-OCCURRENCE-2-SATISFIABILITY is NP-hard.

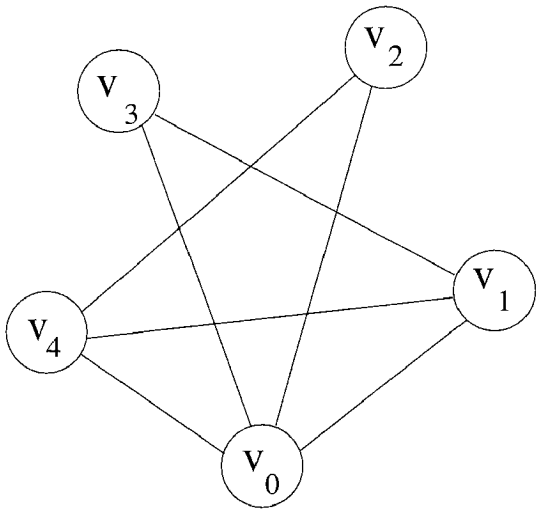


Figure 3.4: A graph with five vertices

Suppose we are given an instance I of MAXIMUM CLIQUE, i.e. an undirected graph $G = (V, E)$, where $V = v_0, \ldots, v_{n-1}$. Let $m := |E|$. Figure 3.4 shows an exapmle. We construct an instance I' of MAXIMUM HIDDEN SET ON POLYGON WITH HOLES as follows. I' consists of a polygon with holes. The polygon is basically a regular $2n$-gon with holes, but we replace every other vertex by a comb-like structure. Each hole is a small triangle designed to block the view of two combs from each other, whenever the two vertices, to which the combs correspond, are connected by an edge in the graph. Figure 3.5 shows as an example the polygon with holes constructed from the graph in Fig. 3.4. (Note that only the solid lines are lines of the polygon and also note that the combs are not shown in Fig. 3.5.)

Let the regular $2n$-gon consist of vertices $v_0, v'_0, \ldots, v_{n-1}, v'_{n-1}$ in counterclockwise order, to indicate that we map each vertex $v_i \in V$ in the graph to a vertex v_i in the polygon.

We need some notation, first. Let $e_{i,j}$ denote the intersection point of the line segment from v'_{i-1} to v'_i with the line segment from v_i to v_j, as indicated in Fig. 3.6. (Note that we make liberal use of the notation index for the vertices, i.e. v_{i+1} is strictly speaking $v_{i+1 \bmod n}$, accordingly for

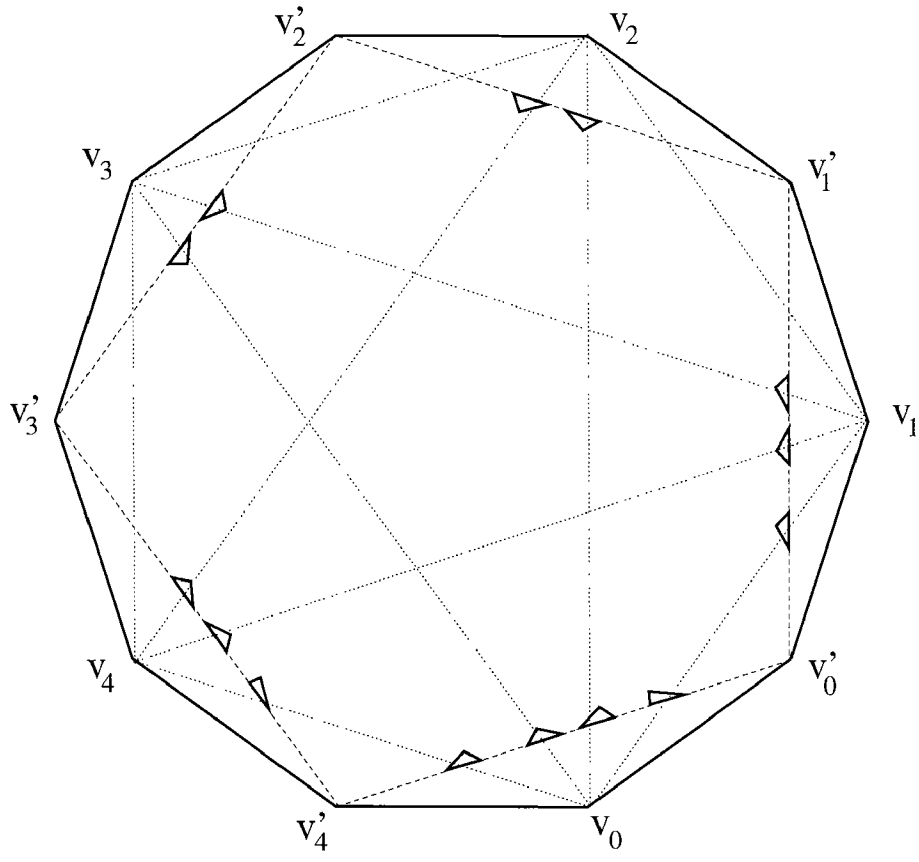


Figure 3.5: Polygon constructed from the graph in Fig. 3.4

v_{i-1}.) Let d denote the minimum of the distances of $e_{i,j}$ from $e_{i,j+1}$, where the minimum is taken over all $i, j = 1, \ldots, n$. Let $e^-_{i,j}$ ($e^+_{i,j}$) denote the point at distance $\frac{d}{3}$ from $e_{i,j}$ on the line from v'_{i-1} to v'_i that is closer to v'_{i-1} (v'_i). Let m_i be the midpoint of the line segment from vertex v_i to v_{i+1} and let m'_i be the intersection point of the line from v'_i to m_i and from $e^+_{i,i+1}$ to $e^-_{i+1,i}$ (see Fig. 3.6). Finally, let $e^T_{i,j}$ denote the intersection point of the line from $e^-_{i,j}$ to m'_i and the line from $e^+_{i,j}$ to m'_{i-1}. The detailed construction of these points is shown in Fig. 3.6.

We let the triangle formed by the three vertices $e^+_{i,j}$, $e^-_{i,j}$, and $e^T_{i,j}$ be a hole in the polygon iff there exists an edge in G from v_i to v_j. Recall Fig. 3.5, which gives an example.We now refine the polygon obtained so far by

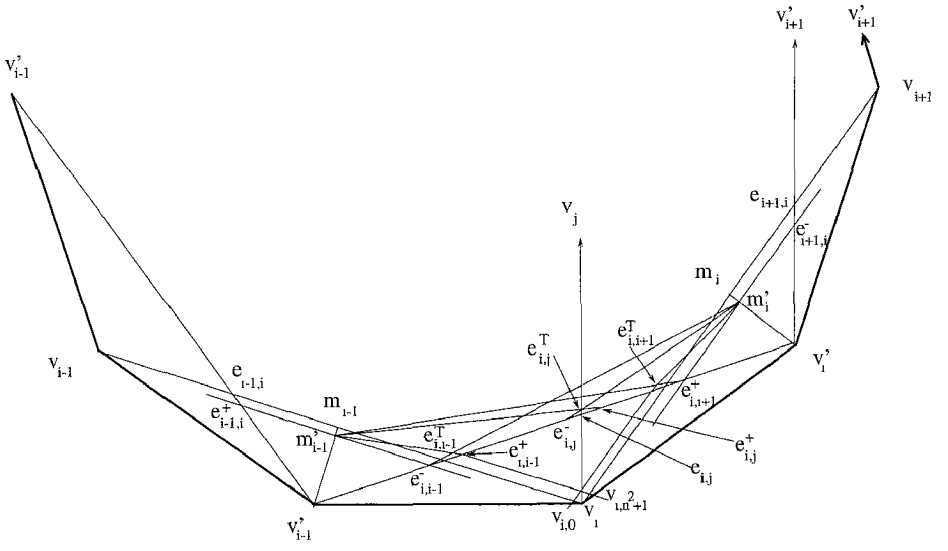


Figure 3.6: Points $e_{i,j}^-$, $e_{i,j}^+$, and $e_{i,j}^T$

cutting off a small portion at each vertex v_i. For each $i \in \{0, \ldots, n\}$, we introduce two new vertices $v_{i,0}$ and v_{i,n^2+1} as indicated in Fig. 3.6. Vertex $v_{i,0}$ is defined as the intersection point of the line that is parallel to the line from v_{i-1} to v_i and goes through point $e_{i,i-1}^+$ and of the line from v_i to v_i'. Symmetrically, vertex v_{i,n^2+1} is defined as the intersection point of the line that is parallel to the line from v_{i+1} to v_i and that goes through point $e_{i,i+1}^-$ and of the line from v_i to v_{i-1}'.

We fix $n^2 - 1$ additional vertices $v_{i,1}, \ldots, v_{i,n^2}$ on the line segment from $v_{i,0}$ to v_{i,n^2+1} for each i as shown in Fig. 3.7. For a fixed i, the two vertices $v_{i,l}$ and $v_{i,l+1}$ have equal distance for all $l \in \{0, \ldots, n^2\}$. Finally, we fix n^2 additional vertices $w_{i,l}$ for $l \in \{0, \ldots, n^2\}$ for each i. Vertex $w_{i,l}$ is defined as the intersection point of the line from vertex v_{i-1}' through $v_{i,l}$ with the line from vertex vertex v_i' through $v_{i,l+1}$. The polygon between two vertices v_{i-1}' and v_i' is now given by the following ordered sequence of vertices:

$$v_{i-1}', v_{i,0}, w_{i,0}, v_{i,1}, w_{i,1}, \ldots, v_{i,n^2}, w_{i,n^2}, v_{i,n^2+1}, v_i'$$

as indicated in Fig. 3.7. We call the set of all triangles $v_{i,l}, w_{i,l}, v_{i,l+1}$ for a fixed i and all $l \in \{0, \ldots, n^2\}$ the *comb of* v_i.

We have the following property of the construction.

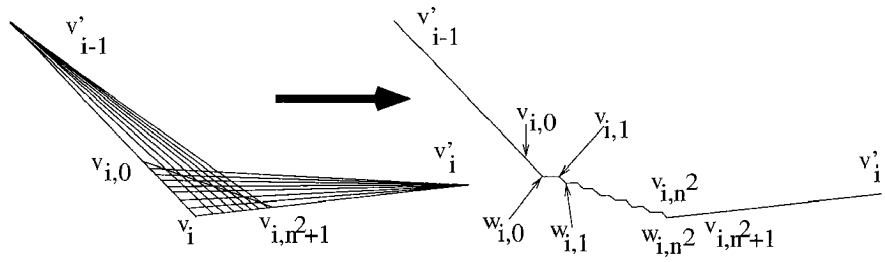


Figure 3.7: Construction of the comb of v_i

Lemma 3.2.1 *In any feasible solution S' of the* MAXIMUM HIDDEN SET ON POLYGON WITH HOLES *instance I', at most $2n$ points in S can be placed outside the combs.*

Proof: In each of the n trapezoids $\{v'_{i-1}, v'_i, v_{i,n^2+1}, v_{i,0}\}$ (see Figs. 3.5 and 3.6), there can be at most one point, which gives n points in total. Moreover, by our construction any point p in the trapezoid $\{v'_{i-1}, v'_i, m'_i, m'_{i-1}\}$ (not in the holes) can see every point p' in the n-gon $\{v'_0, \ldots, v'_n\}$ except for points p' in any of the holes and (possibly) except for points p' in the triangles $\{v'_{i-1}, m'_{i-1}, e^+_{i-1,i}\}$ and $\{v'_i, m'_i, e^-_{i+1,i}\}$ (see Fig. 3.6). Therefore, all points in S' that lie in the n-gon $\{v'_0, \ldots, v'_n\}$ must lie in only one of the n polygons $\{e^+_{i-1,i}, m'_{i-1}, m'_i, e^-_{i+1,i}, v'_i, v'_{i-1}\}$. Obviously, at most n points can be hidden in any one of these polygons.

□

We have the following observation, which follows directly from the construction:

Observation 3.2.2 *Any point in the comb of v_i completely sees the comb of vertex v_j, if (v_i, v_j) is not an edge in the graph. If (v_i, v_j) is an edge in the graph, then no point in the comb of v_i sees any point in the comb of v_j*

Given a feasible solution S' of the MAXIMUM HIDDEN SET ON POLYGON WITH HOLES instance I', we obtain a feasible solution S of the MAXIMUM CLIQUE instance I as follows: A vertex $v_i \in V$ is in the solution S, iff at least one point from S' lies in the comb of v_i. To see that S is a feasible solution, assume by contradiction that it is not a feasible solution. Then, there exists a pair of vertices $v_i, v_j \in S$ with no edge between them. But then, there is by construction no hole in the polygon to block the view between the comb of v_i and the comb of v_i.We need to show that the construction of I' can be done in polynomial time and that a feasible solution can be transformed

in polynomial time. There are $2n^2 + 1$ vertices in each of the n combs. We have additional n vertices v_i'. There are 2 holes for each edge in the graph and each hole consists of 3 vertices. Therefore, the polygon consists of $6m + 2n^3 + 2n$ vertices. It is known in computational geometry that the coordinates of intersection points of lines with rational coefficients can be expressed with polynomial length. All of the points in our construction are of this type. Therefore, the construction is polynomial. The transformation of a feasible solution can obviously be done in polynomial time.

We obtain our inapproximability result, again, by using the technique of gap-preserving reductions, which consists of transforming a promise problem into another promise problem. Let OPT denote the size of an optimum solution of the MAXIMUM CLIQUE instance I, let OPT' denote the size of an optimum solution of the MAXIMUM HIDDEN SET ON POLYGON WITH HOLES instance I', let $k \leq n$, and let $\epsilon > 0$.

