Lower Bounds for Fundamental Geometric Problems

by

Jeffrey Gordon Erickson

B.A. (Rice University) 1987
M.S. (University of California at Irvine) 1992

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION
of the
UNIVERSITY of CALIFORNIA at BERKELEY

Committee in charge:

Professor Raimund Seidel, Chair
Professor Umesh Vazirani
Professor Bernd Sturmfels

Fall 1996

The dissertation of Jeffrey Gordon Erickson is approved:

Chair Date

Date

Date

University of California at Berkeley

Fall 1996

Lower Bounds for Fundamental Geometric Problems

Abstract

Lower Bounds for Fundamental Geometric Problems

by

Jeffrey Gordon Erickson

Doctor of Philosophy in Computer Science

University of California at Berkeley

Professor Raimund Seidel, Chair

We develop lower bounds on the number of primitive operations required to solve several fundamental problems in computational geometry. For example, given a set of points in the plane, are any three colinear? Given a set of points and lines, does any point lie on a line? These and similar questions arise as subproblems or special cases of a large number of more complicated geometric problems, including point location, range searching, motion planning, collision detection, ray shooting, and hidden surface removal.

Previously these problems were studied only in general models of computation, but known techniques for these models are too weak to prove useful results. Our approach is to consider, for each problem, a more specialized model of computation that is still rich enough to describe all known algorithms for that problem. Thus, our results formally demonstrate inherent limitations of current algorithmic techniques. Our lower bounds dramatically improve previously known results and in most cases match known upper bounds, at least up to polylogarithmic factors.

In the first part of the thesis, we develop lower bounds for several degeneracy-detection problems, using *adversary arguments*. For example, we show that detecting colinear triples of points requires $\Omega(n^2)$ sidedness queries in the worst case. Our lower bound follows from the construction of a set of points in general position with several "collapsible" triangles, any one of which can be made degenerate without changing the orientation of any other triangle. Using similar techniques, we prove lower bounds for deciding, given a set of points in $\mathbb{R}^d$, whether any $d+1$ points lie on a hyperplane, whether any $d+2$ points lie on a sphere, or whether the convex hull of the point is simplicial.

In the second part, we consider offline range searching problems, which are usually solved using geometric divide-and-conquer techniques. To study these problems, we introduce the class of *partitioning algorithms*. We prove that any partitioning algorithm requires $\Omega(n^{4/3})$ time to detect point-line incidences in the worst case. Using similar techniques, we prove an $\Omega(n^{4/3})$ lower bound for deciding if a set of points lies entirely above a set of hyperplanes in dimensions five and higher.

Professor Raimund Seidel
Dissertation Committee Chair

To Nancy, who asked, "Then what?".

It was a large room. Full of people. All kinds.
And they had all arrived at the same building at more or less the same time.
And they were all free.
And they were all asking themselves the same question:

What is behind that curtain?

— Laurie Anderson, "Born, Never Asked", *Big Science*, 1982

Contents

List of Figures vi

1 Introduction 1
- 1.1 Old Results 2
  - 1.1.1 Output Size 2
  - 1.1.2 Decision Trees 3
  - 1.1.3 Semigroup Arithmetic 7
- 1.2 New Results 9

I Adversary Lower Bounds 12

2 Affine Degeneracies 13
- 2.1 Lower Bounds for a Restricted Problem 15
  - 2.1.1 The Planar Lower Bound 16
  - 2.1.2 Higher Dimensions 17
  - 2.1.3 Beating the Lower Bound 18
- 2.2 Lower Bounds for The General Problem 20
  - 2.2.1 An Alternate Proof in Two Dimensions 22
- 2.3 Allowable Queries 23
- 2.4 Implications and Open Problems 26
- 2.5 Out on a Limb 28

3 Convex Hull Problems 31
- 3.1 Preliminaries 32
- 3.2 The Lower Bound 33
- 3.3 Real Convex Hull Algorithms 37
- 3.4 Open Problems 38

4 Spherical Degeneracies 40
- 4.1 Circular Degeneracies 41
- 4.2 Proper Spherical Degeneracies 42
- 4.3 Open Problems 45

5 Linear Satisfiability Problems 47
5.1 Preliminaries 49
5.2 The Lower Bound 50
5.2.1 The Infinitesimal Adversary Configuration 51
5.2.2 Moving Back to the Reals 55
5.2.3 Perturbing into General Position 57
5.3 Matching Nonuniform Upper Bounds 58
5.4 Conclusions and Open Problems 59

II Divide-and-Conquer Lower Bounds 61

6 Hopcroft's Problem 62
6.1 Easy Quadratic Lower Bounds 64
6.2 Incidences and Monochromatic Covers 68
6.2.1 One Dimension 69
6.2.2 Two Dimensions 71
6.2.3 Three Dimensions 72
6.2.4 Higher Dimensions 76
6.2.5 A Lower Bound in the Semigroup Model 80
6.3 Partitioning Algorithms 81
6.3.1 The Basic Lower Bound 82
6.3.2 The Lower Bound for the Decision Problem 83
6.3.3 The Lower Bound for the Counting Problem 85
6.3.4 Containment Shortcuts Don't Help 86
6.4 Real Algorithms for Hopcroft's Problem 88
6.5 Conclusions and Open Problems 90

7 Halfspace Emptiness 92
7.1 Projective Polyhedra 93
7.2 Polyhedral Separation 96
7.3 Polyhedral Covers 99
7.4 Polyhedral Partitioning Algorithms 102
7.5 Conclusions and Open Problems 106

Bibliography 108

Index of Notation 122

Index 124

List of Figures

2.1 A planar adversary construction for nonvertical degeneracies. 16
2.2 The dual version of our planar adversary construction. 17
2.3 A planar adversary construction for arbitrary degeneracies. 22
2.4 Another planar adversary construction for arbitrary degeneracies. 23
2.5 Minimum area triangles are not necessarily collapsible. 27

3.1 The convex hull adversary construction in three dimensions. 36

6.1 An easy adversary construction for Hopcroft's problem. 65
6.2 A harder adversary construction for Hopcroft's problem 66
6.3 Collections of points and lines with simple relative orientation matrices. . . 69
6.4 $I_3(n, m) = mn$. Every point lies on every plane. 73
6.5 Comparison of lower bounds for $\mu_2^\circ(n, m)$ and $\mu_3^\circ(n, m)$. 76
6.6 Eliminating containment shortcuts. 87

7.1 The suspension and projection of a polygon by a point. 95
7.2 Worst-case configuration for halfspace emptiness. 103

Acknowledgements

The scariest thing about saying your thank-yous
is the certainty that you will leave someone out.
In the meantime... Well, there's nothing for it but:
It's a list of names, and a long one.

— Neil Gaiman, Afterword to *Sandman*, 1996

Most of my research at Berkeley was supported by Raimund Seidel's NSF PYI grant CCR-9058440 and by a GAANN fellowship. Other research was done at Smith College during the 1993 Regional Geometry Institute (funded by NSF grant DMS-9013220), Universität des Saarlandes, the Max-Planck-Institut für Informatik (unofficially), Freie Universität Berlin, and Universiteit Utrecht. Thanks to Joe O'Rourke, Emo Welzl, Otfried Schwarzkopf, and especially Kurt Mehlhorn for their hospitality. Most of this thesis has been previously published [73, 74, 76, 78] or will be soon [77], although I have made several corrections and additions.

As much as I may have tried, a thesis is far from an individual effort. I am particularly grateful to Raimund Seidel, Umesh Vazirani, and Bernd Sturmfels for their careful reading of the thesis and their helpful suggestions. Comments by anonymous readers and referees led to several significant improvements in both content and presentation, and the elimination of at least one major bug. While credit can and should be spread around, blame should never be; all remaining errors are my own responsibility!

Raimund Seidel has been an wonderful advisor and a good friend for the last four years. I thank him for his guidance, his humor, his sanity, and his (im)patience; I only wish my timing had been better. I would also like to thank Martina Seidel for her hospitality in Saarbrücken, and Miriam Seidel for letting me play with her stuffed animals.

It is impossible to overstate the influence of the Berkeley theory group. Dick Karp, who introduced me to adversaries and little birdies, is without question the best teacher I have ever had. Umesh Vazirani introduced me to complexity theory, somehow making it seem simple. John Canny introduced me to semi-algebraic sets and their potential uses in computational geometry, although the result is probably not what he had in mind. Thanks also to the many profs, grads, undergrads, and postdocs who put up with my stupid questions, and who offered their own much better questions in return, especially Micah Adler, Nina Amenta, Sanjeev Arora, Elwyn Berlekamp, Ethan Bernstein, John Binder, Manuel

Blum, David Forsyth, Dan Garcia, Mor Harchol-Balter, Ari Juels, Mike Mitzenmacher, Sridhar Rajagopalan, Troy Shahoumian, Alistair Sinclair, Dan Spielman, Bernd Sturmfels, Randi Thomas, Taku Tokuyasu, Hal Wasserman, and David Wolfe. Hopefully no one will be disappointed that my thesis only mentions randomization once. Finally, thanks to Caffé Nefeli for the daily fix.

I am deeply indebted to David Eppstein for introducing me to computational geometry, my first taste of real theoretical research, and a thoroughly unrealistic standard of productivity. The other Irvine theory faculty—Mike Dillencourt, Dan Hirschberg, and George Lueker—also supported and inspired my transition from professional software engineer to amateur theoretical computer scientist. I would especially like to thank Dick Taylor for letting me into graduate school against his (and the admissions committee's) better judgment.

I am honored that the computational geometry community has welcomed me into their ranks. An early visit with Marshall Bern and David Dobkin at Xerox PARC was an inspiration. Whether they realize it or not, the interest and encouragement of Scot Drysdale and Mike Goodrich have been invaluable. Extra special thanks to Jack Snoeyink and Emo Welzl for service far above and beyond the call of duty.

Without my "real world" work experience, first at StyleWare and later at Claris, I might never have learned how to turn knowledge and inspiration into something concrete. Thanks especially to Bob Hearn, Scott Holdaway, Scott Lindsey, Paul Lee, Ernie Pan, Syd Polk, and my dog Bo-bo for their friendship and humor.

My parents, Jon Erickson and Constance Cook, have encouraged and supported my bizarre mathematical habits for as long as I can remember. None of this would have been possible without the numerous opportunities that they gave me.

Finally, I would like to thank my wife Kim Whittlesey for her love and enthusiastic support. I hope the next seventy years will be as wonderful as the last seven.

Chapter 1

Introduction

The overwhelming majority of research in computational geometry is devoted to the design and analysis of algorithms and data structures, in an effort to find the fastest possible solutions to geometric problems. This thesis, however, is concerned with lower bounds on the complexity of these problems. Here, the goal is to describe inherent limits on the ability of a model of computation to solve a geometric problem, or in other words, to define the phrase “fastest possible solution”.

This thesis describes new lower bounds on the complexity of several fundamental decision problems that arise in computational geometry. For example: Given a set of points in the plane, are any three colinear? Given a set of points and lines in the plane, does any point lie on any line? These and similar questions arise as subproblems or special cases of a large number of more complicated geometric problems, including point location, range searching, motion planning, collision detection, ray shooting, visibility, and hidden surface removal [86, 75].