Lemma 3.2.3 $OPT \geq k \Longrightarrow OPT' \geq n^2 k$

Proof: If $OPT \geq k$, then there exists a clique in I of size k. We obtain a solution for I' of size $n^2 k$ by simply letting the n^2 vertices $w_{i,l}$ for $l \in \{0, \ldots, n^2\}$ be in the solution if and only if vertex $v_i \in V$ is in the clique. The solution thus obtained for I' is feasible (see Observation 3.2.2). □

Lemma 3.2.4 $OPT < \frac{k}{n^{1/2-\epsilon}} \Longrightarrow OPT' < \frac{n^2 k}{n^{1/2-\epsilon}} + 2n$

Proof: We prove the contraposition: $OPT' \geq \frac{n^2 k}{n^{1/2-\epsilon}} + 2n \Longrightarrow OPT \geq \frac{k}{n^{1/2-\epsilon}}$.Suppose we have a solution of I' with $\frac{n^2 k}{n^{1/2-\epsilon}} + 2n$ points. At most $2n$ of the points in the solution can be outside the combs, because of Lemma 3.2.1. Therefore, at least $\frac{n^2 k}{n^{1/2-\epsilon}}$ points must be in the combs. From the construction of the combs, it is clear that at most n^2 points can hide in each comb. Therefore, the number of combs that contain at least one point from the solution is at least $\frac{\frac{n^2 k}{n^{1/2-\epsilon}}}{n^2} = \frac{k}{n^{1/2-\epsilon}}$. The transformation of a solution as described above yields a solution of I with at least $\frac{k}{n^{1/2-\epsilon}}$ vertices. □

Lemmas 3.2.3 and 3.2.4 transform the NP-hard promise problem of MAXIMUM CLIQUE, where we are promised that the optimum solution consists of either at least k or strictly less than $\frac{k}{n^{1/2-\epsilon}}$ vertices, into the NP-hard promise problem of MAXIMUM HIDDEN SET ON POLYGON WITH HOLES, where we are promised that an optimum solution consists of either at least $n^2 k$ or strictly less than $\frac{n^2 k}{n^{1/2-\epsilon}} + 2n$ hidden points. Thus, MAXIMUM HIDDEN

SET ON POLYGON WITH HOLES cannot be approximated with an approximation ratio of:

$$\frac{n^2k}{\frac{n^2k}{n^{1/2-\epsilon}}+2n} > \frac{n^{\frac{1}{2}-\epsilon}}{2}$$

where we have assumed that $k \geq 2$. We need to express the number of graph vertices n by the number of polygon vertices $|I'|$ of the polygon of instance I'. Note that $|I'| \leq 10n^3$ and therefore $n \geq \frac{|I'|^{\frac{1}{3}}}{3}$. Thus:

$$\frac{n^{\frac{1}{2}-\epsilon}}{2} \geq \frac{\frac{|I'|^{\frac{1}{6}-\frac{\epsilon}{3}}}{3^{\frac{1}{2}-\epsilon}}}{2} > \frac{|I'|^{\frac{1}{6}-\frac{\epsilon}{3}}}{4}$$

This yields the main result of this section.

Theorem 3.2.5 MAXIMUM HIDDEN VERTEX SET ON POLYGON WITH HOLES *cannot be approximated by any polynomial time algorithm with an approximation ratio of* $\frac{|I'|^{1/6-\gamma}}{4}$, *where* $|I'|$ *is the number of vertices in the polygon, and where* $\gamma > 0$, *unless* $NP = P$.

If we restrict the hidden set to contain only vertices, we can use the same construction. Actually, we do not need the combs, as our construction guarantees that in any solution there can be at most 2 points hiding at vertices other than v_i. This leads to a different promise problem of MAXIMUM HIDDEN VERTEX SET OF POLYGON WITH HOLES that the promise problem of MAXIMUM CLIQUE is mapped to, namely the promise problem, where we are promised that an optimum solution consists of either at least k vertices or strictly less than $\frac{k}{n^{\frac{1}{2}-\epsilon}} + 2$ vertices. Straight-forward analysis, using the fact that $|I'| \leq 5n^2$, leads to the follwing result.

Theorem 3.2.6 MAXIMUM HIDDEN VERTEX SET ON POLYGON WITH HOLES *cannot be approximated by any polynomial time algorithm with an approximation ratio of* $\frac{|I'|^{1/4-\gamma}}{4}$, *where* $|I'|$ *is the number of vertices in the polygon, and where* $\gamma > 0$, *unless* $NP = P$.

3.3 Hiding in Terrains

Theorem 3.3.1 *The problems* MAXIMUM HIDDEN SET ON TERRAIN *(*MAXIMUM HIDDEN VERTEX SET ON TERRAIN*) cannot be approximated by any polynomial time algorithm with an approximation ratio of* $\frac{|I''|^{1/6-\gamma}}{4}$ *(*$\frac{|I''|^{1/4-\gamma}}{4}$*), where* $|I''|$ *is the number of vertices in the terrain, and where* $\gamma > 0$, *unless* $NP = P$.

Proof: The proof very closely follows the lines of the proof for the inapproximability of MAXIMUM HIDDEN (VERTEX) SET ON POLYGON WITH HOLES. We use the same construction, but given the polygon with holes of instance I' we create a terrain (i.e. instance I'') by simply letting all the area outside the polygon (including the holes) have height h and by letting the area in the interior have height 0. This transformation from a polygon with holes to a terrain is actually the same as described in the chapter on guarding Chapt. 2.

We add four vertices to the terrain by introducing a rectangular bounding box around the regular $2n$-gon. This yields a terrain with vertical walls, which can be easily modified to have steep but not vertical walls, as required by the definition of a terrain. Finally, we triangulate the terrain. The terrain thus obtained looks like a canyon of a type that can be found in the south-west of the United States. □

3.4 Inapproximability Results under Stronger Assumptions

All inapproximability results obtained for hiding in polygons with holes and in terrains rely on the inapproximability result for MAXIMUM CLIQUE that says [29] that this problem cannot be approximated with an approximation ratio of $n^{\frac{1}{2}-\epsilon}$, unless $NP = P$. There is a stronger inapproximability result for MAXIMUM CLIQUE (also from [29]) that says that this problem cannot be approximated with an approximation ratio of $n^{1-\epsilon}$, unless $NP = coR$. We could base our inapproximability results for hiding on this stronger inapproximability result, and thus also get slightly stronger inapproximability results. However, we chose to omit this tedious exercise and encourage the reader to extend our inapproximability results.

Chapter 4

Convex Covering

4.1 Introduction

MINIMUM CONVEX COVER shares with MINIMUM POINT GUARD WITH-(-OUT) HOLES the property that its corresponding decision version is not known to be in NP. Despite of this, we are able to propose a non-trivial approximation algorithm for MINIMUM CONVEX COVER that achieves an approximation ratio that is logarithmic in the number of vertices of the input polygon; no comparable algorithm is known for point guard problems. The secret to the approximation algorithm lies in the fact that we can "discretize" the plane successfully by partitioning the interior of a given polygon into basic components. This step could be applied iteratively, but it suffices to do this for only two iterations for our purpose. Since each iteration produces a partition of only polynomial size, we could iterate for any constant number of steps. However, the degree of the polynomial describing the size of the partition grows exponentially in the number of iterations. Therefore, even our approximation algorithm, which uses only two iterations, will hardly be of practical importance, unless a way is discovered of significantly reducing running time.

In a second part of this chapter, we show APX-hardness for MINIMUM CONVEX COVER by modifying and interpreting an already known reduction [12] in the context of gap-preserving reductions.

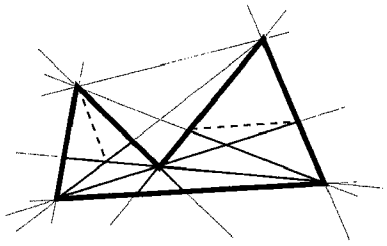


Figure 4.1: Construction of first-order basic triangles

4.2 A Logarithmic Approximation Algorithm

4.2.1 From the Continuous to the Discrete

In this section, we consider simple input polygons with and without holes. Let V_T denote the set of vertices (including the vertices of holes, if any) of a given polygon T. While, in the general MINIMUM CONVEX COVER problem, the vertices of the convex polygons that cover the input polygon can be positioned anywhere in the interior or on the boundary of the input polygon, we restrict their positions in an intermediate step: They may only be positioned on a quasi-grid in the RESTRICTED MINIMUM CONVEX COVER problem.

In order to define the RESTRICTED MINIMUM CONVEX COVER problem more precisely, we partition the interior of a polygon T into *convex components* (as proposed in [26] and already mentioned and generalized in the different context of guarding problems in Sect. 2.5) by drawing a line through each pair of vertices of T. We then triangulate each convex component arbitrarily. We call the triangles thus obtained *first-order basic triangles*. Figure 4.1 shows in an example the first-order basic triangles of a polygon (thick solid lines) with an arbitrary triangulation (fine solid lines and dashed lines). If a polygon T consists of n vertices, drawing a line through each pair of vertices of T will yield less than $\binom{n}{2} \cdot \binom{n}{2} \in O(n^4)$ intersection points. Let V_T^1 be the set of these intersection points that lie in T (in the interior or on the boundary). Note that $V_T \subseteq V_T^1$. The first-order basic triangles are a triangulation of V_T^1 inside T, therefore the number of first-order basic triangles is also $O(n^4)$. The RESTRICTED MINIMUM CONVEX COVER problem asks for a minimum number of convex polygons, with vertices restricted to V_T^1, that together cover the input polygon T. We call V_T^1 a quasi-grid that is imposed on T. For solving the RESTRICTED MINIMUM CONVEX COVER problem, we make use of a finer quasi-grid: Simply partition T by drawing lines through each pair of points from V_T^1. This

yields again convex components, and we triangulate them again arbitrarily. This higher resolution partition yields $O(n^{16})$ intersection points, which define the set V_T^2. We call the resulting triangles *second-order basic triangles*. Obviously, there are $O(n^{16})$ second-order basic triangles. Note that $V_T \subseteq V_T^1 \subseteq V_T^2$.

4.2.2 Optimum Solution of MINIMUM CONVEX COVER vs. Optimum Solution of RESTRICTED MINIMUM CONVEX COVER

The quasi-grids V_T^1 and V_T^2 serve the purpose of making a convex cover computationally efficient while at the same time guaranteeing that the cover on the discrete quasi-grid is not much worse than the desired cover in continuous space. The following theorem proves the latter.

Theorem 4.2.1 *Let T be an arbitrary simple input polygon with n vertices. Let OPT denote the size of an optimum solution of* MINIMUM CONVEX COVER *with input polygon T and let OPT' denote the size of an optimum solution of* RESTRICTED MINIMUM CONVEX COVER *with input polygon T. Then:*

$$OPT' \leq 3 \cdot OPT$$

Proof: We proceed as follows: We show how to *expand* a given, arbitrary convex polygon C to another convex polygon C' with $C \subseteq C'$ by iteratively expanding edges. We then replace the vertices in C' by vertices from V_T^1, which results in a (possibly) non-convex polygon C'' with $C' \subseteq C''$. Finally, we describe how to obtain three convex polygons C_1'', C_2'', C_3'' with $C'' = C_1'' \cup C_2'' \cup C_3''$ that only contain vertices from V_T^1. This will complete the proof, since each convex polygon from an optimum solution of MINIMUM CONVEX COVER can be replaced by at most 3 convex polygons that are in a solution of RESTRICTED MINIMUM CONVEX COVER. Following this outline, let us present the proof details.

Let C be an arbitrary convex polygon inside polygon T. Let the vertices of C be given in in clockwise order. We obtain a series of convex polygons $C^1, C^2, \ldots, C'$ with $C = C^0 \subseteq C^1 \subseteq C^2 \subseteq \ldots \subseteq C'$, where C^{i+1} is obtained from C^i as follows (see Fig. 4.2):

Let a, b, c, d be consecutive vertices (in clockwise order) in the convex polygon C^i that lies inside polygon T. Let vertices $b, c \notin V_T$, with b and c not on the same edge of T. Then, the edge (b, c) is called *expandable*. If there exists no expandable edge in C^i, then $C' = C^i$, which means we have found the end of the series of convex polygons. If (b, c) is an expandable edge, we *expand* the edge from vertex b to vertex c as follows:

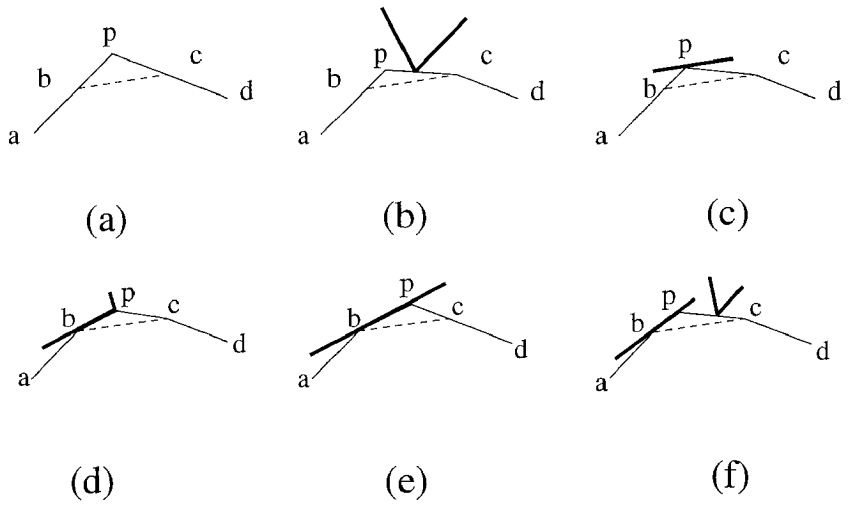


Figure 4.2: Expansion of edge (b,c)

- If b does not lie on the boundary of T, then we let a point p start on b and move on the halfline through a and b away from b until either one of two events happens: p lies on the line through c and d, or the triangle p, c, b touches the boundary of T. Fix p as soon as the first of these events happens.

  Figure 4.2 shows a list of all possible cases, where the edges from polygon T are drawn as thick edges: Point p either lies on the intersection point of the lines from a through b and from c through d as in case (a), or there is a vertex v_l on the line segment from p to c as in case (b), or p lies on an edge of T as in case (c).