In the last several years, computational geometers have developed powerful techniques for solving these problems efficiently, at least in theory, and there is a common belief that the best known algorithms for these problems are optimal or very close to optimal. Unfortunately, there are still large gaps between the running times of these algorithms and the best known lower bounds. Currently available methods for deriving lower bounds in general models of computation (algebraic decision trees, for example) are simply not powerful enough to give very good results.

The approach taken in this thesis, therefore, is to develop new lower bounds in models of computation that are more specialized to each problem. For each problem, the model we consider is powerful enough to describe all known algorithms for solving it. Thus, the lower bounds we develop formally demonstrate the limitations of solving the problem using currently available techniques.

Results in this area are important for two reasons. A good lower bound for a problem, in a sufficiently general model of computation, indicates that an efficient solution to the problem is essentially impossible. In such a situation, research is more profitably aimed at deriving approximate solutions, deriving algorithms that are efficient (or correct) only with high probability (either with respect to some input distribution or internal randomization), or considering useful special cases.

Lower bounds in specialized models of computation are also useful, provided those models accurately describe all known algorithms. Such results direct algorithms researchers to consider fundamentally new techniques for solving geometric problems, in an effort to avoid the limitations of current approaches. Moreover, by describing exactly why current approaches fail, these results ease the discovery of new approaches.

1.1 Old Results

Before describing our specific new results, we begin with a whirlwind tour of previously known lower bounds in computational geometry. Along the way, we will point out inherent limitations of the techniques used to derive these bounds.

In order to say anything meaningful about algorithmic complexity, we must first agree on what an algorithm is. All lower bounds are expressed in terms of a *model of computation*, a set of assumptions about the operations algorithms are allowed to perform and the how much "time" each of these operations costs. (Actually, all upper bounds are expressed in terms of a model of computation, too, but the model is rarely described explicitly.) For purposes of proving lower bounds, we almost always ignore all but a few simple operations such as branches, input/output, assignments, pointer traversals, or arithmetic, and then restrict further which of these operations we allow.

1.1.1 Output Size

For many geometric problems, the only available way to prove tight lower bounds is by bounding the combinatorial complexity of the output. For example, any algorithm that constructs an arrangement of n hyperplanes in $\mathbb{R}^d$ must take time $\Omega(n^d)$[1] in the worst case, because the arrangement can have that many cells [62]. Similarly, any algorithm that constructs the convex hull of a set of n points in $\mathbb{R}^d$ must take time $\Omega(n^{\lfloor d/2 \rfloor})$ in the worst case, because there are polytopes with that many facets [87].

This approach is clearly worthless if the output size is always small, or if we want to prove output-sensitive lower bounds of the form $\Theta(f(n) + r)$, where r is the output size. For almost all the problems we consider in this thesis, the output size is a single bit.

1.1.2 Decision Trees

Decision trees are one of the simplest and most widely studied models of computation. A k-*ary decision tree* is a rooted directed tree in which every internal node has k children. If the degree k is unspecified, we take it to be a small constant; in practice k is almost always either 2 or 3. Associated with each internal node is a question about the input, or *query*, with k possible answers. Each answer is associated with a unique outgoing edge. Each leaf is labeled with an output value. To compute with such a tree, we start at the root and proceed down to a leaf. At each internal node, the answer to its associated query tells us which child to go to next. The running time of the algorithm is the number of queries asked, which in the worst case is the depth of the tree. Memory management, data movement, arithmetic, and other aspects of real-world algorithms are simply ignored. Typically, a decision tree is designed only to deal with inputs of a particular size. Thus, an algorithm in this model is represented by a family of decision trees, one for each possible input size.

One of the easiest techniques for proving lower bounds on the depth of decision trees is based on *information theory*: if a problem has N possible outputs, then any decision tree that solves it must have at least N leaves, and thus its depth must be at least $\lceil \log_k N \rceil$. For example, finding an unknown integer between 1 and 1 000 000 using only yes-no questions requires at least $\lceil \log_2 1\,000\,000 \rceil = 20$ questions. Since n items can be ordered in $n!$ different ways, any k-ary decision tree that sorts a list of n items must have depth $\lceil \log_k(n!) \rceil = \Omega(n \log n)$.

There are many problems for which the information-theoretic bound is far too weak to be useful. For example, consider the simple problem of choosing the largest of n

[1] We assume the reader is familiar with the standard asymptotic notations $o(\cdot), O(\cdot), \Theta(\cdot), \Omega(\cdot)$, and $\omega(\cdot)$. Otherwise, see [52, 103].

input items. Any reasonable algorithm obviously requires linear time[2], but since there are only n possible outputs, the information-theoretic lower bound is only $\Omega(\log n)$. In fact, a matching $O(\log n)$ upper bound can be obtained by a decision tree that uses queries like "Is the largest item in the first half of the list?"

If we restrict the questions the algorithm can ask, we can often improve the information-theoretic lower bound by using an *adversary argument*. The argument works as follows. Instead of choosing a single input in advance and letting the algorithm ask questions about it, an all-powerful malicious adversary *pretends* to choose an input, and answers questions in whatever way will make the algorithm do the most work. If the algorithm doesn't ask enough queries, then there will be several different inputs, each consistent with the adversary's answers, that should result in different outputs. Whatever the algorithm outputs, the adversary can "reveal" an input that is consistent with all of its answers, but inconsistent with the algorithm's output. The adversary approach hinges critically on the fact that a decision tree only has access to its input through its queries; the algorithm cannot distinguish the adversary from an honest user that chooses an input in advance.

Adversary algorithms have proven particularly useful in proving lower bounds on the depth of *comparison trees*. In a comparison tree, the input is a set of items from some totally ordered domain (typically $\mathbb{Z}$ or $\mathbb{R}$), and every query is a comparison between two input values. The following simple adversary argument implies that any comparison tree that chooses the largest of n input values $x_1, x_2, \ldots, x_n$ must have depth at least $n-1$. The adversary initially presents an arbitrary list of distinct values, say $x_i = i$ for all i. If an algorithm declares that x_n is the largest input value after fewer than $n-1$ comparisons, there must be at least one other input value $x_i \neq x_n$ that is bigger than anything the algorithm compared it to. The adversary can change the value of x_i to $n+1$, and then "reveal" its modified input, proving the algorithm wrong. Since the algorithm cannot distinguish between the original input and the modified input, it cannot possibly give the correct result for both.

Adversary arguments have also been used to prove lower bounds on the number of comparisons required to choose the largest and smallest elements in a list, the second-largest element [102, 104] (see also [60]), the median element [21, 122], or the kth largest element for arbitrary values of k [122, 99]. The minimum number of comparisons required

[2] unless we allow a small probability of error [148]

to solve the last two problems is still not known; the simplest open case is $k = 3$. Several more references can be found in [17].

We can generalize comparison trees by allowing more complicated query expressions. A d*th order algebraic decision tree* [125, 126, 138] is a ternary decision tree whose input is a vector $(x_1, x_2, \ldots, x_n)$ of real numbers, and each of whose queries asks for the sign of a multivariate *query polynomial* $f(x_1, x_2, \ldots, x_d)$ of degree at most d. For example, in a comparison tree, every query is of the form $x_i - x_j$. Typically, the parameter d is omitted, and assumed to be a fixed constant. An important special case of algebraic decision trees are *linear decision trees*, in which every query polynomial is linear (*i.e.*, $d = 1$) [58, 117].

The *element uniqueness problem* asks, given a list of n numbers, whether any two are equal. It seems "obvious" that any algorithm that solves this problem must sort the input values[3], and so the decision tree complexity "ought" to be $\Omega(n \log n)$. Unfortunately, since there are only two possible outputs, YES and NO, the information-theoretic bound is trivial. A simple adversary argument establishes a lower bound of $n - 1$ in the comparison tree model, but that is hardly satisfactory.

Dobkin and Lipton [58] proved a lower bound of $\Omega(n \log n)$ for the element uniqueness problem in the linear decision tree model, and therefore also in the comparison tree model, as follows. Fix a linear decision tree, and for each leaf ℓ, let X_ℓ be the set of input vectors for which computation reaches ℓ. X_ℓ is the intersection of several hyperplanes and linear halfspaces, and is therefore a convex polytope; in particular, X_ℓ is connected. Let $W \subset \mathbb{R}^n$ be the set of input vectors whose elements are distinct. In order for the decision tree to be correct, W must equal the union of X_ℓ over all leaves ℓ whose output label is YES. Clearly W has $n!$ disjoint connected components. It follows that any linear decision tree that solves the element uniqueness problem must have at least $n!$ YES leaves. Otherwise, there must be some leaf ℓ such that X_ℓ intersects more than one component of W and therefore intersects the complement of W, a contradiction. Any ternary tree with $n!$ leaves has depth at least $\lceil \log_3(n!) \rceil = \Omega(n \log n)$. More generally, any linear decision tree that solves the *set membership problem* for a set W with $\#W$ connected components must have depth at least $\lceil \log_3(\#W) \rceil$.

Following ideas of Yao [154], this argument was later generalized to higher-order

[3] Like many "obvious" statements, this is actually false! The discriminant $\prod_{i<j}(x_i - x_j)$ can be directly computed using $O(n \log n)$ multiplications (and $O(n \log^2 n)$ additions) without sorting the inputs [15, 141]. The input elements x_i are distinct if and only if this expression is not equal to zero, but the expression gives us (almost) no information about the sorted order of the inputs.

algebraic decision trees by Steele and Yao [138] and Ben-Or [16]. The generalization hinges on the following theorem of algebraic geometry independently proven by Petrovskiĭ and Oleĭnik [120, 121], Thom [147], and Milnor [115]. (See also [13, 14, 152].) A *semialgebraic* set is the set of points satisfying a finite number of polynomial equations and inequalities.

Theorem 1.1 (Petrovskiĭ/Oleĭnik/Thom/Milnor). *Let V be a semialgebraic set in $\mathbb{R}^n$, defined by t polynomial inequalities of maximum degree d. The sum of the Betti numbers of V is $(td)^{O(n)}$; in particular, V has $(td)^{O(n)}$ connected components.*

This theorem implies that, if t is the depth of a dth order algebraic decision tree that solves the set membership problem for W, then W has at most $3^t(td)^{O(n)}$ connected components. It follows immediately that the depth must be $\Omega(\log \#W - n \log d)$; in particular, the complexity of the element uniqueness problem is $\Omega(n \log n)$. Ben-Or [16] strengthened the argument further, deriving a similar lower bound in the stronger *algebraic computation tree* model.

More recent techniques imply lower bounds based on different complexity measures of the set W, such as its higher order Betti numbers [158], its Euler characteristic [19, 159], or the number of its lower-dimensional faces [118, 157, 94]. Lower bounds can also be derived by considering the complexity of the complement of W, the interior of W, the closure of W, or the intersection of W with another semi-algebraic subset of $\mathbb{R}^n$. For further generalizations, see [156, 93]. All of these techniques are essentially information-theoretic; in every case, the implied lower bound is the logarithm of the complexity of W.

A large class of geometric problems can be formalized as asking whether a point lies in a (semi-)algebraic set W defined by a polynomial number of constant-degree polynomial (in-)equalities. Fortunately, this is precisely the framework in which these lower bound arguments apply. As a consequence, it is quite easy to derive $\Omega(n \log n)$ lower bounds for many of these problems. Unfortunately, this is the best we can do. The Petrovskiĭ/Oleĭnik/Thom/Milnor theorem and its generalizations [13, 14, 152] imply that the complexity of W, in any reasonable sense of the word "complexity", is at most $n^{O(n)}$. Thus, for these problems, *no known lower bound technique can imply lower bounds bigger than $\Omega(n \log n)$ in the algebraic decision tree model.* An $\omega(n \log n)$ lower bound for *any* natural problem solvable in polynomial time would be a major breakthrough. Quadratic lower bounds are known for a few NP-complete problems [59, 16, 93], but again, this is the best lower bound we can prove, since for these problems the number of defining inequalities is only singly-exponential.

1.1.3 Semigroup Arithmetic

The *semigroup arithmetic model* was introduced by Fredman [82, 81] and refined by Yao [155] and Chazelle [34] to study the complexity of range searching data structures. In this model, each point in the input is given a value from an abelian semigroup satisfying a mild technical condition[4]. These points are preprocessed into a data structure, so that the sum of the weights of the points contained in an arbitrary query range can be computed quickly. The data structure consists of a collection of partial sums, called *generators*, which are added together to produce the answer to each range query. The size of the data structure is the total number of generators. The time required to answer a range query is the minimum number of generators whose sum is the correct answer. The only computational activity considered in this model is the addition of semigroup values; branches, pointer traversals, memory allocation, and other aspects of real-world range searching algorithms are ignored.

Algorithms in the semigroup model can exploit special properties of the semigroup, but they are *not* allowed to exploit the particular weights on the points; the same algorithm must work for *any* assignment of weights to the points. In effect, answers must be computed symbolically. Subtraction of semigroup values is also disallowed, even if the semigroup is actually a group, as is frequently the case.

The semigroup model was originally introduced by Fredman [81, 82], who derived lower bounds for dynamic orthogonal and halfplane range searching data structures, which must support the insertion and deletion of points as well as range queries. The model was first applied to static range searching problems by Yao [155], who proved lower bounds for orthogonal range searching in two dimensions. Vaidya [150] proved lower bounds for orthogonal range searching in higher dimensions, which were later improved by Chazelle [34]. Chazelle also derived lower bounds for simplex range searching [33], and with Brönnimann and Pach, halfspace range searching [23]. All of these online lower bounds give tradeoffs between the space required by the data structure and the resulting query time. The model

[4]Specifically, the semigroup must be *faithful*: any two identically equal linear forms must involve exactly the same set of variables, although not necessarily with the same coefficients. For example, $(\mathbb{Z}, +)$, $(\{0, 1\}, \vee)$, $(\mathbb{R}, \max)$, and $(2^P, \cup)$ (for any nonempty set P) are faithful semigroups, but $(\{0\}, +)$ and $\mathbb{Z}/2\mathbb{Z}$ are not [155].

was later generalized to deal with offline problems, where the ranges are all known in advance. Chazelle and Rosenberg [42, 41] derived lower bounds for computing partial sums in multi-dimensional arrays (a special case of orthogonal range searching where the points lie on a lattice), and Chazelle [38] proved lower bounds for both orthogonal and simplex range searching. With the exception of halfspace range searching [23], all of these lower bounds are optimal, up to polylogarithmic or n^ε factors, in the semigroup model. For further details, we refer the reader to an excellent survey by Matoušek [112].

Every set of points and set of ranges defines a bipartite incidence graph, where the presence of an edge denotes that a point lies in a range. A key step in the proofs of many semigroup lower bounds is the construction of a set of points and ranges, such that (some subgraph of) the incidence graph has several edges but no large complete bipartite subgraphs. For example, Fredman's lower bounds for dynamic halfplane range searching [82] rely on a construction of Erdős of n points and n lines in the plane with $\Omega(n^{4/3})$ point-line incidences; since any pair of lines intersects in at most one point, the incidence graph for this point-line configuration has no $K_{2,2}$. We will use a similar technique in the second part of this thesis.

Unfortunately, the semigroup arithmetic model is inappropriate for studying the complexity of range *emptiness* problems, where we just want to know whether any points lie inside a given query range. If the range is empty, then the algorithm will perform no additions; conversely, if we perform even a single addition, the range must not be empty. The problem is that algorithms in this model are not allowed to "know" the weights assigned to the points. All of the previous lower bounds hold when weights are taken from the faithful semigroup $(\{0,1\}, \vee)$, but *not* under the assumption that every point is given weight 1. This may be somewhat counterintuitive, since we can "obviously" remove the points with weight zero before doing anything else. However, in many range searching applications, weights are often not assigned in advance, but are determined implicitly by some other criterion, for example, presence in or absence from some other query range.

Very little is known about the complexity of range searching in the *group* arithmetic model, where subtractions are also allowed. Willard [153] considers dynamic orthogonal range searching data structures and shows that under some fairly restrictive assumptions, allowing subtractions cannot make these structures more efficient. Chazelle derives nontrivial (but very weak) lower bounds for offline halfplane [37] and orthogonal range counting [38] by examining the spectra of point-range incidence matrices. Absolutely nothing is known about range searching over more complicated domains such as rings or fields.

A useful special case of range searching is range *reporting*, where we want to know which points are in each query range, not just how many. Since the output from a reporting query can be quite large, we would like bounds of the form $\Theta(f(n)+r)$, where r is the number of points reported. Techniques similar to those used in the semigroup model imply lower bounds of this form in Tarjan's *pointer machine* model [144]. In this model, a data structure is represented by a directed graph in which every node has constant outdegree. The graph has a special starting node, or *source*. Each node in the graph is either unlabeled or is labeled with the index of exactly one point. A query is answered by visiting nodes in the graph, starting at the source and traversing edges in arbitrary order, until the index of every point in the query range has been seen at least once. Query algorithms are also allowed to modify the data structure by adding new nodes and adding or deleting edges between previously visited nodes. The query time is the number of nodes visited; all other aspects of computation are ignored. Chazelle [34] derives lower bounds for orthogonal range reporting in this model; his techniques were applied to simplex range reporting by Chazelle and Rosenberg [44]. Again, this model is inappropriate for studying range emptiness problems, since if a query range is empty, we don't need to do any work at all.

1.2 New Results

The remainder of this thesis divides naturally into two parts.

In Part I (Chapters 2 through 5), we derive lower bounds for several degeneracy-detection problems. For each of the problems we consider, the previously best lower bound, in any model of computation, was only $\Omega(n \log n)$ [138, 16].

In Chapter 2, we show that, in the worst case, $\Omega(n^d)$ sidedness queries are required to decide, given n points in $\mathbb{R}^d$, whether any $d+1$ lie on a common hyperplane. Since there is an algorithm that solves this problem in time $O(n^d)$ [39, 68, 69], our lower bound is tight. Our lower bound follows from an extremely simple adversary argument, based on the construction of a set of points in general position with $\Omega(n^d)$ "collapsible" simplices, any one of which can be made degenerate without changing the result of any other sidedness query. If the algorithm doesn't do enough work, then the adversary can collapse some unchecked simplex, resulting in a degenerate set of points that the algorithm cannot distinguish from the original nondegenerate set. We also show that our lower bound still holds if we allow a wide variety of other computational primitives, such as coordinate comparisons and slope comparisons.

In Chapter 3, using similar techniques, we show that $\Omega(n^{\lceil d/2 \rceil - 1})$ sidedness queries are required to determine is the convex hull of n points in $\mathbb{R}^d$ is simplicial, or to count the number of convex hull facets. This matches known upper bounds when d is odd [36].

In Chapter 4, we show that $\Omega(n^{d+1})$ insphere queries are required to decide if any $d+2$ points lie on a common finite-radius sphere in $\mathbb{R}^d$. In the plane, $\Omega(n^3)$ incircle queries are required to decide if any four points lie on a common circle or line. These lower bounds are optimal [68, 69].

In Chapter 5, we prove an $\Omega(n^{\lceil r/2 \rceil})$ lower bound for the following problem: For some fixed linear equation in r variables, given a set of n real numbers, do any r of them satisfy the equation? Our lower bound holds in a restricted linear decision tree model, in which each decision is based on the sign of an arbitrary linear combination of r or fewer inputs. In this model of computation, our lower bound is as large as possible. Previously, this lower bound was known only for even r, and only for one special case [55, 56, 80]. A key step in the lower bound proof is the introduction of formal infinitesimals into the adversary configuration. We use a theorem of Tarski [146] to show that if we can construct a hard input containing infinitesimals, then for every decision tree algorithm, there exists a corresponding set of real numbers which is hard for that particular algorithm.

In Part II (Chapters 6 and 7), we derive lower bounds for problems that are typically solved by geometric divide-and-conquer techniques.

In Chapter 6, we establish new lower bounds on the complexity of the following basic geometric problem, attributed to John Hopcroft: Given a set of n points and m hyperplanes in $\mathbb{R}^d$, is any point contained in any hyperplane? We define a general class of *partitioning algorithms*, and show that in the worst case, for all m and n, any such algorithm requires time $\Omega(n \log m + n^{2/3}m^{2/3} + m \log n)$ in two dimensions, or $\Omega(n \log m + n^{5/6}m^{1/2} + n^{1/2}m^{5/6} + m \log n)$ in three or more dimensions. We obtain slightly higher bounds for the counting version of Hopcroft's problem in four or more dimensions. Informally, a partitioning algorithm divides space into a constant number of regions, determines which points and lines intersect which regions, and recursively solves the resulting subproblems. Our planar lower bound is within a factor of $2^{O(\log^*(n+m))}$ of the best known upper bound [111].[5] The previously best lower bound was only $\Omega(n \log m + m \log n)$ [138, 16]. We develop our lower bounds in two stages. First we define a combinatorial representation of the relative order type of a set of points and hyperplanes, called a *monochromatic cover*, and derive lower bounds on its size in the worst case. We then show that the running time of any partitioning algorithm is bounded below by the size of some monochromatic cover. As a related result, using a straightforward adversary argument, we derive a *quadratic* lower bound on the complexity of Hopcroft's problem in a surprisingly powerful decision tree model of computation.

Finally, in Chapter 7, we derive a lower bound of $\Omega(n \log m + n^{2/3}m^{2/3} + m \log n)$ for the following halfspace emptiness problem: Given a set of n points and m hyperplanes in $\mathbb{R}^5$, is every point above every hyperplane? This matches the best known upper bound up to polylogarithmic factors [107, 3, 29], and improves the previously best lower bound $\Omega(n \log m + m \log n)$ [138, 16]. We also obtain marginally better bounds in higher dimensions. Our lower bound applies to partitioning algorithms in which every query region is a polyhedron with a constant number of facets.