- If b lies on the boundary of T, i.e. on some edge of T, say from v_k to v_{k+1}, then let p move as before, except that the direction of the move is now on the way from v_k through b up until v_{k+1} at most (instead of the ray from a through b).

  Figure 4.2 shows a list of all possible cases: Point p either lies at vertex v_{k+1} as in case (d), or on the intersection point of the lines from b to v_{k+1} and from d through c as in case (e), or there is a vertex v_l on the line segment from p to c as in case (f).

A new convex polygon C_p^i is obtained by simply adding point p as a vertex in the ordered set of vertices of C^i between the two vertices b and c.

Furthermore, eliminate all vertices in C_p^i that have collinear neighbors and that are not vertices in V_T.

Note that an edge from two consecutive vertices b and c with $b, c \notin V_T$ can always be expanded in such a way that the triangle b, p, c that is added to the convex polygon is non-degenerate, i.e., has non-zero area, unless b and c both lie on the same edge of polygon T. This follows from the cases (a) - (f) of Fig. 4.2.

Now, let $C^{i+1} = C_p^i$, if either a new vertex of T has been added to C_p^i in the expansion of the edge, which is true in cases (b), (d), and (f), or the number of vertices of C_p^i that are not vertices of T has decreased, which is true in case (a). If p is as in case (c), we expand the edge (p, c), which will result in either case (d), (e), or (f). Note that in cases (d) and (f), we have found C^{i+1}. If p is as in case (e), we expand the edge (p, d), which will result in either case (d), (e), or (f). If it is case (e) again, we repeat the procedure by expanding the edge from p and the successor (clockwise) of d. This needs to be done at most $|C^i|$ times, since the procedure will definitely stop once it gets to vertex a. Therefore, we obtain C^{i+1} from C^i in a finite number of steps. Let τ_i denote the number of vertices in C^i that are also vertices in T and let $\hat{\tau}_i$ be the number of vertices in C^i that are not vertices in T. Now note that $\phi(i) = \hat{\tau}_i - 2\tau_i + 2n$ is a function that bounds the number of remaining steps, i.e., it strictly decreases with every increase in i and cannot become negative. The existence of this bounding function implies the finiteness of the series $C^1, C^2, \ldots, C'$ of convex polygons.

By definition, there are no expandable edges left in C'. Call a vertex of C' a T-vertex, if it is a vertex in T. From the definition of expandable edges, it is clear that there can be at most two non-T-vertices between any two consecutive T-vertices in C', and if there are two non-T-vertices between two consecutive T-vertices, they must both lie on the same edge in T. Let the T-vertices in C' be $t_1, \ldots, t_l$ in clockwise order, and let the non-T-vertices between t_i and t_{i+1} be $\tilde{t}_{i,1}$ and $\tilde{t}_{i,2}$ if they exist. We now replace each non-T-vertex $\tilde{t}_{i,j}$ in C' by one or two vertices $\hat{t}_{i,j}^1$ and $\hat{t}_{i,j}^2$ that are both elements of V_T^1. This will transform the convex polygon C' into a non-convex polygon C'' (we will show later how C'' can be covered by at most three convex polygons C_1'', C_2'', C_3'').

To this end, let a, b, c be the first-order basic triangle in which non-T-vertex $\tilde{t}_{i,j}$ lies, as illustrated in Fig. 4.3. Points a, b, c are all visible from both vertices t_i and t_{i+1}. To see this, assume by contradiction that the view from, say, t_i to a is blocked by an edge e of T. Since $\tilde{t}_{i,j}$ must see t_i, the edge e must contain a vertex e' in the triangle $t_i, a, \tilde{t}_{i,j}$, but then a cannot be a vertex of the first-order basic triangle in which $\tilde{t}_{i,j}$ lies, since the line from vertex t_i through vertex e' would cut through the first-order

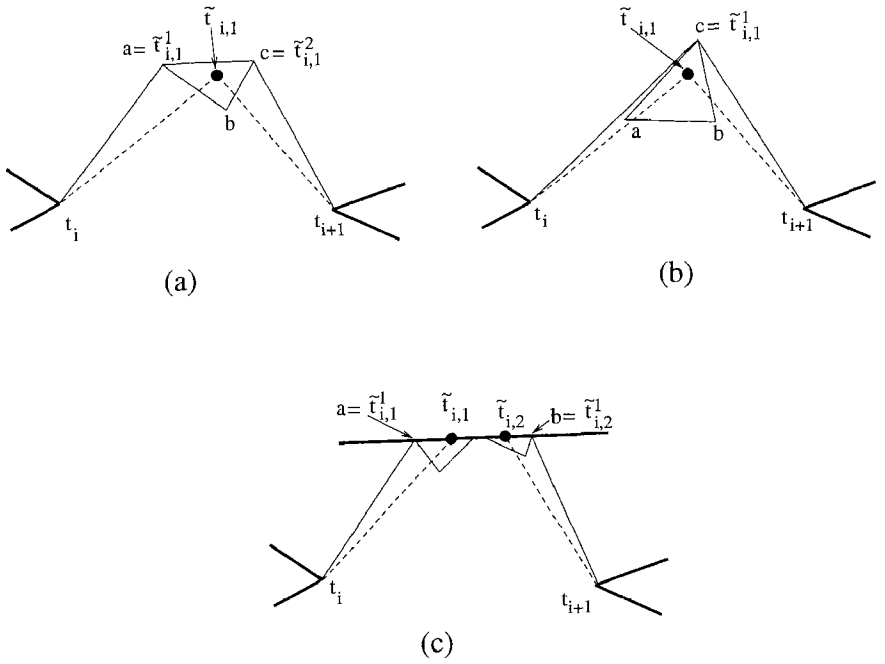


Figure 4.3: Replacing non-T-vertices

basic triangle, an impossibility. Now, let d_i be the intersection point of the line from t_{i-1} through t_i and the line from t_{i+1} through t_{i+2}. With similar arguments, the triangle t_i, d_i, t_{i+1} completely contains triangle a, b, c.

Assume that only one non-T-vertex $\tilde{t}_{i,1}$ exists between t_i and t_{i+1}. If the triangle formed by t_i, t_{i+1} and a completely contains the triangle $t_i, \tilde{t}_{i,1}, t_{i+1}$, we let $\tilde{t}^1_{i,1} = a$, likewise for b and c (see Fig. 4.3 (b)). Otherwise, we let $(\tilde{t}^1_{i,1}, \tilde{t}^2_{i,1})$ be $(a, b), (a, c)$, or (b, c) as in Fig. 4.3 (a), such that the polygon $t_i, \tilde{t}^1_{i,1}, \tilde{t}^2_{i,1}, t_{i+1}$ is convex and completely contains the triangle $t_i, \tilde{t}_{i,1}, t_{i+1}$. This is always possible by the definition of points a, b, c.

Now, assume that two non-T-vertices $\tilde{t}_{i,1}$ and $\tilde{t}_{i,2}$ exist between t_i and t_{i+1}. From the definition of C', we know that $\tilde{t}_{i,1}$ and $\tilde{t}_{i,2}$ must lie on the same edge e of T. Therefore, the basic triangle in which $\tilde{t}_{i,1}$ lies must contain a vertex a either at $\tilde{t}_{i,1}$ or preceeding $\tilde{t}_{i,1}$ on edge e along T in clockwise order.

Let $\tilde{t}^1_{i,1} = a$. The basic triangle in which $\tilde{t}_{i,2}$ lies must contain a vertex b either at $\tilde{t}_{i,2}$ or succeeding $\tilde{t}_{i,2}$ on edge e. Let $\tilde{t}^1_{i,2} = b$. See Fig. 4.3 (c). Note that the convex polygon $t_i, \tilde{t}^1_{i,1}, \tilde{t}^1_{i,2}, t_{i+1}$ completely contains the

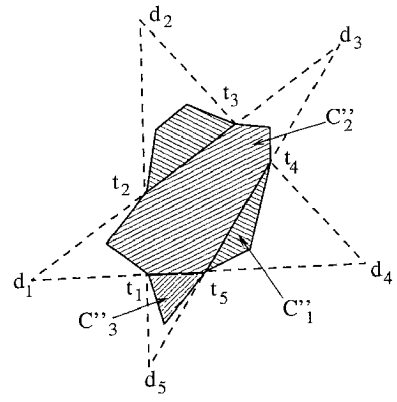


Figure 4.4: Covering C'' with three convex polygons

polygon $t_i, \tilde{t}_{i,1}, \tilde{t}_{i,2}, t_{i+1}$.

After applying this change to all non-T-vertices in C', we obtain a (possibly) non-convex polygon C''. First, assume that C'' contains an odd number f of T-vertices. We let C''_1 be the polygon defined by vertices $t_i, \tilde{t}^k_{i,j}$ and t_{i+1} for all j, k and for all odd i, but $i \neq f$. By construction, C''_1 is convex. Let C''_2 be the polygon defined by vertices $t_i, \tilde{t}^k_{i,j}$ and t_{i+1} for all j, k and for all even i. Finally, let C''_3 be the polygon defined by vertices $t_f, \tilde{t}^k_{f,j}$ and t_1 for all j, k. Figure 4.4 shows an example. Obviously, C''_1, C''_2, and C''_3 are convex and together cover all of C''. Second, assume that C'' contains an even number of T-vertices, and cover it with only two convex polygons using the same concept. This completes the proof. □

4.2.3 Finding Maximum Convex Polygons

Assume that each second-order basic triangle from a polygon T is assigned a weight value of either 1 or 0. In this section, we present an algorithm using dynamic programming that computes the convex polygon M in a polygon T that contains a maximum number of second-order basic triangles with weight 1 and that only has vertices from V^1_T. For simplicity, we call such a polygon a *maximum convex polygon*. The weight of a polygon M is defined as the sum of the weights of the second-order basic triangles in the polygon and is denoted by $|M|$. We will later use the algorithm described below to iteratively compute a maximum convex polygon with respect to the triangles that are not yet covered, to eventually obtain a convex cover for T.

Let $a, b, c \in V^1_T$. Let $P_{a,b,c}$ denote the maximum convex polygon that:

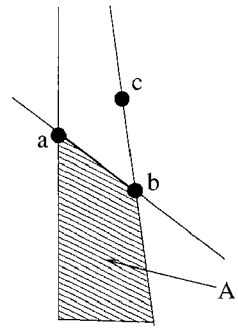


Figure 4.5: Dynamic Programming

- contains only vertices from V_T^1, and
- contains vertices a, b, c in counterclockwise order, and
- has a as its left-most vertex, and
- contains additional vertices only between vertices a and b.

Given three vertices $a, b, c \in V_T^1$, let A be the (possibly infinite) area of points that are:

- to the right of vertex a, and
- to the left of the line oriented from b through a, and
- to the left of the line oriented from b through c.

For an illustration, see Fig. 4.5. Let

$$P'_{a,b,c} = \max_{d \in V_T^1 \cap A} P_{a,d,b} \cup \Delta a, b, c,$$

where max is defined as follows (to simplify notation):

$$\max\{P_1, P_2\} = \begin{cases} P_1 & \text{if } |P_1| \geq |P_2| \\ P_2 & \text{otherwise} \end{cases} .$$

Lemma 4.2.2 *$P_{a,b,c} = P'_{a,b,c}$, if the triangle a, b, c is completely contained in the polygon T.*

Proof: Consider $P_{a,b,c}$, which is maximum by definition. $P_{a,b,c}$ must contain additional vertices between a and b (otherwise the lemma is trivially

true). Let d' be the predecessor of b in the counterclockwise order of $P_{a,b,c}$. Vertex d' must lie in area A as defined above, otherwise the polygon a, d', b, c would either be non-convex, not have a as its left-most vertex, or not be in the required counterclockwise order. Now consider $P'' = P_{a,b,c} - \Delta a, b, c$. From the definition of area A it is clear the P'' can only contain vertices that lie in A. Now $P_{a,d',b}$ is maximum by definition, and it is considered when computing $P'_{a,b,c}$. □

Let M be a maximum convex polygon for a polygon T with weights assigned to the second-order basic triangles. Let a be the left-most vertex of M, let c be the predecessor of a in M in counterclockwise order, and let b be the predecessor of c. Then $|P_{a,b,c}| = |M|$ by definition.

We will now use Lemma 4.2.2 to construct an algorithm, which takes as input a polygon T and an assignment of weight 0 or 1 to each second-order basic triangle of T and computes the maximum convex polygon. To this end, we fix vertex $a \in V_T^1$. Let a' be a point with the same x-coordinate and smaller y-coordinate than a. Now, order all other vertices $b \in V_T^1$ to the right of a according to the angle formed by b, a, a'. Let the resulting ordered set be B and let B' be the empty set. Take the smallest element b from B, remove it from B and add it to set B', then for all $c \in V_T^1 \backslash B'$ and to the right of a, compute weight $|\Delta a, b, c|$ of the triangle a, b, c and compute $P_{a,b,c}$ according to Lemma 4.2.2. Compute weight $|P_{a,b,c}|$ by adding $|\Delta a, b, c|$ to $|P_{a,d,b}|$, where d is the maximizing argument. Note that the computation of $P_{a,b,c}$ according to Lemma 4.2.2 is always possible, since all possible vertices d in $P_{a,d,b}$ lie to the left of the line from b to a (see also definition of area A) and have therefore smaller angles d, a, a' than b, a, a', and have therefore already been computed. The algorithm is executed for every $a \in V_T^1$, and the maximum convex polygon found is returned.

Note that $|T| = n$, $|V_T^1| = O(n^4)$, and $|V_T^2| = O(n^{16})$. Ordering $O(n^4)$ vertices takes $O(n^4 \log n)$ time. Computing the weight of a triangle takes $O(n^{16})$ time. Computing $P_{a,b,c}$ takes $O(n^4)$ time. We have to compute the weight of $O(n^8)$ triangles, which gives a total time of $O(n^{24})$. Finally, we have to execute our algorithm for each $a \in V_T^1$, which gives a total running time of $O(n^{28})$. Space requirements are $O(n^{12})$ by using pointers.