At the end of each chapter, we outline some relevant open problems and suggest directions for further research.

Sadly enough, surveying the status of lower bounds in computational geometry is a fairly easy task.

— Bernard Chazelle, "Computational Geometry: A Retrospective", 1994

[5]The iterated logarithm $\log^* n$ is 1 for all $n \leq 2$ and $1 + \log^*(\log_2 n)$ for all $n > 2$.

Part I

Adversary Lower Bounds

Down with Euclid! Death to the triangles!

— Jean Dieudonné, c. 1960

An adversary means opposition and competition,
but not having an adversary means grief and loneliness.

— Zhuangzi (Chuang-tsu), c. 300 BC

Chapter 2

Affine Degeneracies

A fundamental problem in computational geometry is determining whether a given set of points is in "general position." A simple example of this type of problem is determining, given a set of points in the plane, whether any three of them are colinear. In 1983, van Leeuwen [151] asked for an algorithm to solve this problem in time $o(n^2 \log n)$. Chazelle, Guibas, and Lee [39] and Edelsbrunner, O'Rourke, and Seidel [68] independently discovered an algorithm that runs in time and space $O(n^2)$ by constructing the arrangement of lines dual to the input points.[1] Edelsbrunner *et al.* [68] also solved the higher-dimensional version of this problem, which we call the *affine degeneracy problem* Their algorithm, given n points in $\mathbb{R}^d$, determines whether $d+1$ of them lie on the same hyperplane, in time and space $O(n^d)$.[2] Edelsbrunner and Guibas [65] later improved the space bound to $O(n)$ in all dimensions.

A basic primitive used by all of these algorithms is the *sidedness query*: Given $d+1$ points $p_0, p_1, \ldots, p_d$, does the point p_0 lie "above", on, or "below" the oriented hyperplane $\mathrm{aff}(p_1, \ldots, p_d)$? These are also sometimes called orientation tests, simplex queries, or (in the plane) triangle queries. The result of a sidedness query is given by the sign of

[1] We refer readers unfamiliar with projective duality to [140].

[2] The original analysis of their algorithm was flawed. A correct proof of the crucial Zone Theorem was later given by Edelsbrunner, Seidel, and Sharir [69].

the following determinant.

$$\begin{vmatrix} 1 & p_{01} & p_{02} & \cdots & p_{0d} \\ 1 & p_{11} & p_{12} & \cdots & p_{1d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & p_{d1} & p_{d2} & \cdots & p_{dd} \end{vmatrix}$$

The value of this determinant is $d!$ times the signed volume of the simplex spanned by $p_0, \ldots, p_d$. The *orientation* of a simplex $(p_0, p_1, \ldots, p_d)$ is the result of a sidedness query on its vertices (in the order presented). If the orientation is zero, we say that the simplex is *degenerate.*

In the algebraic decision tree and algebraic computation tree models, there is a somewhat trivial lower bound of $\Omega(n \log n)$ on finding affine degeneracies in any dimension, since it takes $\Omega(n \log n)$ time just to determine whether all the points are distinct [138, 16]. Prior to the results described in this chapter, no better lower bound was known in any model of computation.

Two sets of labeled points are said to have the same *order type* if corresponding simplices have the same orientation. The order type of a set of points can be represented by the face lattice of its dual hyperplane arrangement or by its lambda-matrix [88], both representations requiring space $\Omega(n^d)$. One might consider representing order types by canonical sets of points. Unfortunately, the full field of algebraic numbers is required to represent every planar order type [96], and even among integer order types, point coordinates must be doubly-exponential in the worst case [90].

The fastest known algorithm for determining the order type of a set of points constructs its dual hyperplane arrangement in time and space $O(n^d)$ [68, 69]. Even though all known representations of order type require space $\Omega(n^d)$, there is some hope of a smaller representation, and thus, a faster algorithm, since it is known that there are only $(n/d)^{\Theta(d^2 n)} = 2^{\Theta(n \log n)}$ order types [89]. Prior to the results in this chapter, the information-theoretic lower bound of $\Omega(n \log n)$ was the only lower bound known for this problem.

In this chapter, we first derive a lower bound of $\Omega(n^d)$ on the number of sidedness queries required to decide if a set of n points in $\mathbb{R}^d$ is affinely degenerate, or to determine the set's order type. This matches known upper bounds. Our lower bound holds in a decision tree model of computation in which every decision is based on the result of a sidedness query.

We are not allowed, for example, to compare the values of different sidedness determinants. This is not quite as unreasonable a restriction as it may appear at first glance; all known algorithms for determining degeneracy or order type rely (or can be made to rely) exclusively on sidedness queries [39, 65, 68]. Our lower bound implies that there is no hope of improving these algorithms unless other primitives are used.

These lower bounds follow from an extremely simple adversary argument. We describe a nondegenerate set of points that contains $\Omega(n^d)$ independent "collapsible" simplices, any one of which the adversary can make degenerate without changing the orientation of any other simplex. If an algorithm fails to perform a sidedness query for every collapsible simplex, the adversary can move the points so that the perturbed set is degenerate, and the algorithm will be unable to distinguish between the original set and the perturbed set. The adversary's point set consists of rational points on a particular polynomial curve.

Later in the chapter, we describe a large class of "allowable" primitives, which do not improve the lower bound even by a single sidedness query, even if we permit our algorithms to perform an arbitrarily large (but finite) number of them. Allowable queries include coordinate comparisons, slope comparisons, comparisons of second-order points defined as vertices of the dual hyperplane arrangements, and so forth. In fact, almost every bounded-degree multivariate polynomial is an allowable query.

2.1 Lower Bounds for a Restricted Problem

We begin by considering a restricted version of the degeneracy problem. Say that a hyperplane in $\mathbb{R}^d$ is *vertical* if it contains a line parallel to the x_d axis. The *nonvertical affine degeneracy problem* asks, given a set of n points in $\mathbb{R}^d$, whether there is a nonvertical hyperplane passing through $d+1$ of them. In this section, we prove the following theorem.

Theorem 2.1. *Any decision tree that detects nonvertical affine degeneracies in $\mathbb{R}^d$, using only sidedness queries, must have depth $\Omega(n^d)$.*

In order to give a more intuitive picture, we first consider the planar case, and then generalize to arbitrary dimensions.

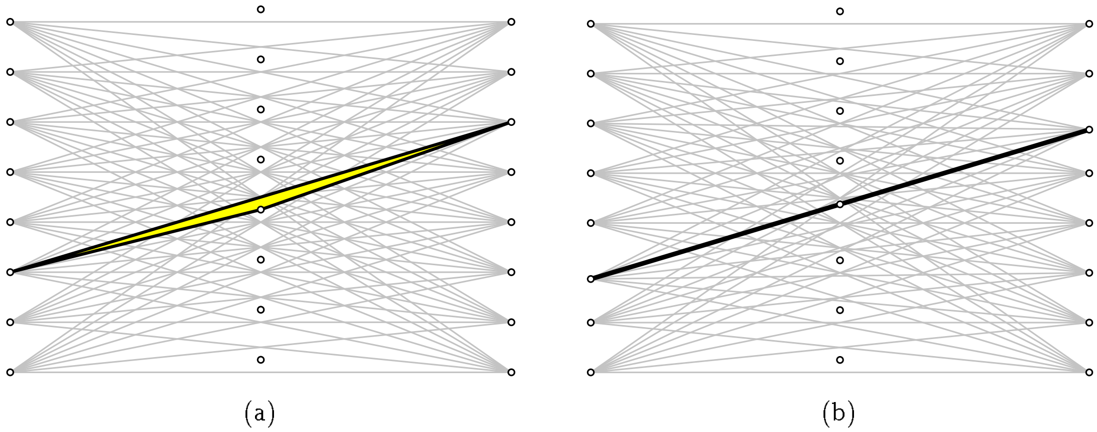


Figure 2.1. A planar adversary construction for nonvertical degeneracies. (a) The initial set, with one collapsible triangle shaded. (b) The perturbed set, showing collapsed triangle.

2.1.1 The Planar Lower Bound

Without loss of generality, we assume n is a multiple of 3. The adversary "presents" the following set of points:

$$S \triangleq \bigcup_{i=1}^{n/3} \{(-1, 4i), (0, 4i+1), (1, 4i)\}.$$

The set S consists of three smaller sets of points, evenly spaced along vertical line segments. See Figure 2.1(a). If we pick points p and r from the left and right segments, respectively, there is a unique point q in the middle segment such that the vertical distance from q to $\overleftrightarrow{pr}$ is exactly one. We shall refer to each such triple $\{p, q, r\}$ as a *collapsible triangle*, for the following reason. Without loss of generality, let q lie below $\overleftrightarrow{pr}$. If we perturb the set by moving p and r down by $1/2$ and moving q up by $1/2$, then the three points become colinear. See Figure 2.1(b). No other degeneracies are introduced by this perturbation; moreover, no other triangle changes orientation.

The adversary's point set S contains $n^2/9 = \Omega(n^2)$ collapsible triangles. If the algorithm does not check the orientation of every collapsible triangle, the adversary perturbs the set so that some unchecked triangle becomes degenerate. The algorithm cannot distinguish between the original point set and the perturbed point set. This completes the proof in the planar case.

It may be helpful to see what this construction looks like in the dual setting. Here

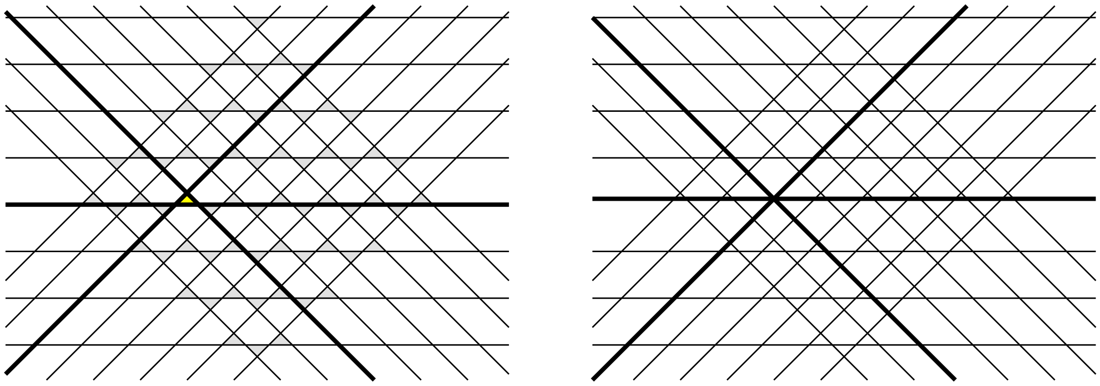


Figure 2.2. The dual version of our planar adversary construction. (a) The initial set, with collapsible triangles shaded. (b) The perturbed set, showing collapsed triangle.

we are given n lines in the plane and asked if any three of them have a common intersection.[3] The dual of the adversary's point set consists of three bundles of parallel lines. Two of the bundles meet in a mesh of squares, and the third cuts through the squares at a $45°$ angle, so that each square in the mesh has a small triangle cut off one corner. See Figure 2.2(a). Each of the small triangles in the mesh corresponds to a collapsible triangle in the primal point set. To collapse a triangle, the adversary simply translates its three bounding lines so that they intersect at the triangle's centroid. See Figure 2.2(b).