4.2.4 An Approximation Algorithm for Minimum Convex Cover

Given a polygon T, we obtain a convex cover by iteratively applying the algorithm for computing a maximum convex polygon from Sect. 4.2.3. It works as follows for an input polygon T.

1. Let all second-order basic triangles have weight 1. Let $S = \emptyset$.

2. Find the maximum convex polygon M of polygon T using the algorithm from Sect. 4.2.3, and add M to the solution S. Decrease the weight of all second-order basic triangles that are contained in M to 0.[1]

3. Repeat step 2 until there are no second-order basic units with weight 1 left. Return S.

To obtain a performance guarantee for this algorithm, consider the MINIMUM SET COVER instance I, which has all second-order basic triangles as elements and where the second-order basic triangles with weight 1 of each convex polygon in T, which only contains vertices from V_T^1, form a set in I. The greedy heuristic for MINIMUM SET COVER achieves an approximation ratio of $1 + \ln n'$, where n' is the number of elements in I [31] and it works in exactly the same way as our algorithm. However, we do not have to (and could not afford to) compute all the sets of the MINIMUM SET COVER instance I (which would be a number exponential in n'): It suffices to always compute a set, which contains a maximum number of elements not yet covered by the solution thus far. This is achieved by reducing the weights of the second-order basic triangles already in the solution to 0; i.e. a convex polygon with maximum weight is such a set.

Note that $n' = O(n^{16})$. Therefore, our algorithm achieves an approximation ratio of $O(\log n)$ for RESTRICTED MINIMUM CONVEX COVER on input polygon T. Because of Theorem 4.2.1, we know that the solution found for RESTRICTED MINIMUM CONVEX COVER is also a solution for the unrestricted MINIMUM CONVEX COVER that is at most a factor of $O(\log n)$ off the optimum solution.

As for the running time of this algorithm, observe that the algorithm adds to the solution in each round a convex polygon with non-zero weight. Therefore, there can be at most $O(n^{16})$ rounds, which yields a total running time of $O(n^{44})$. This completes the proof of the main theorem of this section:

Theorem 4.2.3 MINIMUM CONVEX COVER *for input polygons with or without holes can be approximated by a polynomial time algorithm with an approximation ratio of* $O(\log n)$, *where* n *is the number of polygon vertices.*

[1] Note that by the definition of second-order basic triangles, a second-order basic triangle is either completely contained in M or completely outside M.

4.3 An Inapproximability Result

The upper bound of $O(\log n)$ on the approximation ratio for MINIMUM CONVEX COVER is not tight: We will now prove that there is a constant lower bound on the approximation ratio, and hence a gap remains. More precisely, we show that MINIMUM CONVEX COVER is APX-hard. Our proof of the APX-hardness of MINIMUM CONVEX COVER for input polygons with or without holes uses the construction that is used to prove the NP-hardness of this problem for input polygons without holes[2] [12]. However, we reduce the problem MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY rather than MAXIMUM SATISFIABILITY (as done in the original reduction [12]) to MINIMUM CONVEX COVER, and we design the reduction to be gap-preserving [3]. MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY is APX-complete [3].

The reduction is constructed as follows: For a given instance I of MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY with n variables $x_1, \ldots, x_n$ and m clauses $c_1, \ldots, c_m$ we construct an instance I' of MINIMUM CONVEX COVER. To stick to the notation of [12], let $l_i \leq 5$ denote the number of literals of variable x_i in the clauses, and let $l = \sum_{i=1}^{n} l_i$ be the total number of literals.

For each literal in I, we construct a literal pattern, which we call a "beam machine", as illustrated in Fig. 4.6. A beam machine allows us to send a beam, i.e., a slim convex polygon in one of two possible directions out of the beam machine towards a structure that represents a clause. The beam machines of all literals of a variable are then combined into a variable structureas illustrated in Fig. 4.7. All these variable structures are then arranged in a half circle such that the beams emitted from the beam machines reach the appropriate clause checkers, which are are simple dents. An overview of the whole structure is given in an example in Fig. 4.8. After this overview, let us give a more detailed description.

The beam machine that is constructed for each literal is shown in Fig. 4.6. Since no two of the four vertices a, a', b, and b' see each other, at least four convex polygons are needed to cover the beam machine. Two of these are the maximal convex polygons a, c, d and a', c', d. The remaining areas around the mouth and the ear (the triangle) at b or b' can be covered by a large convex polygon shown in light-gray in Fig. 4.6. Finally, a fourth convex polygon is needed to cover the other ear (at b' in Fig. 4.6). This polygon, which we call a beam, is very slim and can be extended indefinitely beyond the mouth outside the beam machine. The large light-gray convex polygon thus acts as a switch: depending on whether we let it cover the ear at b or b', we can turn on the indefinite beam polygon at the other ear.

[2] APX-hardness for MINIMUM CONVEX COVER for input polygons without holes implies APX-hardness for the same problem for input polygons with holes.

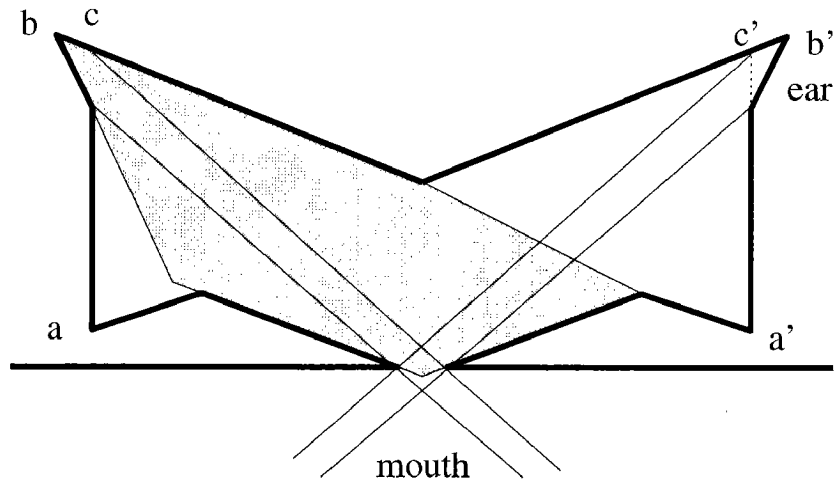


Figure 4.6: The beam machine

However, we cannot turn on both beams and still only use four polygons to cover the beam machine. Note that we can "focus" and "aim" the beam by slightly bending the whole beam machine or by making the ears smaller.

The variable structure is illustrated in Fig. 4.7. Its basic shape is butterfly-like. The beam machines for each occurrence of the variable in a literal in a clause are set on top of the butterfly with the positive literals on the right wing and the negative literals on the left wing of the butterfly. For each literal we have a dent on the bottom line of the wing. If we cover each dent of the left or right wing with a maximal convex polygon, i.e., with a polygon that covers the whole dent and then extends canonically, then we have covered almost all of the left or right wing except for the area around the mouth of the variable structure and except for a small triangular region for each literal that lies between two dents. These triangles are called beam locks. We can cover the beam locks either by beams emanating from the beam machines or by a single large convex polygon which also covers the region around the mouth of the variable structure. Such a polygon is drawn in light-gray in Fig. 4.7. In a similar way as in the beam machine, this large convex polygon acts as a switch: in order to cover the whole variable structure with a minimum number of convex polygons, we can have the beam locks of only one wing covered with such a single polygon; the beam locks of the other wing must be covered by the beams of the beam machines. In Fig. 4.7, beams that are turned on are drawn in dark-gray, while beams that are turned off are medium-gray. Thus, in Fig. 4.7, all beam machines

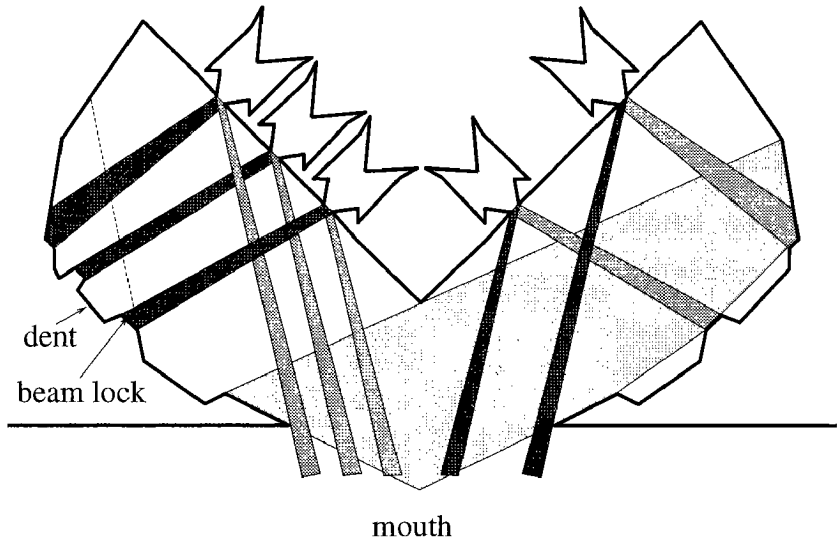


Figure 4.7: The variable structure

of positive literals are turned off and all beam machines of negative literals are turned on and can shine infinitely far beyond the mouth of the variable structure.

We need four convex polygons to cover each beam machine, thus we need $4l_i$ convex polygons to cover the beam machines in the variable structure for variable x_i. For each literal we need an additional polygon to cover the dent, and we need one additional large switcher polygon to cover the mouth and the beam locks of either the positive or negative literals. Thus, a minimum number of $5l_i+1$ convex polygons are required to cover the variable structure of variable x_i. Note that if the beams of only one negative and one positive literal that are both aimed towards and beyond the mouth of the variable structure are turned on, then $5l_i + 2$ convex polygons are needed to cover the variable structure. On the other hand, if the beams of all (positive and negative) literals that cover the beam locks are turned on, there are still $5l_i + 1$ convex polygons needed to cover the variable structure, since we also need to cover the area around the mouth.

We arrange all variable structures in a half-circle-like shape above a base line, which contains triangular dents that represent the clauses as illustrated in Fig. 4.8. This is done in such a way that a beam emanating from a beam machine of a literal that appears in a clause reaches the corresponding dent (the clause checker) that represents that clause and thus covers it. Note

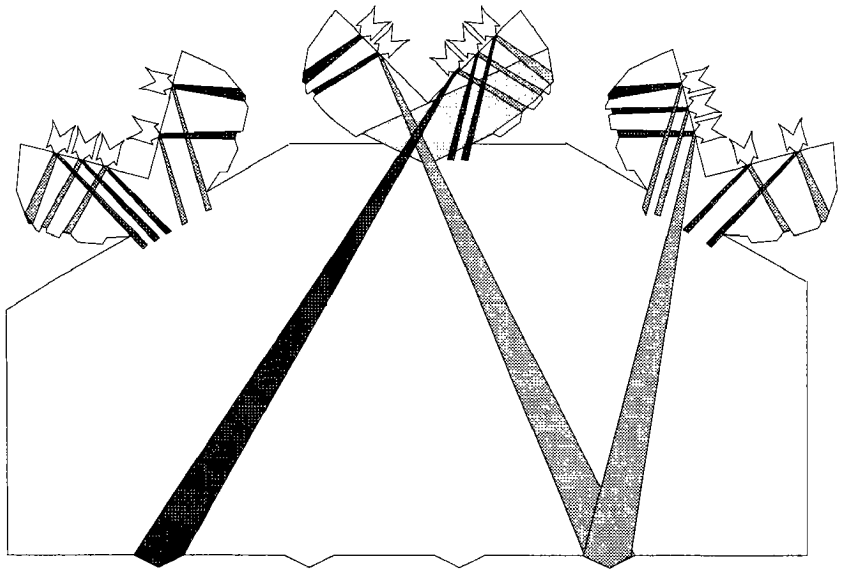


Figure 4.8: Overview of the construction

that we can arrange the variable structures in such a way that they cannot interfere with each other, i.e., no convex polygon can cover any beam locks or areas around the mouth of two different variable structures. We can achieve this by making the angles at the mouth of each variable structure very small.

Theorem 4.3.1 *Let I be an instance of* MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY *consisting of n variables, m clauses with a total of l literals, and let I' be the corresponding instance of* MINIMUM CONVEX COVER. *Let OPT be the maximum number of satisfied clauses of I by any assignment of the variables. Let OPT' be the minimum number of convex polygons needed to cover the polygon of I'. Then:*

$$\begin{aligned} OPT = m &\implies OPT' = 5l + n + 1 \\ OPT < (1 - 15\epsilon)m &\implies OPT' > 5l + n + 1 + \epsilon n \end{aligned}$$

Proof: The first implication is trivial: If we have a variable assignment that satsfies all variables, we turn on the beams that are aimed towards the clause checkers of all beam machines that represent literals that are satisfied by the assignment. We turn on the beams that are aimed towards the beam locks for all other beam machines. Thus, we need $5l_i + 1$ convex polygons to

cover variable structure x_i. If we sum this up over all n variables we obtain $5l + n$ convex polygons, we need one additional polygon to cover the space between the base line and the variable structures.

Since each clause is satisfied, we must have for each clause checker at least one beam turned on that covers it. Thus, the convex polygons as just described cover all of I'.

We prove the second implication by proving its contraposition, i.e., $OPT' \leq 5l + n + 1 + \epsilon n \Longrightarrow OPT \geq (1 - 15\epsilon)m$. To this end, we show how to transform the convex polygons of any solution S' of the MINIMUM CONVEX COVER instance I' in such a way that their total number does not increase and in such a way that a truth assignment of the variables can be "inferred" from the convex polygons that satisfies the desired number of clauses.

Suppose we are given a solution S' of the CONVEX COVER instance with $|S'| \leq 5l + n + 1 + \epsilon n$.

By construction, the variable generator for variable x_i must be covered by at least $5l_i + 1$ convex polygons. Moreover, by construction, there is no convex polygon, which covers a part of a beam lock in any variable generator and a part of a clause checker. There is not even a convex polygon which covers a part of a beam lock and touches the horizontal line, on which the clause checkers lie. Similarly, note that there is no convex polygon which can cover a part of an ear of a beam machine and a part of any clause checker, except for the clause checker associated with the beam machine.