2.1.2 Higher Dimensions

For the d-dimensional problem, the adversary's point set consists of $d+1$ smaller sets. The points in each smaller set are evenly spaced along vertical line segments $l_0, l_1, \ldots, l_d$. These line segments intersect any horizontal hyperplane at the centroid and vertices of a regular $(d-1)$-simplex.

Without loss of generality, we assume n is a multiple of $3d$. Each of the outer segments $l_1, \ldots, l_d$ contains $2n/3d$ points, and l_0 contains the remaining $n/3$ points. The x_d coordinates of the outer points are multiples of $2d$ between 0 and $4n/3 - 2d$. Thus, any hyperplane defined by d points, one from each outer segment, intersects the x_d-axis at an even integer coordinate between 0 and $4n/3 - 2d$. The points in the inner set lie at alternate odd integer coordinates between 1 and $4n/3 + 1$. This gives us $\lfloor (d-1)/2 \rfloor$

[3]The restriction to nonvertical colinearities in the primal setting is reflected in the dual by ignoring the intersection points "at infinity" between parallel lines.

"wasted" points at the top of the inner segment, which we can ignore.

Suppose we pick one point from each of the outer sets. These points define a hyperplane h. The vertical distance between h and the unique point in the inner set that is closest to h is exactly 1. We refer to each such set of $d+1$ points as a *collapsible simplex*. The adversary can make any collapsible simplex degenerate by simultaneously moving the inner point up and the outer points down (or vice versa) a distance of $1/2$. Clearly, no other simplex changes orientation because of this perturbation. There are $(2n/3d)^d = \Omega(n^d)$ collapsible simplices in the adversary's point set, each of which must be checked by the algorithm.

This completes the proof of Theorem 2.1.

Since collapsing a simplex changes the order type of the set, we immediately have the following corollary.

Corollary 2.2. *Any decision tree that determines the order type of a set of n points in $\mathbb{R}^d$, using only sidedness queries, must have depth $\Omega(n^d)$.*

2.1.3 Beating the Lower Bound

If we know *in advance* that the points lie on $d+1$ vertical lines, then our $\Omega(n^d)$ lower bound can be defeated for all $d > 2$. In this special case, we can detect nonvertical degeneracies in $O(n^{d/2})$ time if d is even, and in $O(n^{(d+1)/2} \log n)$ time if d is odd. The algorithms that achieve these running times do not use only sidedness queries, but also compute the signs of certain linear forms. In addition to providing a pedagogical example of the importance of choosing the right model of computation, these algorithms suggest that a new approach may be required to extend our lower bounds into more general models of computation, at least in higher dimensions.

Suppose we are given $d+1$ sets $S_0, S_1, \ldots, S_d \subset \mathbb{R}^d$, each containing n points, such that each set S_i is contained in a vertical line l_i. The only possible nonvertical degeneracies contain one point from each line. The positions of the lines l_i determine constants a_i such that points $p_0 \in l_0, \ldots, p_d \in l_d$ lie on a nonvertical hyperplane if and only if their dth coordinates satisfy the equation

$$\sum_{i=0}^{d} a_i p_{id} = 0.$$

We describe two algorithms, one for even dimensions and one for odd dimensions. Our algorithms compare the weighted sums of tuples of dth coordinates of points, where the weight of each point is determined by the set from which it is taken. We call such a query a *tuple comparison*. Both algorithms work in two phases, a sorting phase and a scanning phase. In the sorting phase, both algorithms perform $\lceil d/2 \rceil$-tuple comparisons. In the scanning phase, both algorithms perform sidedness queries. In the odd-dimensional case, the sidedness queries we perform are actually $(d+1)/2$-tuple comparisons. In the discussion that follows, p_i always refers to a point in S_i.

If d is even, we sort all possible values of the expressions

$$\sum_{i=0}^{d/2-1} a_i p_{id} \quad \text{and} \quad \sum_{i=d/2}^{d-1} a_i p_{id}.$$

Then for each point $p_d \in S_d$, we scan through the two lists, looking for a pair of elements whose sum is $-a_d p_{dd}$. This algorithm runs in $O(n^{d/2+1})$ time.

If d is odd, we sort all possible values of the expressions

$$\sum_{i=0}^{\lfloor d/2 \rfloor} a_i p_{id} \quad \text{and} \quad \sum_{i=\lceil d/2 \rceil}^{d} -a_i p_{id},$$

and then simultaneously scan through the two lists for duplicate elements. This algorithm runs in $O(n^{(d+1)/2} \log n)$ time.

A simple variant of the odd-dimensional algorithm can be used to solve a slightly more general problem, in which the points are only constrained to lie on two vertical $(d+1)/2$-flats, which necessarily intersect at a vertical line l. Instead of sorting weighted sums, we sort the possible positions at which the affine hulls of $(d+1)/2$-tuples of points from the same $(d+1)/2$-flat intersect l. This algorithm also runs in $O(n^{(d+1)/2} \log n)$ time.

The special case of the affine degeneracy problem solved by these algorithms is an example of a *linear satisfiability problem*: Given a set of n real numbers, does any subset satisfy a fixed linear equation? We will consider linear satisfiability problems in greater detail in Chapter 5. The main result of that chapter (Theorem 5.1) implies that the algorithms we have just described are optimal, except possibly for a logarithmic factor when d is odd, when only sidedness queries and $\lceil d/2 \rceil$-tuple comparisons are allowed.

2.2 Lower Bounds for The General Problem

The *weird moment curve*, denoted $\omega_d(t)$, is the parameterized curve

$$\omega_d(t) = \left(t, t^2, \ldots, t^{d-1}, t^{d+1}\right).$$

where the parameter t ranges over the reals. The weird moment curve is similar to the standard moment curve $(t, t^2, \ldots, t^{d-1}, t^d)$, except that the degree of the last coordinate is increased by one.

If we project the weird moment curve down a dimension by dropping the last coordinate, we get a standard moment curve. Since every set of points on the standard moment curve is affinely nondegenerate, no d points on the d-dimensional weird moment curve lie on a single $(d-2)$-flat. However, it is possible for $d+1$ points to all lie on a single hyperplane. The following lemma characterizes these affine degeneracies.

Lemma 2.3. *Let $x_0 < x_1 < \cdots < x_n$ be real numbers. The orientation of the simplex $(\omega_d(x_0), \omega_d(x_1), \ldots, \omega_d(x_d))$ is given by the sign of $\sum_{i=0}^{d} x_i$. In particular, the simplex is degenerate if and only if $\sum_{i=0}^{d} x_i = 0$.*

Proof: The orientation of the simplex $(\omega_d(x_0), \omega_d(x_1), \ldots, \omega_d(x_d))$ is given by the sign of the determinant of the following matrix.

$$M = \begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^{d-1} & x_0^{d+1} \\ 1 & x_1 & x_1^2 & \cdots & x_1^{d-1} & x_1^{d+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & x_d & x_d^2 & \cdots & x_d^{d-1} & x_d^{d+1} \end{bmatrix}$$

The determinant of M is an antisymmetric polynomial of degree $\binom{d+1}{2} + 1$ in the variables x_i, and it is divisible by $(x_i - x_j)$ for all $i < j$. It follows that

$$\frac{\det M}{\prod_{i<j}(x_j - x_i)}$$

is a symmetric polynomial of degree one, and we easily observe that its leading coefficient is 1. (This polynomial is well-defined, since the x_i are distinct.) The only such polynomial is $\sum_{i=0}^{d} x_i$. $\square$

This result, or at least its proof, is hardly new. If we replace the weird moment curve by any polynomial curve, the orientation of a simplex is given by the sign of a Schur polynomial [131]. A determinantal formula for Schur polynomials was discovered by Jacobi in the mid-1800's [100].

Theorem 2.4. *Any decision tree that decides whether a set of n points in $\mathbb{R}^d$ is affinely degenerate, using only sidedness queries, must have depth $\Omega(n^d)$. If $d \geq 3$, this lower bound holds even when the points are known in advance to be in convex position.*

Proof: Let X denote the set of integers from $-dn$ to n. If we lift X up to the weird moment curve, the resulting set of points $\omega_d(X)$ contains $\Omega(n^d)$ degenerate simplices. To pick a degenerate simplex, choose arbitrary distinct positive elements $x_1, x_2, \ldots, x_d \in X$, and let $x_0 = -\sum_i x_i$.

The adversary initially presents the point set $\omega_d(X')$, where X' denotes the set $X+1/(2d+2) = \{x+1/(2d+2) \mid x \in X\}$. This point set is affinely nondegenerate, since the the sum of any $d+1$ elements in X' is always a half-integer. Choose arbitrary distinct positive elements $x'_1, x'_2, \ldots, x'_d \in X'$, and let $x'_0 = 1/2 - \sum_i x'_i$. The points $\omega_d(x'_i)$ form a collapsible simplex. To collapse it, the adversary shifts the points back to their original positions $\omega_d(x_i)$. The collapsed simplex is obviously degenerate. Moreover, since the expression $\sum_{i=0}^{d} x'_i$ changes by at most $1/2 - 1/(2d+2) < 1/2$ for any other simplex, no other simplex changes orientation. In particular, the collapsed simplex is the only degenerate simplex.

The adversary's point set contains $\binom{n}{d} = \Omega(n^d)$ collapsible simplices. If an algorithm does not check the orientation of every collapsible simplex, then the adversary perturbs the input so that some unchecked simplex becomes degenerate. The algorithm cannot distinguish between the original point set and the perturbed point set, even though only one of them is degenerate.

Since every set of points on the standard moment curve is in convex position, every set of points on the d-dimensional weird moment curve in convex position if $d \geq 3$. (Given a set of points in convex position in the plane, we can easily determine whether any three are colinear in $O(n \log n)$ time.) $\square$

Figure 2.3 illustrates the new two-dimensional construction. In order to make the colinearities more visible, the figure uses a curve of the form $y = x^3 - \alpha x$; since this is a linear transformation of the unit cubic $y = x^3$, all colinearities are preserved.

We emphasize that if the points are known *in advance* to lie on the weird moment curve, affine degeneracies can be detected in $O(n^{d/2})$ time if d is even, and in

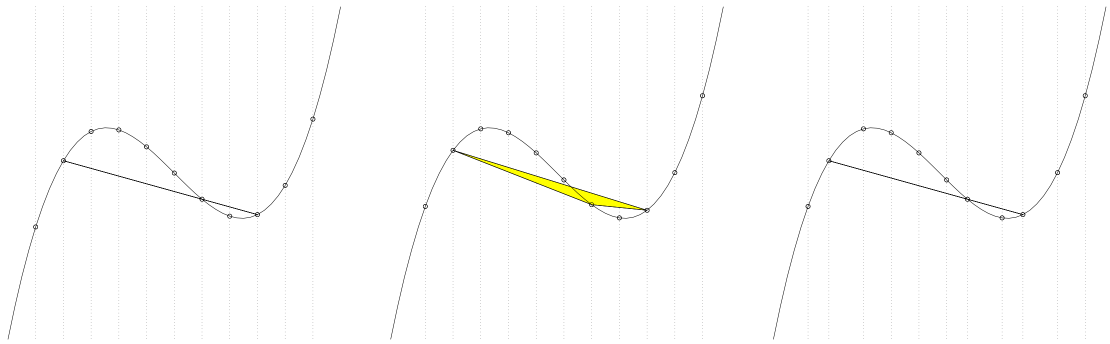


Figure 2.3. A planar adversary construction for arbitrary degeneracies. (a) The degenerate configuration, with one degenerate triangle emphasized. (b) The adversary configuration, with the corresponding collapsible triangle. (c) The corresponding collapsed configuration.