Proceed in the following order:

1. Determine which convex polygon in S' covers the midpoint on the line segment between the clause checkers of clause c_1 and c_2. Transform this polygon in such a way that it covers all of the area between the clause checkers and the variable generators. Note that no convex polygon that covers this midpoint can also cover any beam lock, ear of a beam machine or clause checker. Therefore, this transformation is not "dangerous".

2. For each clause checker proceed as follows: For each convex polygon in S' that covers part of the clause checker and that is not a regular beam, which leads to a beam machine associated with the clause checker, turn the polygon into a beam to anyone of the associated beam machines.

3. If there exists a convex polygon in S' that covers parts of the interior of at least two different variable structure, then choose any variable structure in which it lies and cut off all other parts. This operation is "safe", since, by construction, such a polygon cannot cover the

beam locks or the area around the mouths of two different variable structures.

4. For each variable structure proceed as follows:

   - If the variable structure for x_i is covered by $5l_i+2$ or more convex polygons, then rearrange the convex polygons in such a way that all beams that point to clause checkers are turned on for positive and negative literals. By construction, this is always possible with $5l_i + 2$ convex polygons.
   - If the variable structure for x_i is covered by $5l_i + 1$ convex polygons and one beam from a beam machine for literal x_i ($\neg x_i$) that is aimed at its associated clause checker is turned on, then rearrange all convex polygons in the variable generator in such a way that all beams from beam machines for literal x_i ($\neg x_i$) that are aimed at the associated clause checkers are turned on.

The convex cover obtained this way is still a feasible solution. After this transformation, we have for each variable structure x_i that either for all negative and positive literals, the beams that are aimed towards the clause checkers are turned on, or only for all positive or negative literals, the beams that are aimed towards the clause checkers are turned on, or for negative and positive literals, the beams that are aimed towards the beam locks are turned on.

We set the truth values for the variables as follows: If all beams of literal x_i ($\neg x_i$) that are aimed at clause checkers, and no beams of literal $\neg x_i$ (x_i) that are aimed at clause checkers are turned on, then let the variable x_i have truth value TRUE (FALSE). If either all or no beams (of both literals x_i and $\neg x_i$) that are aimed at clause checkers are turned on, then let variable x_i be TRUE.

By construction, every solution of I' must consist of at least $5l + n + 1$ convex polygons. If we transform a solution of I' with $5l+n+1+\epsilon n$ convex polygons as indicated above, we get at most ϵn variable structures in which the beams of all literals (positive and negative) that are aimed at the clause checkers are turned on. By assigning all this variables the value TRUE, we falsify at most 5 clauses for each variable, since each variable appears at most 5 times as a literal.

Therefore, we get a solution of I with at least $m - 5\epsilon n$ clauses satisfied. Since $3m \geq n$, the solution has at least $m(1 - 15\epsilon)$ satisfied clauses. □

In the promise problem of MAXIMUM 5-OCCURRENCE-3-SATISFIABILITY as described above, we are promised that either all clauses are satisfiable

or at most a fraction of $1 - 15\epsilon$ of the clauses is satisfiable, and we are to find out, which of the two possibilities is true. This problem is NP-hard for small enough values of $\epsilon > 0$ (see Chapt. 1). Therefore, Theorem 4.3.1 implies that the promise problem for MINIMUM CONVEX COVER, where we are promised that the minimum solution contains either $5l + n + 1$ convex polygons or at least $5l + n + 1 + \epsilon n$ convex polygons, is NP-hard as well, for small enough values of $\epsilon > 0$. Therefore, MINIMUM CONVEX COVER cannot be approximated with a ratio of: $\frac{5l+n+1+\epsilon n}{5l+n+1} \geq 1 + \frac{\epsilon n}{25n+n+1} \geq 1 + \frac{\epsilon}{27}$, where we have used that $l \leq 5n$ and $n \geq 1$. This establishes the following:

Theorem 4.3.2 MINIMUM CONVEX COVER *on input polygons with or without holes is APX-hard.*

Chapter 5

Visibility Graphs

5.1 Introduction

In this chapter, we study the class of visibility graphs.

Definition 5.1.1 *A graph $G = (V, E)$ with vertices $v_1, \ldots, v_n$ is a* visibility graph, *iff there exists a simple polygon T (with or without holes) consisting of vertices $p_1, \ldots, p_n$ such that the polygon vertices p_i and p_j see each other, iff $(v_i, v_j) \in E$.*

The *visibility graph characterization* problem consists of finding a set of graph-theoretic properties that exactly define visibility graphs. It is closely related to the *visibility graph recognition* problem, which consists of determining if a given graph is a visibility graph. A lot of work has been done on the visibility graph characterization problem (see [27, 23, 42] or [44] for a survey), but it still is not satisfactorily solved.

A different approach to "characterizing" the class of visibility graphs is to determine the computational complexity (and in case of NP-hardness the approximability) of classic graph-theoretic problems on visibility graphs. This is what is usually done after a new graph class has been discovered or defined. In our particular case, the fact that we cannot reconstruct a polygon, if we are given a visibility graph forces us to distinguish two variations of our approach. We can either assume that we are only given a graph and we know from an oracle that we trust that it is a visibility graph, and we then try to solve some graph-theoretic problem (such as MAXIMUM CLIQUE) for the graph. In a second variation, we are given a graph and a proof that the given graph is a visibility graph, which we can verify in polynomial time. Obviously, this proof should be a polygon. Since we can

easily compute the visibility graph of a given polygon (in time $O(|E|)$ [30]), we might as well assume that we are only given the polygon, and we are to solve some graph-theoretic problem on the visibility graph of this polygon. In what follows, we will always assume the second variation, i.e., we are given a polygon, rather than a graph.[1]

A considerable amount of work has been done that falls in the realm of this approach, because many classic graph-theoretic problems have a geometric interpretation in the context of visibility graphs. We will show in this chapter that most of our results for guarding and hiding problems from previous chapters translate into results for problems on visibility graphs. This even holds to a lesser extent for MINIMUM CONVEX COVER. Another indicator for the interest in this approach is the fact that the problem MINIMUM COLORING ON VISIBILITY GRAPH is mentioned as an open problem (with respect to its computational complexity) in an open problems list [40].

5.2 Guarding or MINIMUM DOMINATING SET

The problem MINIMUM DOMINATING SET ON VISIBILITY GRAPH corresponds to finding a minimum set C of polygon vertices such that each polygon vertex can be seen from at least one vertex in C. This problem is obviously a variation of guarding. We can interpret MINIMUM DOMINATING SET ON VISIBILITY GRAPH as MINIMUM VERTEX RESTRICTED VERTEX GUARD for polygons with or without holes as opposed to MINIMUM BOUNDARY RESTRICTED VERTEX GUARD.

MINIMUM DOMINATING SET ON VISIBILITY GRAPH is a special case of the general MINIMUM DOMINATING SET problem and therefore, it can be approximated by the greedy heuristic that is also used for MINIMUM SET COVER with a logarithmic approximation ratio [32].

It is easy to see that the inapproximability result for MINIMUM VERTEX GUARD WITHOUT HOLES from Sect. 2.2 carries over to MINIMUM DOMINATING SET ON VISBILITY GRAPH, as we have not used the boundary of the input polygon[2], except for the vertices, in order to obtain the

[1] It does matter, which of the two variations we use. There exists, for example, a polynomial time algorithm for MAXIMUM CLIQUE on visibility graphs, but it only works for the second variation (see also [8]).

[2] This is of course only true for MINIMUM BOUNDARY RESTRICTED VERTEX GUARD WITHOUT HOLES. In the corresponding proof for MINIMUM BOUNDARY RESTRICTED POINT GUARD WITHOUT HOLES, which is the only proof presented in detail, we have made use of distinguished arrows, which of course are part of the boundary. This, however, is not necessary for the corresponding vertex guard problem: since guards can only be positioned at vertices, the role of the distinguished arrows can be taken by the vertices that they are connected to.

APX-hardness result. Therefore, we can still prove Theorem 2.2.6. Thus:

Corollary 5.2.1 MINIMUM DOMINATING SET ON VISIBILITY GRAPH FOR POLYGONS WITHOUT HOLES *is APX-hard.*

It is also easy to see that the inapproximability results for MINIMUM VERTEX GUARD WITH HOLES from Sect. 2.3 carries over to MINIMUM DOMINATING SET ON VISBILITY GRAPH for input polygons with holes. We can prove Lemma 2.3.8 without any significant changes. Thus:

Corollary 5.2.2 MINIMUM DOMINATING SET ON VISIBILITY GRAPH FOR POLYGONS WITH HOLES *cannot be approximated by a polynomial time algorithm with an approximation ratio of $\frac{1-\epsilon}{12} \ln n$ for any $\epsilon > 0$, where n is the number of the polygon vertices, unless $NP \subseteq TIME(n^{O(\log \log n)})$.*

5.3 Hiding or MAXIMUM INDEPENDENT SET

We consider the problem MAXIMUM INDEPENDENT SET ON VISIBILITY GRAPH, in which we are given a simple polygon with n vertices and we are to find the maximum independent set in the corresponding visibility graph. This problem corresponds to finding a maximum set of polygon vertices that are hidden from each other. Thus this problem is exactly the problem MAXIMUM HIDDEN VERTEX SET, for which we have shown inapproximability results in Chapter 3. Thus, we can copy the results from that chapter:

MAXIMUM INDEPENDENT SET ON VISIBILITY GRAPH is APX-hard for polygons without holes and NP-hard to approximate with an approximation ratio of $\frac{n^{\frac{1}{6}-\gamma}}{4}$ for all $\gamma > 0$ for polygons with holes.

5.4 MAXIMUM CLIQUE

5.4.1 Introduction

In this section we study the problem MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH(OUT) HOLES, in which we are given a simple polygon with(out) holes with n vertices and we are to find the largest clique in the corresponding visibility graph. Note that in the case of polygons without holes, this problem corresponds to finding a largest (with respect to number of vertices) convex subpolygon of a given polygon. The geometric interpretation in the case of polygons with holes is unclear. This problem has potential applications in the setting up of antenna networks in terrains (see the

previous chapters on guarding and hiding for the relationship of polygons with terrains), where all antennas must see each other in order to guarantee optimum connectivity.

We will show that MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES cannot be approximated by any polynomial time algorithm with an approximation ratio of $\frac{n^{1/8-\epsilon}}{4}$ for any $\epsilon > 0$, unless $NP = P$. Thus, MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES is almost as hard to approximate as clique on general graphs. As usual, we propose a gap-preserving reduction from MAXIMUM CLIQUE on general graphs to get this result.

The problem MAXIMUM CLIQUE ON VISIBILITY GRAPH WITHOUT HOLES is known to be solvable in time $O(n^3)$ by slightly adopting algorithms [5, 14, 15] that were developed to solve different problems (such as finding empty convex polygons that are maximum with respect to the number of vertices by connecting some of the input points). We propose an additional $O(n^3)$ algorithm for this problem for polygons without holes in Sect. 2.3.4, which uses dynamic programming. Our method also solves the weighted version of this problem, in which each vertex is assigned a weight value and the total weight of all vertices in the clique is to be maximized. We will use this weighted version (only with weights 0 and 1) to obtain an approximation algorithm for another visibility graph problem (see Sect. 5.5). Our polynomial algorithm for MAXIMUM CLIQUE ON VISIBILITY GRAPH WITHOUT HOLES bears some ressemblance to the dynamic programming approach to find a maximum convex polygon in the interior of a given polygon as proposed in Chapter 4. However, we are much more concerned about running time in this chapter than in Chapter 4, which is why we present the algorithm in detail.

This gap of "solvable in cubic time" vs. "almost as hard to approximate as clique" is the most extreme gap ever discovered between the two versions of a visibility problem on polygons with vs. without holes.

5.4.2 Finding Cliques in Polygons without Holes

Our polynomial time algorithm for MAXIMUM CLIQUE ON VISIBILITY GRAPH WITHOUT HOLES uses dynamic programming.

Suppose we are given a simple polygon T without holes, which consists of n vertices $v_1, \ldots, v_n$ in counterclockwise order. We first compute the visibility graph $G = G(T)$ of this polygon, which can be done in time $O(|E|)$, where E is the set of edges in G [30]. This allows us to answer queries of the form "Does vertex v_i see vertex v_j ?" in time $O(1)$. As we will use a weighted version of this problem to find an approximation algorithm for

MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPH WITHOUT HOLES, we introduce a non-negative weight w_i for each vertex v_i. We are now to find a clique in G that has a maximum total weight. In the following, all operations are modulo n, where applicable.

Let $A_{i,j,k}$ with $i < j \leq k$ be the *maximum* clique (with respect to its weight) among all cliques, which consist of vertices v_i, v_j and v_k and additional vertices $v_{j'}$ with $i < j' < j$. Let $|A_{i,j,k}|$ denote the weight of $A_{i,j,k}$. The optimum solution OPT is:

$$OPT = A_{i,j,j} \text{ where } i,j \text{ are such that } |A_{i,j,j}| = \max_{1 \leq i < j \leq n} |A_{i,j,j}|$$

Given all $A_{i,j,j}$, OPT can be computed in $O(n^2)$ time. A can be considered to be a three-dimensional table. It is initialized as follows:

$$A_{i,i+1,j} = \{v_i, v_{i+1}, v_j\}, \forall i,j, \text{ where vertices } v_i, v_{i+1}, v_j \text{ all see each other}$$

This initialization can be done in time $O(n^3)$. The remaining entries of the table A are initialized with empty sets and then computed according to Lemma 5.4.1.