$O(n^{(d+1)/2} \log n)$ time if d is odd, by simple algorithms that use more complicated queries, similar to the algorithms described in Section 2.1.3.

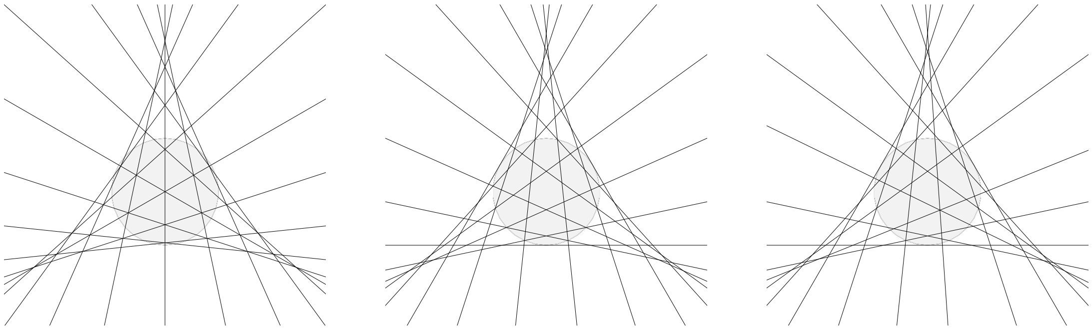
2.2.1 An Alternate Proof in Two Dimensions

In the 1950's, Sylvester noted that a set of n integer points on the unit cubic can have $n^2/8$ collinear triples [96]. Füredi and Palásti [84] improve this lower bound to roughly $n^2/6$ using a slightly different construction, which we describe below. We can use their construction to slightly improve our lower bound for the two-dimensional affine degeneracy problem. The resulting lower bound is the best that can be derived using our techniques, except possibly for some lower-order terms.

Füredi and Palásti describe their construction in the dual. Let $L(\alpha)$ be the line passing through the point $(\cos\alpha, \sin\alpha)$ at angle $-\alpha/2$ to the x-axis. The line $L(\alpha)$ also passes through the point $(\cos(\pi-2\alpha), \sin(\pi-2\alpha))$; if this is the same point as $(\cos\alpha, \sin\alpha)$, then the line is tangent to the unit circle at that point. Three lines $L(\alpha), L(\beta), L(\gamma)$ are concurrent if and only if $\alpha+\beta+\gamma \equiv 0 \pmod{2\pi}$. It follows that the set of lines $\{L(2\pi i/n) \mid 1 \le i \le n\}$ has $1 + \lfloor n(n-3)/6 \rfloor$ concurrent triples. See Figure 2.4(a). See [84] for further details. Related results are described in [24] and [72].

The set of lines $\{L((2i-1)\pi/n) \mid 1 \le i \le n\}$ has no concurrent triples, but its arrangement has $\lceil n(n-3)/3 \rceil$ triangular cells, each bounded by a triple of lines of the form

$$L((2i-1)\pi/n),\ L((2j-1)\pi/n),\ L((2k-1)\pi/n),$$



(a) (b) (c)

Figure 2.4. Another planar adversary construction for arbitrary degeneracies, following a construction of Füredi and Palásti. (a) The degenerate configuration. (b) The adversary configuration. (c) A collapsed configuration.

where $i+j+k \equiv 1$ or $2 \pmod{n}$. See Figure 2.4(b). Each of these triangles is collapsible; to collapse such a triangle, we shift each of its three defining lines by $\pi/3n$, resulting in the lines

$$L((2i-2/3)\pi/n),\ L((2j-2/3)\pi/n),\ L((2k-2/3)\pi/n),$$

if $i+j+k \equiv 1 \pmod{n}$, or

$$L((2i-4/3)\pi/n),\ L((2j-4/3)\pi/n),\ L((2k-4/3)\pi/n),$$

if $i+j+k \equiv 2 \pmod{n}$. See Figure 2.4(c). We easily verify that the collapsed triangle is degenerate, and that no other triangle changes orientation, since the sum of any other triple of defining angles changes by at most $2\pi/3n < \pi/n$.

Theorem 2.5. *Any decision tree that decides whether a set of* n *points in* $\mathbb{R}^2$ *is affinely degenerate, using only sidedness queries, must have depth at least* $\lceil n(n-3)/3 \rceil$.

Grünbaum [96] proved that a simple arrangement of n lines in the projective plane can have at most $\lfloor n(n-1)/3 \rfloor$ triangular cells if n is even, and at most $\lfloor n(n-2)/3 \rfloor$ if n is odd. Thus, we cannot hope to prove a lower bound bigger than $n^2/3 + O(n)$ using collapsible triangles.

2.3 Allowable Queries

In this section, we identify a general class of computational primitives which, if added to our model of computation, do not affect our lower bounds. In fact, even if we allow any finite number of these primitives to be performed at no cost, the number of required sidedness queries is the same. These primitives include comparisons between coordinates of input points in any number of directions, comparisons between coordinates of hyperplanes defined by d-tuples of points, and in-sphere queries.

The model of computation we consider is a restriction of the algebraic decision tree model. Recall that in this model, the result of every query is given by the sign of a multivariate *query polynomial*, evaluated at the coordinates of the input. If the sign is zero (resp. nonzero), we say that the input is *degenerate* (resp. *nondegenerate*) with respect to that query. For example, a set of points is affinely degenerate if and only if it is degenerate with respect to some sidedness query.

A *projective transformation* of $\mathbb{R}^d$ (or more properly, of the projective space $\mathbb{RP}^d$) is any map that takes hyperplanes to hyperplanes. If we represent the points of $\mathbb{R}^d$ in homogeneous coordinates, a projective transformation is equivalent to a linear transformation of $\mathbb{R}^{d+1}$.

Let $X = \{-dn, 1-dn, \ldots, n\}$ be the set of numbers described in the proof of Theorem 2.4. We call an algebraic query *allowable* if for some projective transformation ϕ, the point configuration $\phi(\omega_d(X))$ is nondegenerate with respect to that query. Our choice of terminology is justified by the following theorem.

Theorem 2.6. *Any decision tree that decides whether a set of n points in $\mathbb{R}^d$ is affinely degenerate, using only sidedness queries and a finite number of allowable queries, requires $\Omega(n^d)$ sidedness queries in the worst case. If $d \geq 3$, this lower bound holds even when the points are known in advance to be in convex position.*

Proof: Every d-dimensional projective transformation can be written as a $(d+1)\times(d+1)$ real matrix. For any polynomial q, the set of projective tranformations ϕ such that $q(\phi(\omega_d(X))) = 0$ is an algebraic variety in $\mathbb{R}^{(d+1)\times(d+1)}$. It follows that if *some* projective transformation makes $\omega_d(X)$ nondegenerate with respect to an algebraic query, then *almost every* projective transformation (*i.e.*, all but a measure-zero subset) makes $\omega_d(X)$ nondegenerate. Moreover, for any finite set of allowable queries, almost every projective transformation makes $\omega_d(X)$ nondegenerate with respect to all of them. Let ϕ be such a transformation.

Now consider the degenerate configuration $\phi(\omega_d(X))$ as a single point in the configuration space $\mathbb{R}^{dn}$. Every algebraic query induces an algebraic surface in this space, consisting of all configurations that are degenerate with respect to that query. Since algebraic surfaces are closed, if $\phi(\omega_d(X))$ is nondegenerate with respect to some finite set of alllowable queries, then for all X' in an open neighborhood of X in $\mathbb{R}^n$, the configuration $\phi(\omega(X'))$ is also nondegenerate with respect to that set of queries.

The theorem now follows from a slight modification of the proof of Theorem 2.4. Let $\varepsilon > 0$ be some sufficiently small real number. The set $\phi(\omega_d(X + \varepsilon))$ is affinely nondegenerate, but has $\Omega(n^d)$ collapsible simplices, each corresponding to a degenerate simplex in $\phi(\omega_d(X))$. No allowable query can distinguish between $\phi(\omega_d(X+\varepsilon))$ and any collapsed configuration, or even between $\phi(\omega_d(X + \varepsilon))$ and $\phi(\omega_d(X))$. □

We give below a (nonexhaustive!) list of allowable queries. We leave the proofs that these queries are in fact allowable as easy exercises.

- Comparisons between points in any fixed direction are allowable. In fact, we can allow the input points to be presorted in any finite number of fixed directions. A similar result was described by Seidel in the context of three-dimensional convex hull lower bounds [133, Theorem 5]. We emphasize that the directions in which these comparisons are made must be fixed in advance. No matter how we transform the adversary configuration, there is always *some* direction in which a point comparison can distinguish it from a collapsed configuration.

- More generally, deciding which of two points is hit first by a hyperplane rotating around a fixed $(d - 2)$-flat is allowable. We can even presort the points by their cyclic orders around any finite number of fixed $(d - 2)$-flats. If the $(d - 2)$-flat is “at infinity”, then “rotation” is just translation, and we have the previous notion of point comparison. We can interpret this type of query in dual space as a comparison between the intersections of two hyperplanes with a fixed line. Again, we emphasize that the $(d-2)$-flats must be fixed in advance.

- Sidedness queries in any fixed lower-dimensional projection are allowable. This is a natural generalization of point comparisons, which can be considered sidedness queries in a one-dimensional projection. We can even specify in advance the complete order types of the projections onto any finite number of fixed affine subspaces. (As a technical point, we would not actually include this information as part of the input, since this would drastically increase the input size. Instead, knowledge of the projected order types would be hard-wired into the algorithm.)

- "Second-order" comparisons between vertices of the dual hyperplane arrangement, in any fixed direction, are also allowable. Such a query can be interpreted in the primal space as a comparison between the intersections of two hyperplanes, each defined by a d-tuple of input points, with a fixed line. To prove that such a query is allowable, it suffices to observe that a projective transformation of the primal space induces a projective transformation of the dual space, and vice versa. Note that a second-order comparison is algebraically equivalent to a sidedness query if the two d-tuples share $d-1$ points.

- Since most projective transformations do not map spheres to spheres, in-sphere queries are allowable. Given $d+2$ points, an in-sphere query asks whether the first point lies "inside", on, or "outside" the oriented sphere determined by the other $d+1$ points. (See Chapter 4.) Similarly, in-sphere queries in any fixed lower-dimensional projection are allowable.