Lemma 5.4.1 *Assume vertices v_i, v_j, and v_k see each other. Then, $A_{i,j,k} = A_{i,j',j} \cup v_k$, where j' is such that $|A_{i,j',j}| = \max |A_{i,j'',j}|$, where the maximum is taken over all j'' with $i \leq j'' \leq j$ and where $v_{j''}$ sees v_i, v_j, and v_k.*

Proof: The proof is inductive. Suppose we know that the lemma holds for $A_{i,j,k'}$ with $k' < k$. To show that it also holds for $A_{i,j,k}$, we assume by contradiction that there exists a clique P', which consists of vertices v_i, v_j and v_k and additional vertices $v_{l'}$ with $i < l' < j$ and which is strictly heavier than $A_{i,j,k}$ (as computed in the Lemma). Let v_l be the vertex in P' that is the neighbor of v_j in P' in clockwise order, when we interpret the clique P' as a convex polygon. Now, consider the clique $A_{i,l,j}$, which is maximum by assumption. Because v_j is the neighboring vertex of v_l in P', we have $|P'| \leq |A_{i,l,j}| + w_k$.

We will now argue that vertex v_k can be added to the clique $A_{i,l,j}$ and the resulting set of vertices (i.e. $A_{i,l,j} \cup v_k$) is still a clique. Consider Fig. 5.1. First, note that vertex v_j must lie to the right of the line from v_i to v_k, because vertices v_i, v_j and v_k all see each other and because $i \leq j \leq k$. Since $v_i, v_l, v_j, v_k \in P'$ and $i \leq l \leq j \leq k$ and since P' is a clique, vertex v_l must lie to the right of the line from vertex v_i to v_k and to the left of the line from v_j to v_k. Now, consider all vertices $l'' \in A_{i,l,j}$ that lie between i and l (i.e. $i < l'' < l$). By definition of $A_{i,l,j}$, all these vertices see v_i, v_l and v_j. This implies that all vertices $v_{l''}$ also see v_k, because any polygon segment

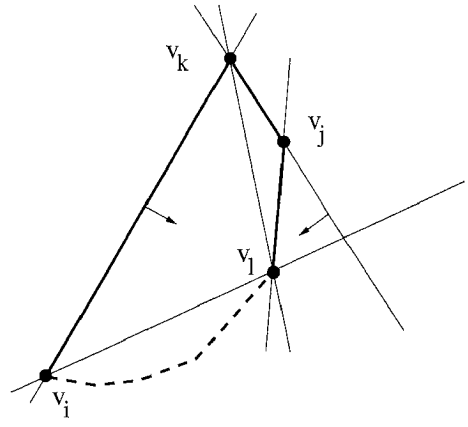


Figure 5.1: Proof of Lemma 5.4.1

blocking the view of some vertex $v_{l'}$ to v_k would imply the existence of a polygon segment that would block the view of $v_{l''}$ to either v_i or v_l. We have shown that all vertices in $A_{i,l,j}$ also see v_k, therefore $A_{i,l,j} \cup v_k$ is a clique as well. The polygon $A_{i,l,k}$ is among those polygons over which the maximum is taken in the Lemma to compute $A_{i,j,k}$. Therefore, $|A_{i,j,k}| \geq P'$, which is a contradiction to the assumption that P' is strictly heavier than $A_{i,j,k}$. □

A trivial implementation of the algorithm thus suggested would have a running time of $O(n)$ for each of the $O(n^3)$ table entries, which results in an overall running time of $O(n^4)$. It is, however, possible to implement the algorithm with a total running time of $O(n^3)$. To achieve this, we show how to compute $A_{i,j,k}$ with i, j fixed and $A_{i,j',j}$ already computed for $i \leq j' \leq j$, in time $O(n)$ (for all k with $j \leq k \leq i$). This directly leads to an $O(n^3)$ algorithm, since there are only $O(n^2)$ pairs i, j.

To speed up the algorithm, fix i, j. Then compute all v_k with $j \leq k \leq i$ that are visible from v_i and v_j. Let K denote the counterclockwise ordered set of all these vertices v_k. Let L denote the clockwise ordered set of vertices v_l with $i \leq l \leq j$ that are visible to both v_i and v_j.

For each vertex $v_l \in L$ (working from v_j towards v_i): Determine, which vertices $v_k \in K$ are visible from v_l. Let $k' < k''$. Note that if v_l sees $v_{k'} \in K$, then it also sees $v_{k''} \in K$. Let $v_{k_{\min}}$ denote the first $v_k \in K$ that sees v_l. It suffices to just "link" $v_{k_{\min}} \in K$ to $A_{i,l,j}$ (depending on the implementation, a "link" could be an entry in some record field or a pointer). Note that as we work our way through L from v_j to v_i, the $v_{k_{\min}}$'s

get smaller, i.e. proceed towards v_j. Thus, determining $v_{k_{\min}}$ can be done in total time $O(|K|)$ for all $v_l \in L$ (if $(|K| > |L|$, otherwise it is $O(|L|))$. We now scan through K. If $v_k \in K$ is "linked" to some $A_{i,j,l}$, we compare the weight of $A_{i,j,l}$ with the weight of the currently optimum solution. If $|A_{i,j,l}|$ is greater than the weight of the currently optimum solution, we update the currently optimum solution to $A_{i,j,l}$. If v_k is not "linked", we link it to the currently optimum solution. Now, set $A_{i,j,k}$ to the currently optimum solution with v_k added. We also store $|A_{i,j,k}|$. This scanning through K can be done in time $O(|K|)$. Thus, the total running time to compute $A_{i,j,k}$ for all k is $O(\max\{|L|, |K|\})$, which is $O(n)$.

Let us summarize the result of this section:

Theorem 5.4.2 *The weighted version of* MAXIMUM CLIQUE ON VISIBILITY GRAPH WITHOUT HOLES, *where non-negative weights are assigned to the vertices, can be solved in time $O(n^3)$ using dynamic programming.*

5.4.3 Finding Cliques in Polygons with Holes

We propose a gap-preserving reduction from the MAXIMUM CLIQUE problem to the MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES problem. Our reduction maps the promise problem of MAXIMUM CLIQUE to the promise problem MAXIMUM CLIQUE ON POLYGONS WITH HOLES. Suppose we are given an instance I of the promise problem MAXIMUM CLIQUE, i.e., a graph $G = (V, E)$ with $n := |V|$ and an integer k with $2 \leq k \leq n$, where $\epsilon > 0$ is arbitrarily small, but fixed. We are promised that the size of a maximum clique in the graph G is either at least k or strictly less than $\frac{k}{n^{1/2-\epsilon}}$. It is NP-hard to decide which of these two cases is true (see Sect. 1.3).

The polygon with holes that we construct is virtually the same as the one that we used to show an inapproximability result for MAXIMUM HIDDEN SET ON POLYGONS WITH HOLES in Sect. 3.2. The basic idea of the reduction is shown in Figs. 3.4 and 3.5. For each instance I of MAXIMUM CLIQUE, i.e., for each graph $G = (V, E)$ with $n := |V|$ (as shown in an example in Fig. 3.4), we construct an instance I' of MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES, i.e., a polygon with holes (as shown in an example in Fig. 3.5). We shortly repeat the main steps of the construction: The main polygon is in the shape of a regular $2n$-gon with vertices named v_i and v'_i for $i \in \{1, \ldots, n\}$. For each vertex pair $(v_i, v_j) \notin E$, we construct two small triangular holes, one around the intersection point of the line segment from v_i to v_j and the line segment from v'_i to v'_{i+1}, and one around the intersection point of the line segment from v_i to v_j and the line segment from v'_j to v'_{j+1}. These triangular holes are designed to block

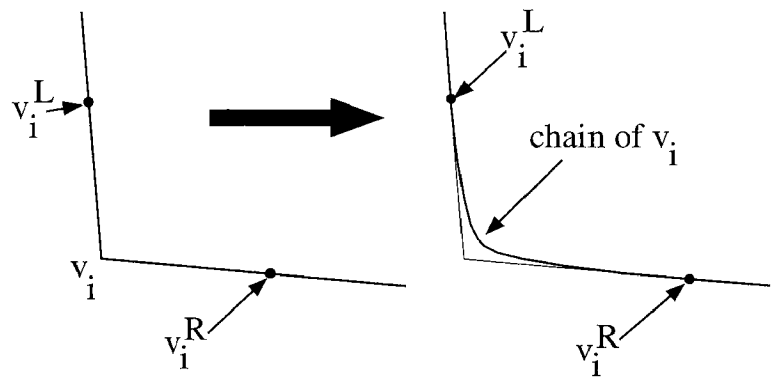


Figure 5.2: Chain of vertex v_i

the view of vertices v_i and v_j that are not supposed to see each other, since they are not connected by an edge in the input graph. See Sect. 3.2 for more details.

In order to make the reduction work, we refine the polygon with holes obtained thus far as follows:

For each i, let $v_i^L := v_{i,0}$ and $v_i^R := v_{i,n^2+1}$. See Sect. 3.2 for the definition of these points. For each vertex v_i, we replace the two line segments from v_i^L to v_i to v_i^R by a convex chain of of $n^3 - 1$ line segments (called the *chain of* v_i). This is illustrated in Fig. 5.2. By the way that we chose points v_i^L and v_i^R, it is ensured that any two vertices from chains of v_i and v_j see each other, iff $(v_i, v_j) \in E$.

The following two lemmas allow us to prove the main result of this section. Let OPT denote the size of an optimum solution of the MAXIMUM CLIQUE instance I and let OPT' denote the size of an optimum solution of the MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES instance I'. Let $\epsilon > 0$.

Lemma 5.4.3 $OPT \geq k \implies OPT' \geq n^3 k$

Proof: If $OPT \geq k$, then there exists a clique of size k in I. We obtain a clique in I' of size $n^3 k$ by simply letting all the n^3 vertices of the chain of v_i be in the solution, if vertex $v_i \in V$ is in the clique.

□

Lemma 5.4.4 $OPT < \frac{k}{n^{1/2-\epsilon}} \implies OPT' < \frac{n^3 k}{n^{1/2-\epsilon}} + 3n^2$

Proof: We prove the contraposition: $OPT' \geq \frac{n^3k}{n^{1/2-\epsilon}} + 3n^2 \Longrightarrow OPT \geq \frac{k}{n^{1/2-\epsilon}}$. Suppose we have a solution of I' with $\frac{n^3k}{n^{1/2-\epsilon}} + 3n^2$ points. Since there are at most $n(n-1)$ holes with 3 vertices each and n additional vertices v_i', there can be at most $3n(n-1) + n \leq 3n^2$ vertices in the clique that are not part of the chain of some v_i. Therefore, at least $\frac{n^3k}{n^{1/2-\epsilon}}$ vertices of the clique must be in chains. Since a chain consists of only n^3 vertices, each chain can contribute at most n^3 vertices to the clique. Therefore, the number of chains that contain at least one point from the solution is at least $\frac{\frac{n^3k}{n^{1/2-\epsilon}}}{n^3} = \frac{k}{n^{1/2-\epsilon}}$. Since no two vertices of two different chains v_i and v_j see each other unless $(v_i, v_j) \in E$, we immediately have a solution for I with at least $\frac{k}{n^{1-\epsilon}}$ vertices by letting v_i be in the clique if at least one point of the chain of v_i is in the solution.

□

Lemmas 5.4.3 and 5.4.4 transform the promise problem of MAXIMUM CLIQUE as mentioned above into a promise problem of MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES, where we are promised that an optimum solution contains either at least n^3k vertices or strictly less than $\frac{n^3k}{n^{1/2-\epsilon}} + 3n^2$ vertices. It is also NP-hard to decide, which of the two cases is true, since otherwise, we could solve the NP-hard promise problem of MAXIMUM CLIQUE. MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES can therefore not be approximated by any polynomial time approximation algorithm with an approximation ratio of:

$$\frac{n^3k}{\frac{n^3k}{n^{1/2-\epsilon}} + 3n^2} = \frac{n^3k}{\frac{n^3k+3n^{1-2\epsilon}}{n^{1/2-\epsilon}}} \geq \frac{n^3k}{\frac{2n^3k}{n^{1/2-\epsilon}}} = \frac{n^{1/2-\epsilon}}{2}$$

We now need to express the size $|I'|$ of the MAXIMUM CLIQUE ON VISIBILITY GRAPH WITH HOLES instance I' by the size n of the MAXIMUM CLIQUE instance I. According to the construction, $|I'| \geq 2n^4$. We proceed:

$$\frac{n^{1/2-\epsilon}}{2} \geq \frac{\frac{|I'|^{\frac{1}{4}(\frac{1}{2}-\epsilon)}}{2^{\frac{1}{4}(\frac{1}{2}-\epsilon)}}}{2} \geq \frac{|I'|^{\frac{1}{8}-\frac{\epsilon}{4}}}{4}$$

This completes the proof of our main theorem of this section:

Theorem 5.4.5 MAXIMUM CLIQUE ON VISIBILITY GRAPHS WITH HOLES *cannot be approximated by any polynomial time algorithm with an approximation ratio of* $\frac{|I'|^{1/8-\gamma}}{4}$, *where* $|I'|$ *is the number of vertices in the polygon and where* $\gamma > 0$, *unless* $NP = P$.

5.5 Minimum Clique Partition

5.5.1 Introduction

The problem Minimum Clique Partition consists of finding a partitioning of the vertices of a given graph into a minimum number of disjoint vertex sets, each of which must be a clique in the graph. Again, we can define this problem on visibility graphs of polygons with or without holes. In the case of polygon without holes, this problem is closely related to Minimum Convex Cover without Holes. Minimum Clique Partition on Visibility Graphs without Holes is a variant of Minimum Convex Cover without Holes, where only the vertices are of interest (not the edges or the interior area of the polygon).

Again, it is easy to see that the APX-hardness result for Minimum Convex Cover presented in Chapter 5 carries over to Minimum Clique Partition on Visibility Graphs without Holes.