- Distance comparisons between pairs of points or pairs of projected points are allowable. More generally, comparing the measures of pairs of simplices of dimension less than d — for example, comparing the areas of two triangles when $d > 2$ — defined either by the original points or by any fixed projection, are allowable.

On the other hand, comparing the volumes of arbitrary simplices of *full* dimension is *not* allowable. In any projective transformation of $\omega_d(X)$, all of the degenerate simplices have the same (zero) volume. It is not possible to collapse a simplex in any adversary configuration while maintaining the order of the volumes of the other collapsible simplices.

2.4 Implications and Open Problems

A problem similar to finding degeneracies is finding the minimum measure simplex. Unfortunately, our results are *not* sufficient to improve the $\Omega(n \log n)$ lower bound on this problem. Any algorithm that finds the minimum measure simplex must be able to compare the values of arbitrary sidedness determinants, and such comparisons are not allowed in

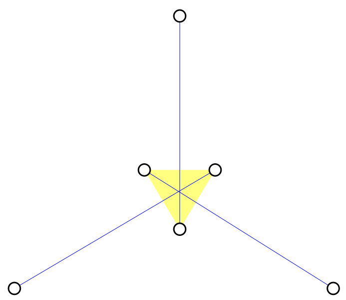


Figure 2.5. Minimum area triangles are not necessarily collapsible.

any model of computation in which our lower bounds hold. This difference may be best understood by looking at the one-dimensional case: $\Omega(n \log n)$ comparisons are required to sort a list of n numbers, but a stronger model is required to say anything about finding the closest pair. It seems impossible to apply our "collapsible simplex" argument in a model that allows comparisons between simplex volumes; a radically new idea is called for. We quickly note that the minimum measure simplex is not necessarily collapsible; see Figure 2.5.

The planar affine degeneracy problem is an example of what Gajentaan and Overmars [86] call *3SUM-hard problems*.[4] Formally, a problem is 3SUM-hard if the following problem can be reduced to it in subquadratic time:

3SUM: Given a set of real numbers, do any three sum to zero?

Thus, a subquadratic algorithm for any 3SUM-hard problem would imply a subquadratic algorithm for 3SUM, and a sufficiently powerful quadratic lower bound for 3SUM would imply similar lower bounds for every 3SUM-hard problem. Examples of 3SUM-hard problems include several degeneracy detection, separation, hidden surface removal, and motion planning problems in two and three dimensions. Gajentaan and Overmars [86] show that the planar affine degeneracy problem is 3SUM-hard, by considering a lifting from the reals to the unit cubic.[5] In fact, the restricted problem considered in Section 2.1.1 is equivalent to 3SUM since there are simple linear-time reductions in both directions. Our results imply a quadratic lower bound for 3SUM; we will present further details in Chapter 5.

Given these reductions, one might think that we have just proven that every

[4] Some earlier papers, including [73], used the more suggestive but potentially misleading term "n^2-hard" [85] (but see [20]).

[5] This observation was the initial inspiration for my "weird moment curve" argument.

3SUM-hard problem requires $\Omega(n^2)$ time. Unfortunately, this is not the case. Many of the reductions discussed in [86] require primitives that our models of computation do not allow. In these cases, one may still be able to achieve quadratic lower bounds by directly applying the techniques in this chapter. For example, consider the following problem, which Gajentaan and Overmars call SEPARATOR2: Given a set of n non-intersecting line segments in the plane, is there a line that separates the set into two non-empty subsets? Using the techniques in this paper, one can derive a quadratic lower bound for this problem, under a model that allows sidedness queries and allowable queries among the endpoints of the segments.

Even so, some 3SUM-hard problems, like the minimum-area triangle problem, cannot be solved in the models in which our techniques apply. Many of these problems already have $O(n^2)$ solutions that use primitives outside our models.

In light of these shortcomings, an obvious open question is whether our lower bound also holds in models where even more queries are allowed. Ultimately, of course, we would like a lower bound that holds in a general model of computation such as algebraic decision trees, but this seems to be completely out of reach.

The other possibility, of course, is that there is a subquadratic algorithm in some completely different model of computation. The situation may be comparable to sorting or element uniqueness—$\Omega(n \log n)$ time is required to sort using algebraic decision trees [16], but there are significantly faster sorting algorithms in integer RAM models [83, 9].

Are there faster algorithms for useful special cases? For example, a set of n points in the plane in (loosely) convex position has only n collapsible triangles, and we can easily detect colinear triples in such a set in $O(n \log n)$ time. Is there a "structure-sensitive" algorithm for detecting affine degeneracies, whose running time depends favorably on the number of collapsible triangles? Such an algorithm might be useful for solving real-world instances of other 3SUM-hard problems such as planar motion planning and hidden surface removal [86].

2.5 Out on a Limb

At the risk of annoying the reader, let me close this chapter by outlining some more evidence that the three-colinear-points problem "really" requires $\Omega(n^2)$ time. Readers looking for more theorems will be disappointed; my aim is only to provide some intuition and hopefully provoke further research.

An arrangement of *pseudolines* is a collection of curves in the plane, each homeomorphic to a straight line, such that any pair intersect transversely in exactly one point. Such an arrangement is *simple* if no three pseudolines pass through a single point. An pseudoline arrangement is *stretchable* if it can be continuously deformed into an arrangement of straight lines. A theorem of Mnëv [116] (see also [137, 97, 128]) implies that determining if a pseudoline arrangement is stretchable is NP-hard.[6]

Every known algorithm that detects degeneracies in arrangements of lines [39, 65, 68, 69] can also be used to detect degeneracies in arrangements of pseudolines. In fact, there is no known algorithmic separation of lines and pseudolines. That is, there is no known problem that can sensibly be asked about both lines and pseudolines (for example, "Sort the intersection points." or "How many edges are in the kth level?"), such that an efficient algorithm is known for the straight line version that doesn't also work for the pseudoline version. In light of Mnëv's theorem, this is perhaps not terribly surprising. (A relevant pseudo-algorithmic result is Steiger and Streinu's proof that any decision tree that sorts the intersection points of a pseudoline arrangement must have depth $\Omega(n^2 \log n)$, but the vertices of a line arrangement can be sorted by a *nonuniform* algorithm that uses only $O(n^2)$ comparisons [139]. See Chapter 5 for further discussion of nonuniform algorithms.)

There are $2^{\Theta(n^2)}$ combinatorially distinct arrangements of n pseudolines in the plane [105, 79]. (Recall from the beginning of this chapter that only $n^{\Theta(n)}$ of these are stretchable [89].) It follows immediately that determining the order type of a pseudoline arrangement requires $\Omega(n^2)$ time. Moreover, since *every* triangular cell in a pseudoline arrangement can be "flipped" to produce a new pseudoline arrangement and there are arrangements with $\Omega(n^2)$ triangular cells, $\Omega(n^2)$ sidedness queries are necessary to decide if a pseudoline arrangement is simple.

These observations suggest that deciding if a line arrangement is simple requires $\Omega(n^2)$ time because (1) deciding if a *pseudo*line arrangement is simple requires quadratic time, and (2) it is not possible for an efficient algorithm to know that its input consists of straight lines and not arbitrary pseudolines (unless, perhaps, P=NP). This suggestion is far too vague to call a "conjecture"; in particular, I haven't mentioned a specific model of com-

[6] Mnëv showed any primary semialgebraic set defined over the integers is stably homotopy-equivalent to the realization space of some rank-3 oriented matroid (*i.e.*, some pseudoline arrangement). See [18] for further implications of this remarkable result.

putation. Nevertheless, any results in this direction (even just formalizing the "conjecture") would be interesting.

[items 1–19 omitted]
20. I have no arguments to offer, my figures are my proofs.
21. The laws of nature are in harmony with me and sustain me.
22. Laugh away these facts and truths if you can.

— Carl Theodore Heisel, *The Circle Squared Beyond Refutation*, 1934

Chapter 3

Convex Hull Problems

The construction of convex hulls is perhaps the oldest and best-studied problems in computational geometry [6, 10, 11, 12, 29, 28, 30, 36, 49, 50, 91, 101, 110, 123, 130, 132, 134, 136, 142]. Over twenty years ago, Graham described an algorithm that constructs the convex hull of n points in the plane in $O(n \log n)$ time [91]. The same running time was first achieved in three dimensions by Preparata and Hong [123]. Yao [154] proved a lower bound of $\Omega(n \log n)$ on the complexity of identifying the convex hull vertices, in the quadratic decision tree model. This lower bound was later generalized to the algebraic decision tree and algebraic computation tree models by Ben-Or [16]. It follows that both Graham's scan and Preparata and Hong's algorithm are optimal in the worst case. If the output size f is also taken into account, the lower bound drops to $\Omega(n \log f)$ [101], and a number of algorithms match this bound both in the plane [101, 28, 29] and in three dimensions [50, 40].

In higher dimensions, the problem is not quite so completely solved. Seidel's "beneath-beyond" algorithm [132] constructs d-dimensional convex hulls in time $O(n^{\lceil d/2 \rceil})$. After a ten-year wait, Chazelle [36] improved the running time to $O(n^{\lfloor d/2 \rfloor})$ by derandomizing a randomized incremental algorithm of Clarkson and Shor [50]; see also [136]. Since an n-vertex polytope in $\mathbb{R}^d$ can have $\Omega(n^{\lfloor d/2 \rfloor})$ facets [87], Seidel's algorithm is optimal in even dimensions, and Chazelle's algorithm is optimal in all dimensions, in the worst case.

Several faster algorithms are known when the output size is also considered. In 1970, Chand and Kapur [30] described an algorithm that constructs convex hulls in time $O(nf)$, where f is the number of facets in the output. An algorithm of Chan, Snoeyink, and Yap [28] constructs four-dimensional hulls in time $O((n + f)\log^2 f)$, and a recent improvement by Amato and Ramos [6] constructs five-dimensional hulls in time $O((n+f)\log^3 f)$. The fastest algorithm in higher dimensions, due to Chan [29], runs in time $O(n \log f + (nf)^{1-1/(\lfloor d/2 \rfloor+1)} \operatorname{polylog} n)$; this algorithm is optimal when f is sufficiently small. For related results, see [10, 30, 49, 50, 101, 134]. There are still large gaps between these upper bounds and the lower bound $\Omega(n \log f + f)$. Avis, Bremner, and Seidel [11, 12] describe families of polytopes on which current convex hull algorithms perform quite badly, sometimes requiring exponential time (in d) even when the output size is only polynomial.

In this chapter, we consider convex hull problems for which the output size is a single integer, or even a single bit, although the convex hull itself may be large. We show that in the worst case, $\Omega(n^{\lceil d/2 \rceil - 1} + n \log n)$ sidedness queries are required to decide whether the convex hull of n points in $\mathbb{R}^d$ is simplicial, or to determine the number of convex hull facets. This matches known upper bounds when d is odd [36]. The only lower bound previously known for either of these problems is $\Omega(n \log n)$, following from the techniques of Yao [154] and Ben-Or [16]. When the dimension is allowed to vary with the input size, deciding if a convex hull is simplicial is coNP-complete [31], and counting the number of facets is #P-hard in general, and NP-hard for simplicial polytopes [61]. Our results apply when the dimension d is fixed.