In this chapter, we propose an approximation algorithm for Minimum Clique Partition on Visibility Graphs without Holes that iteratively applies the algorithm for the weighted version of Maximum Clique on Visibility Graph without Holes and show that it achieves a logarithmic approximation ratio. Of course, this approach is very similar to the approximation algorithm for Minimum Convex Cover presented in Chapter 4. This result sheds some light on the approximability of Minimum Clique Partition on Visibility Graphs without Holes, but it still is not known whether a constant approximation ratio can be achieved or whether the logarithmic approximation algorithm presented is optimum. There seems to be no straight-forward geometric interpretation of Maximum Clique Partition on Visibility Graph with Holes, but the problem is certainly of theoretic interest, as we propose a gap-preserving reduction in this section from Maximum Clique Partition on general graphs that shows that Maximum Clique Partition on Visibility Graph with Holes cannot be approximated with an approximation ratio of $\frac{n^{1/14-\gamma}}{4}$ for any $\gamma > 0$.

This is the first result for a visibility problem that is NP-hard no matter whether holes are allowed or not, where we are able to show that the approximation properties are clearly different for the cases of polygons with vs. without holes: While Maximum Clique Partition on Visibility Graph with Holes cannot be approximated with an approximation ratio of $\frac{n^{1/14-\gamma}}{4}$ for any $\gamma > 0$, we have a logarithmic approximation algorithm for Minimum Clique Partition on Visibility Graphs without Holes.

5.5.2 Finding Clique Partitions in Polygons without Holes

Our approximation algorithm for MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPH WITHOUT HOLES iteratively applies the polynomial time algorithm for the weighted version of MAXIMUM CLIQUE ON VISIBILITY GRAPH WITHOUT HOLES. It works as follows for a given polygon T:

1. Compute the visibility graph $G(T)$ of the polygon T. Let all vertices have weight 1.
2. Find the maximum weighted clique C in $G(T)$ using the algorithm proposed in Sect. 2.3.4. Let all vertices $v_i \in C$ have weight 0. Add C to the solution S.
3. Repeat step 2 until there are no vertices with weight 1 left. Return S.

To obtain a performance guarantee of this algorithm, consider the MINIMUM SET COVER instance I, which has all polygon vertices v_i as elements and the vertices of each clique in the visibility graph of the polygon are a set in I. The greedy heuristic for MINIMUM SET COVER, which consists of recursively adding to the solution a set, which contains a maximum number of elements not yet covered by the solution, achieves an approximation ratio of $1 + \ln n$, where n is the number of elements in I [31].

Our algorithm works in exactly this way. Note that we do not have to compute all the sets of the MINIMUM SET COVER instance I (which would possibly be a number exponential in n), since it suffices to always compute a set (or clique), which contains a maximum number of vertices not yet covered by the solution, which is achieved by reducing the weights of the vertices already in the solution to 0. Thus, our algorithm is polynomial.

Theorem 5.5.1 MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPH WITHOUT HOLES *can be approximated with an approximation ratio of* $O(\log n)$, *where* n *is the number of polygon vertices, by a greedy heuristic.*

5.5.3 Finding Clique Partitions in Polygons with Holes

MINIMUM CLIQUE PARTITION on general graphs is equivalent to MINIMUM GRAPH COLORING [10]. It cannot be approximated by any polynomial time algorithm with an approximation ratio of $n^{1/7-\epsilon}$, where $\epsilon > 0$ and n is the number of vertices in the graph [10]. We propose a gap-preserving reduction from MINIMUM CLIQUE PARTITION on general graphs to MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPH WITH HOLES. Again, we map the NP-hard promise problem of MINIMUM CLIQUE PARTITION on general graphs,

where we are promised that an optimum solution consists of either at most k or strictly more than $n^{1/7-\epsilon}k$ cliques, to a promise problem of MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPH WITH HOLES, where we are promised that an optimum solution consists of either at most $k+3$ or strictly more than $n^{1/7-\epsilon}k$ cliques.

We use the same construction as used in Sect. 5.4.3. However, we do not need to use the "chains" as introduced in Sect. 5.4.3. Let OPT (OPT') denote the size of an optimum solution of the MAXIMUM CLIQUE PARTITION (MAXIMUM CLIQUE PARTITION ON VISIBILITY GRAPH WITH HOLES) instance I (I'). Let $\epsilon > 0$.

Lemma 5.5.2 *$OPT \leq k \implies OPT' \leq k+3$ and $OPT > n^{1/7-\epsilon}k \implies OPT' > n^{1/7-\epsilon}k$*

Proof: For the first implication: If $OPT \leq k$, then there exists a solution of size k in I. We obtain a solution in I' of size $k+3$ by simply letting all cliques from the solution in I be cliques in I' and by adding three more cliques. One of these consists of all the "bottom" vertices of all holes (i.e. those vertices that lie on line segments between points v'_{i-1} and v'_i for all i). The holes are constructed in such a way that these vertices actually form a clique (see Chapter 3). The second clique consists of the "top" vertices of all holes. The third clique consists of all vertices v'_i. The construction of the reduction ensures that these additional cliques actually are cliques. We prove the contraposition of the second implication: A solution for I' can be interpreted as a solution for I, where the additional vertices of I' are ignored. □

We now proceed as in Sect. 5.4.3 using the same concepts. Lemma 5.5.2 and the fact that $|I'| \geq 3n^2$ allow us to prove:

Theorem 5.5.3 MAXIMUM CLIQUE PARTITION ON VISIBILITY GRAPH WITH HOLES *cannot be approximated by any polynomial time algorithm with an approximation ratio of $\frac{|I'|^{1/14-\gamma}}{4}$, where $|I'|$ is the number of vertices in the polygon and where $\gamma > 0$, unless $NP = P$.*

5.6 HAMILTON CIRCLES in Visibility Graphs

A graph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$ is said to contain a *Hamilton circle*, if there exists a circle in G starting and ending w.l.o.g. at vertex v_1 and visiting each vertex in V exactly once.

Trivially, all visibility graphs of polygons without holes have a Hamilton circle: in a polygon T, given by a (say, clockwise) ordered list of vertices $\{v_1, \ldots, v_n\}$, this ordered list also describes a Hamilton circle.

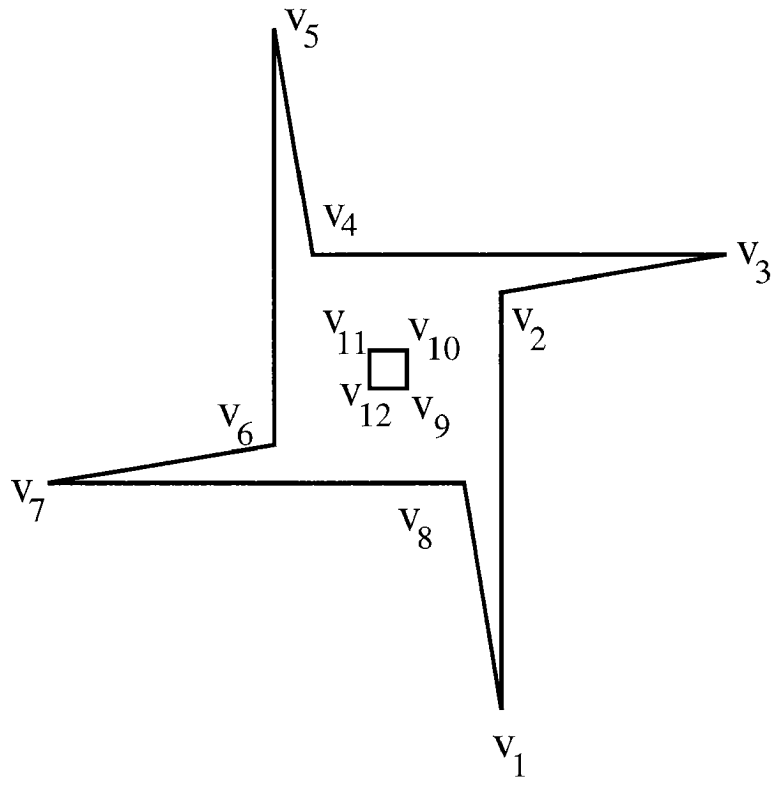


Figure 5.3: A polygon with holes and without a Hamilton circle

Visibility graphs of polygons with holes, on the other hand, do not necessarily contain a Hamilton circle as the polygon in Fig. 5.3 demonstrates [47]: vertices v_1, v_3, v_5, v_7 only see their neighboring vertices. Therefore, a potential Hamilton circle would start at vertex v_1, then w.l.o.g. continue to v_2. Then it would have to continue to v_3, which would be cut off otherwise. The potential Hamilton circle would thus continue along the boundary of the outer polygon until it reaches vertex v_8, from which it has to go back to v_1 in order not to cut off v_1. The vertices $v_9, \ldots, v_{12}$ of the holes are not reached in this potential Hamilton circle.

Chapter 6

Conclusion

We can look at almost all of the results in this thesis in the context of approximability classes as introduced in Chapt. 1. The following tables summarize the results, where each table covers one of the main chapters, i.e., guarding, hiding, convex covering, and visibility graphs. The tables show the best approximation ratios that known approximation algorithms achieve, as well as the strongest inapproximability results known for the problem. The sources of the results are also indicated: they either point to a section of this thesis or to another publication.

Guarding		
Problem:	Best inapproximability Result:	Best approximation algorithm:
MINIMUM FIXED HEIGHT GUARD ON 1.5D TERRAIN	–	$\in P$, 2.1, [37]
MINIMUM (BOUNDARY RESTRICTED) VERTEX GUARD ON POLYGONS WITHOUT HOLES	APX-hard, 2.2	$\in$ "$\log n$", [26]
MINIMUM (BOUNDARY RESTRICTED) EDGE GUARD ON POLYGONS WITHOUT HOLES	APX-hard, 2.2	$\in$ "$\log n$", [26]
MINIMUM (BOUNDARY RESTRICTED) POINT GUARD ON POLYGONS WITHOUT HOLES	APX-hard, 2.2	$\in$ "$\log n$", [26]
MINIMUM (BOUNDARY RESTRICTED) VERTEX GUARD ON POLYGONS WITH HOLES	"$\log n$"-hard, 2.3	$\in$ "$\log n$", [26]
MINIMUM (BOUNDARY RESTRICTED) EDGE GUARD ON POLYGONS WITH HOLES	"$\log n$"-hard, 2.3	$\in$ "$\log n$", [26]
MINIMUM (BOUNDARY RESTRICTED) POINT GUARD ON POLYGONS WITH HOLES	"$\log n$"-hard, 2.3	$\in$ "n^{ϵ}", –

MINIMUM VERTEX GUARD ON TERRAIN	"$\log n$"-hard, 2.4	$\in$ "$\log n$", 2.5
MINIMUM VERTEX GUARD ON TERRAIN WITH TRIANGLE RESTRICTION	"$\log n$"-hard, 2.4	$\in$ "$\log n$", 2.5
MINIMUM POINT GUARD ON TERRAIN	"$\log n$"-hard, 2.4	$\in$ "n^{ϵ}", 2.5
MINIMUM POINT GUARD ON TERRAIN WITH TRIANGLE RESTRICTION	"$\log n$"-hard, 2.4	$\in$ "$\log n$", 2.5
MINIMUM FIXED HEIGHT GUARD ON TERRAIN	"$\log n$"-hard, 2.4	$\in$ "n^{ϵ}", 2.5
MINIMUM FIXED HEIGHT GUARD ON TERRAIN WITH TRIANGLE RESTRICTION	"$\log n$"-hard, 2.4	$\in$ "$\log n$", 2.5

Thus, for some guarding problems, a gap between the best inapproximability result and the best approximation algorithm remains. Most notably, it is not clear whether MINIMUM VERTEX GUARD ON POLYGON WITHOUT HOLES is in APX or whether it is "$\log n$"-complete. This is an important and certainly interesting, open problem.

For some point guard variations as well as for unrestricted fixed height guarding problems, the approach of Sect. 2.5 of discretizing the space of possible guard positions in order to obtain an approximation algorithm does not work. It is therefore an interesting open problem to find approximation algorithms for these problems that achieve non-trivial approximation ratios.

Hiding		
Problem:	Best inapproximability Result:	Best approximation algorithm:
MAXIMUM HIDDEN (VERTEX) SET ON POLYGON WITHOUT HOLES	APX-hard, 3.1	$\in$ "n^ϵ", –
MAXIMUM HIDDEN (VERTEX) SET ON POLYGON WITH HOLES	"n^ϵ"-hard, 3.2	$\in$ "n^ϵ", –
MAXIMUM HIDDEN (VERTEX) SET ON TERRAIN	"n^ϵ"-hard, 3.3	$\in$ "n^ϵ", –

The approximability of hiding problems is settled for input polygons with holes and for terrains, i.e., they are "n^ϵ"-complete. The corresponding algorithms that achieve such ratios are all trivial: an algorithm that simply returns a single vertex always achieves a ratio of n.

However, the situation is not settled for input polygons without holes. Here, a large gap remains.

Convex Covering		
Problem:	Best inapproximability Result:	Best approximation algorithm:
MINIMUM CONVEX COVER ON POLYGON WITH(OUT) HOLES	APX-hard, 4.3	$\in$ "$\log n$", 4.2

Thus, for MINIMUM CONVEX COVER a gap remains: It is not clear, if this problem is in APX or "$\log n$"-complete. This is an open problem for future research. Furthermore, it is not clear whether allowing holes in the input polygons makes a difference.

Visibility Graphs		
Problem:	Best inapproximability result:	Best approximation algorithm:
MINIMUM DOMINATING SET ON VISIBILITY GRAPH FOR POLYGON WITHOUT HOLES	APX-hard, 5.2	$\in$ "$\log n$", [31]
MINIMUM DOMINATING SET ON VISIBILITY GRAPH FOR POLYGON WITH HOLES	"$\log n$"-hard, 5.2	$\in$ "$\log n$", [31]
MAXIMUM INDEPENDENT SET ON VISIBILITY GRAPH FOR POLYGON WITHOUT HOLES	APX-hard, 5.3	$\in$ "n^ϵ", –
MAXIMUM INDEPENDENT SET ON VISIBILITY GRAPH FOR POLYGON WITH HOLES	"n^ϵ"-hard, 5.3	$\in$ "n^ϵ", –
MAXIMUM CLIQUE ON VISIBILITY GRAPH FOR POLYGON WITHOUT HOLES	–	$\in P$, 5.4, [14]
MAXIMUM CLIQUE ON VISIBILITY GRAPH FOR POLYGON WITH HOLES	"n^ϵ"-hard, 5.4	$\in$ "n^ϵ", –
MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPH FOR POLYGON WITHOUT HOLES	APX-hard, 5.5	$\in$ "$\log n$", 5.5
MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPH FOR POLYGON WITH HOLES	"n^ϵ"-hard, 5.5	$\in$ "n^ϵ", –

As a last result, we have from Sect. 5.6 that visibility graphs of polygons without holes always contain a Hamilton circle, while some visibility graphs of polygons with holes do not contain a Hamilton circle.

Thus, for problems restricted to visibility graphs, some gaps in the inapproximability remain. However, the problems MAXIMUM CLIQUE ON VIS-

IBILITY GRAPH and MINIMUM CLIQUE PARTITION ON VISIBILITY GRAPH clearly show that allowing holes in the input polygons does make a difference.

To conclude, let us note that approximation algorithms are very rare for geometric problems in general (see comments in [31]); they are particularly rare for visibility problems (as mentioned in [48]). The same holds to an even greater extent for inapproximability results. It is the sincere hope of the author that this thesis will be a starting point to more intense research in the field of approximability and inapproximability of geometric problems in general and visibility problems in particular. As can be read from the tables above, an interesting set of problems remains open: a challenge for future research.

Bibliography

[1] A. Aggarwal, S. Ghosh, and R. Shyamasundar; *Computational Complexity of Restricted Polygon Decompositions*; Computational Morphology, G. Toussaint, ed., North-Holland, pp. 1 – 11, 1988.

[2] S. Arora; *Probabilistic Checking of Proofs and the Hardness of Approximation Problems*; PhD thesis, U.C. Berkeley, 1994.

[3] S. Arora and C. Lund; *Hardness of Approximations*; in: Approximation Algorithms for NP-Hard Problems (ed. Dorit Hochbaum), PWS Publishing Company, pp. 399 – 446, 1996.

[4] S. Arora and M. Sudan; *Improved low-degree testing and its applications*; Proc. 29th ACM Symposium on the Theory of Computing, pp. 485 – 495, 1997.

[5] D. Avis and D. Rappaport; *Computing the largest empty convex subset of a set of points*; Proc. 1st Ann. ACM Symposium Computational Geometry, pp. 161 – 167, 1985.

[6] P. Bose, T. Shermer, G. Toussaint, and B. Zhu; *Guarding Polyhedral Terrains*; Computational Geometry 7, Elsevier Science B. V., pp. 173 – 185, 1997.

[7] D. P. Bovet and P. Crescenzi; *Introduction to the theory of complexity*; Prentice Hall D. P. Bovet and P. Crescenzi, 1993.

[8] A. Brandstädt, V.B. Le, and J.P. Spinrad; *Graph Classes – a Survey*; SIAM Monographs on Discrete Mathematics and Applications, 1999.

[9] B. Chazelle and D.P. Dobkin; *Optimal Convex Decompositions*; Computational Geometry, C.T. Toussaint (editor), Elsevier Science B. V. (North Holland), pp. 63 – 133, 1985.

[10] P. Crescenzi and V. Kann; *A Compendium of NP Optimization Problems*; in the book by G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, M. Protasi, *Complexity and Approximation. Combinatorial Optimization Problems and their Approximability Properties*, Springer-Verlag, Berlin, 1999; also available in an online-version at http://www.nada.kth.se/theory/compendium/.

[11] P. Crescenzi and L. Trevisan; *On approximation scheme preserving reducibility and its applications*, Proc. 14th Ann. Conf. on Foundations of Software Tech. and Theoret. Comput. Sci., Lecture Notes in Comput. Sci. 880, Springer-Verlag, pp. 330 – 341, 1994.

[12] J. C. Culberson and R. A. Reckhow; *Covering Polygons is hard*; Proc. 29th Symposium on Foundations of Computer Science, pp. 601 – 611, 1988.

[13] L. De Floriani and P. Magillo; *Visibility Algorithms on Triangulated Terrain Models*; International Journal of Geographic Information Systems, 8(1), pp. 13 – 41, Taylor & Francis, London, 1994.

[14] David P. Dobkin, Herbert Edelsbrunner, and Mark H. Overmars; *Searching for Empty Convex Polygons*; Algorithmica, 5, pp. 561 – 571, 1990.

[15] Herbert Edelsbrunner, L. Guibas; *Topologically sweeping an arrangement*; J. Comput. System Sci. 38, pp. 165 – 194, 1989.

[16] S. Eidenbenz and C. Stamm; *Maximum Clique and Minimum Clique Partition in Visibility Graphs*; Lecture Notes in Computer Science (IFIP TCS'2000), 2000.

[17] S. Eidenbenz and P. Widmayer; *An Approximation Algorithm for Minimum Convex Cover with Logarithmic Performance Guarantee*; manuscript, submitted for publication, 2000.

[18] S. Eidenbenz, C. Stamm, and P. Widmayer; *Inapproximability Results for Guarding Polygons and Terrains*; accepted for publication in Algorithmica, 1999.

[19] S. Eidenbenz; *How Many People Can Hide in a Terrain?*; Lecture Notes in Computer Science 1741 (ISAAC'99), pp. 184 – 194, 1999.

[20] S. Eidenbenz, C. Stamm, and P. Widmayer; *Inapproximability of some Art Gallery Problems*; Proc. 10th Canadian Conf. Computational Geometry (CCCG'98), pp. 64 – 65, 1998.

[21] S. Eidenbenz, C. Stamm, and P. Widmayer; *Positioning Guards at Fixed Height above a Terrain – An Optimum Inapproximability Result*; Lecture Notes in Computer Science, Vol. 1461 (ESA'98), p. 187 – 198, 1998.

[22] S. Eidenbenz; *Inapproximability Results for Guarding Polygons without Holes*; Lecture Notes in Computer Science, Vol. 1533 (ISAAC'98), p. 427 – 436, 1998.

[23] H. Everett, D. Corneil; *Negative Results on Characterizing Visibility Graphs*; Computational Geometry 5, pp. 51 – 63, 1995.

[24] U. Feige; *A threshold of* $\ln n$ *for Approximating Set Cover*; Journal of the ACM, Vol. 45 No. 4, pp. 634 – 652, 1998; a preliminary version appeared in Proc. 28th ACM Symposium on the Theory of Computing, pp. 314 – 318, 1996.

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

[26] S. Ghosh; *Approximation algorithms for Art Gallery Problems*; Proc. of the Canadian Information Processing Society Congress, pp. 429 – 434, 1987.

[27] S. Ghosh; *On Recognizing and Characterizing Visibility Graphs of Simple Polygons*; Discrete Comput Geom 17, pp. 143 – 162, 1997.

[28] J. Hastad; *Some optimal inapproximability results*; Proceedings of the 29th Annual ACM Symposium on Theory of Computing, pp. 1 – 10, 1997.

[29] J. Hastad; *Clique is hard to approximate within* $n^{1-\epsilon}$; Proc. of the 37th Symposium on Foundations of Computer Science, IEEE, pp. 627 – 637 1996.

[30] J. Hershberger; *Finding the Visibility Graph of a Polygon in Time Proportional to its Size*; Proc. of the 3rd Annual ACM Symposium on Computational Geometry, Waterloo, pp. 11 – 20, 1987.

[31] D. Hochbaum; *Approximating Covering and Packing Problems: Set Cover, Vertex Cover, Independent Set, and Related Problems*; in: Approximation Algorithms for NP-Hard Problems (ed. Dorit Hochbaum), PWS Publishing Company, pp. 94 – 143, 1996.

[32] D. Johnson; *Approximation Algorithms for Combinatorial Problems*; Journal Comput. System Sci. 9, pp. 256-278, 1974.

[33] M. van Kreveld; *Digital Elevation Models and TIN Algorithms*; in Algorithmic Foundations of Geographic Information Systems (ed. van Kreveld et al.), LNCS tutorial vol. 1340, pp. 37 – 78, Springer, 1997.

[34] V.S.A. Kumar and H. Ramesh; *Covering Rectilinear Polygons with Axis-parallel Rectangles*; Proc. of the 31st Annual ACM Symposium on Theory of Computing, pp. 445 – 454, ACM Press, 1999.

[35] D. T. Lee and A. K. Lin; *Computational Complexity of Art Gallery Problems*; IEEE Trans. Info. Th, pp. 276-282, IT-32, 1986.

[36] A. Lingas; *The Power of Non-Rectilinear Holes*; Proc. 9th COlloquium on Automata, Languages, and Programming, pp. 445 – 454, 1999.

[37] B. Nilsson; *Guarding Art Galleries - Methods for Mobile Guards*; PhD Thesis, Department of Computer Science, Lund University, 1994.

[38] J. O'Rourke and K. J. Supowit; *Some NP-hard Polygon Decomposition Problems*; IEEE Transactions on Information Theory, Vol IT-29, No. 2, 1983.

[39] J. O'Rourke; *Art Gallery Theorems and Algorithms*; Oxford University Press, New York, 1987.

[40] J. O'Rourke; *Open Problems in the Combinatorics of Visibility and Illumination*; in Advances in Discrete and Computational Geometry, eds. B. Chazelle and J.E. Goodman and R. Pollack, (Contemporary Mathematics) American Mathematical Society, Providence, pp. 237 – 243, 1998.

[41] C.H. Papadimitriou and M. Yannakakis; *Optimization, approximation, and complexity classes*; Proc. 20th ACM Symposium on the Theory of Computing, pp. 229 – 234, 1988.

[42] L. Prasad, S.S. Iyengar; *A note on the combinatorial structure of the visibility graph in simple polygons*; Theoretical Computer Science 140, pp. 249 – 263, 1995.

[43] R. Raz and S. Safra; *A Sub-Constant Error-Probability Low-Degree Test, and a Sub-Constant Error-Probability PCP Characterization of NP*; Proc. 29th ACM Symposium on the Theory of Computing, pp. 475 – 484, 1997.

[44] T. Shermer; *Recent results in Art Galleries*; Proc. of the IEEE, 1992.

[45] T. Shermer; *Hiding People in Polygons*; Computing 42, pp. 109 – 131, 1989.

[46] L. Trevisan; *Reductions and (Non-)Approximability*, PhD thesis, University of Rome "La Sapienza", 1997.

[47] R. Ulber and S. Eidenbenz; *Hamilton Circles in Visibility Graphs*; private communication, 1999.

[48] J. Urrutia; *Art gallery and Illumination Problems*; in Handbook on Computational Geometry, edited by J.-R. Sack and J. Urrutia, 1998.

[49] B. Zhu; *Computing the Shortest Watchtower of a Polyhedral Terrain in* $O(n \log n)$ *Time*; in Computational Geometry 8, pp. 181 – 193, Elsevier Science B.V., 1997.

Curriculum Vitae

Stephan J. Eidenbenz

date of birth:	August 25, 1973
place of birth:	Zurich, Switzerland
citizenship:	Switzerland

Education:

1997 - 2000:	PhD student at Institute for Theoretical Computer Science ETH Zurich, Switzerland advisor: Prof. Dr. Peter Widmayer academic title: Dr. sc. tech.
1998 - 1999:	Business Administration studies at GSBA/Oekreal Zuerich academic title: B.B.A. GSBA/Oekreal
1993 - 1997:	studies at ETH Zurich major: Computer Science minor: Operations Research academic title: M. Sc. (Computer Science), ETH
1986 - 1993:	high school in Davos, Switzerland degree: Matura (Swiss high school diploma)

1990 - 1991:	senior year at Tustin high school in Tustin, California
	degree: high school diploma
1980 - 1986:	primary school in Davos, Switzerland

Work experience:

Summer 1999:	organisation of the 25th Workshop on Graph-theoretic Concepts in Computer Science WG'99
Summer 1995/ Spring 1996:	three months design and programming work at Spectrospin AG, Faellanden, Switzerland
Summer/Fall 1993:	two months programming work (PASCAL) at AO Institute, Davos, Switzerland

Publications:

- Maximum Clique and Minimum Clique Partition in Visibility Graphs
  S. Eidenbenz
  Lecture Notes in Computer Science (IFIP TCS2000), 2000
- An Approximation Algorithm for Minimum Convex Cover with Logarithmic Performance Guarantee
  S. Eidenbenz, P. Widmayer
  manuscript, submitted for publication, 2000
- Inapproximability Results for Guarding Polygons and Terrains
  S. Eidenbenz, C. Stamm, P. Widmayer
  accepted for publication in Algorithmica, 1999
- Graph Theoretic Concepts in Computer Science, WG'99
  Peter Widmayer, Gabriele Neyer, Stephan Eidenbenz (Eds.)
  Lecture Notes in Computer Science 1665, 1999
- How Many People Can Hide in a Terrain?
  S. Eidenbenz
  Lecture Notes in Computer Science 1741 (ISAAC'99), pp. 184 - 194, 1999
- Inapproximability Results for Guarding Polygons without Holes
  S. Eidenbenz

Lecture Notes in Computer Science 1533 (ISAAC'98), pp. 427 - 436, 1998

- Inapproximability of Some Art Gallery Problems
  S. Eidenbenz, C. Stamm, P. Widmayer
  Proceedings 10th Canadian Conference on Computational Geometry, CCCG, 1998

- A Modified Longest Side Bisection Triangulation
  S. Eidenbenz, C. Stamm
  Proceedings 10th Canadian Conference on Computational Geometry, CCCG, 1998

- Positioning Guards at Fixed Height above a Terrain – an Optimum Inapproximability Result
  S. Eidenbenz, C. Stamm, P. Widmayer
  Lecture Notes in Computer Science 1461 (ESA'98), 187-198, 1998

- A Prototype System for Light Propagation in Terrains
  M. Beck, S. Eidenbenz, C. Stamm, P. Stucki, P. Widmayer
  Invited Contribution to CGI, 1998