Our lower bounds follow from a generalization of the previous chapter's adversary argument. We start by constructing a set whose convex hull contains a large number of independent degenerate facets. To obtain the adversary configuration, we perturb this set to eliminate the degeneracies, but in a way that the degeneracies are still "almost there". An adversary can reintroduce any one of the degenerate facets, by moving its vertices back to their original position.

3.1 Preliminaries

For a more detailed introduction to the theory of convex polytopes, we refer the reader to Ziegler [161] or Grünbaum [95].

The *convex hull* of a set of points is the smallest convex set that contains it. A *(convex) polytope* is the convex hull of a finite set of points. A hyperplane h *supports* a polytope if the polytope intersects h and lies in a closed halfspace of h. The intersection of a polytope and a supporting hyperplane is called a *face* of the polytope. The *dimension* of a face is the dimension of the smallest affine space that contains it; a face of dimension k is called a k-*face*. The faces of a polytope are also polytopes. Given a d-dimensional polytope, its $(d-1)$-faces are called *facets*, its $(d-2)$-faces are called *ridges*, its 1-faces are *edges*, and its 0-faces are *vertices*.

A polytope is *simplicial* if all its facets, and thus all its faces, are simplices. The convex hull of any affinely nondegenerate set of points is simplicial, but the converse is not true in general, as witnessed by the regular octahedron in $\mathbb{R}^3$. A polytope is *quasi-simplicial* if all of its ridges are simplices, or equivalently, if its facets are simplicial polytopes. A *degenerate facet* of a quasi-simplicial polytope is any facet that is not a simplex. Note that the vertices of a degenerate facet are also the vertices of a degenerate simplex.

3.2 The Lower Bound

Our adversary construction will consist of a set of points on the weird moment curve $\omega_d(t) = (t, t^2, \ldots, t^{d-1}, t^{d+1})$ introduced in Section 2.2. Since any collection of points on the standard moment curve is in convex position, so is any collection of points on the weird moment curve in dimensions 3 and higher. Moreover, the convex hull of any set of points on the weird moment curve is quasi-simplicial, since no d points lie on a common $(d-2)$-flat. However, degenerate facets are possible. The following lemma characterizes degenerate convex hull facets on the weird moment curve. The result is quite similar to Gale's evenness condition [87], which describes which vertices of a cyclic polytope form its facets.

Lemma 3.1. *Let X be a set of real numbers, and let $x_0, x_1, \ldots, x_d$ be elements of X whose sum is zero. The points $\omega_d(x_0), \omega_d(x_1), \ldots, \omega_d(x_d)$ are the vertices of a degenerate facet of $\mathrm{conv}(\omega_d(X))$ if and only if for any two elements $y, z \in X \setminus \{x_0, x_1, \ldots, x_d\}$, the number of elements of $\{x_0, x_1, \ldots, x_d\}$ between y and z is even.*

Proof: Let h be the hyperplane passing through the points $\omega_d(x_0), \omega_d(x_1), \ldots, \omega_d(x_d)$. Such a hyperplane exists by Lemma 2.3. Expanding the appropriate sidedness determinant, we find that an arbitrary point $\omega_d(x)$ lies above, on, or below h according to the sign of the polynomial

$$f(x) = \left(x + \sum_{i=1}^{d} x_i\right) \prod_{i=1}^{d} (x - x_i) = \prod_{i=0}^{d} (x - x_i).$$

The hyperplane h supports $\mathrm{conv}(\omega_d(X))$ if and only if $f(x)$ has the same sign for all $x \in X \setminus \{x_0, x_1, \ldots, x_d\}$.

The polynomial $f(x)$ has degree $d+1$, and vanishes at each x_i. Thus, the sign of $f(x)$ changes at each x_i. In more geometric terms, the weird moment curve crosses the hyperplane h at each of the points $\omega_d(x_i)$. It follows that $f(y)$ and $f(z)$ both have the same sign if and only if an even number of x_i's lie between y and z. $\square$

The main result of this chapter is based on the following combinatorial construction.

Lemma 3.2. *For all n and d, there is a quasi-simplicial polytope in $\mathbb{R}^d$ with $O(n)$ vertices and $\Omega(n^{\lceil d/2 \rceil - 1})$ degenerate facets.*

Proof: First consider the case when d is odd, and let $r = (d-1)/2$. Without loss of generality, we assume that n is a multiple of r. Let X denote the following set of $n + 2n/r = O(n)$ integers.

$$X = \{-rn, -rn + r, \ldots, -r;\ r, r+1, 2r, 2r+1, \ldots, n, n+1\}$$

We can specify a degenerate facet of $\omega_d(X)$ as follows. Arbitrarily choose r elements $a_1, a_2, \ldots, a_r \in X$, all positive multiples of r. Let $a_0 = -\sum_{i=1}^{r} a_i$, let $b_0 = a_0 - r$, and for all $i > 0$, let $b_i = a_i + 1$. Each a_i and b_i is a unique element of X, and no element of X lies between a_i and b_i for any i. The points $\omega_d(a_i)$ and $\omega_d(b_i)$ all lie on a single hyperplane by Lemma 2.3, since

$$\sum_{i=0}^{r} (a_i + b_i) = 2 \sum_{i=0}^{r} a_i = 0.$$

Moreover, since any pair of elements of $X \setminus \{a_i, b_i\}$ has an even number of elements of $\{a_i, b_i\}$ between them, Lemma 3.1 implies that these points are the vertices of a single facet of $\mathrm{conv}(\omega_d(X))$. There are at least $\binom{n/r}{r} = \Omega(n^r)$ ways of choosing such a degenerate facet.

When d is even, let $r = d/2 - 1$, and assume without loss of generality that n is a multiple of r. Let X be the following set of $n + 2n/r + 1 = O(n)$ integers.

$$X = \{-n - rn, -n - rn + r, \ldots, -n - r;\ r, r+1, 2r, 2r+1, \ldots, n, n+1;\ 2n\}.$$

Using similar arguments as above, we easily observe that the polytope $\mathrm{conv}(\omega_d(X))$ has $\Omega(n^r)$ degenerate facets, each of which has $\omega_d(2n)$ as a vertex. $\square$

This result is the best possible when d is odd, since an odd-dimensional n-vertex polytope has at most $O(n^{(d-1)/2})$ facets [161]. In the case where d is even, the best known upper bound is $O(n^{d/2})$, which is a factor of n bigger than the result we prove here. The convex hull of any set of n points on ω_d has at most $O(n^{\lceil d/2\rceil-1})$ degenerate facets, so the lower bound is tight for points on the weird moment curve. We conjecture that our lower bound is tight in general, up to constant factors.

Theorem 3.3. *Any decision tree that decides whether the convex hull of a set of n points in $\mathbb{R}^d$ is simplicial, using only sidedness queries, must have depth $\Omega(n^{\lceil d/2\rceil-1} + n\log n)$.*

Proof: Let X be the set of numbers described in the proof of Lemma 3.2, and let $X' = X + 1/(2d+2)$. Initially, the adversary presents the set of points $\omega_d(X')$. Since $\sum_{i=0}^{d} x_i'$ is always a half-integer, this point set is affinely nondegenerate, so its convex hull is simplicial.

It suffices to consider the case where d is odd. Let $r = (d-1)/2$. Choose $a_0', b_0', a_1', b_1', \dots, a_r', b_r' \in X'$ so that $\sum_{i=0}^{r}(a_i' + b_i') = 1/2$ and no other elements of X' lie between a_i' and b_i' for any i. The corresponding points $\omega_d(a_i'), \omega_d(b_i')$ form a collapsible simplex. To collapse it, the adversary simply moves the points back to their original positions in $\omega_d(X)$. Lemmas 2.3 and 3.2 imply that the collapsed simplex forms a degenerate facet of the new convex hull. Since the sum of any other $(d+1)$-tuple changes by at most $1/2 - 1/(2d+2)$, no other simplex changes orientation. In other words, the only way for an algorithm to distinguish between the original configuration and the collapsed configuration is to perform a sidedness query on the collapsible simplex.

Thus, if an algorithm does not perform a separate sidedness query on every collapsible simplex, then the adversary can introduce a degenerate facet that the algorithm cannot detect. There are $\Omega(n^{\lceil d/2\rceil-1})$ collapsible simplices, one for each degenerate facet in $\operatorname{conv}(\omega_d(X))$.

Finally, the $n\log n$ term follows from the algebraic decision tree lower bound of Ben-Or [16]. □

A three-dimensional version of our construction is illustrated in Figure 3.1. (See also the proof of Theorem 4.3!)

Our lower bound matches known upper bounds when d is odd [36]. We emphasize that if the points are known *in advance* to lie on the weird moment curve, this problem can be solved in $O(n^{\lceil d/4\rceil})$ time if $\lceil d/2\rceil$ is odd, and in $O(n^{\lceil d/4\rceil}\log n)$ time if $\lceil d/2\rceil$ is even,

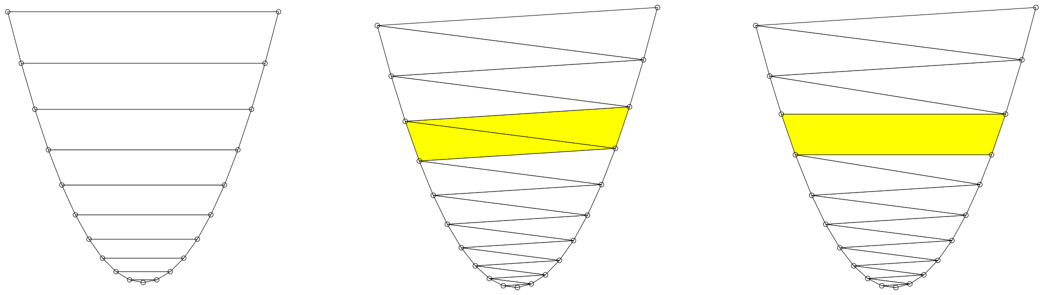


Figure 3.1. The convex hull adversary construction in three dimensions. Bottom views of (a) a quasi-simplicial polytope with $\Omega(n)$ degenerate facets, (b) the simplicial adversary polytope with one collapsible simplex highlighted, and (c) the corresponding collapsed polytope.

by an algorithm that uses more complicated queries, similar to the algorithm described in [73].

The convex hull of the adversary configuration $\omega_d(X')$ has $\lceil d/2 \rceil - 1$ more facets than the convex hull of any collapsed configuration. Thus, we immediately have the following lower bound.

Theorem 3.4. *Any decision tree that computes the number of convex hull facets of a set of n points in $\mathbb{R}^d$, using only sidedness queries, must have depth $\Omega(n^{\lceil d/2 \rceil - 1} + n \log n)$.*

A simple modification of our argument implies the following "output-sensitive" version of our lower bound.

Theorem 3.5. *Any decision tree that decides whether the convex hull of a set of n points in $\mathbb{R}^d$ is simplicial or computes the number of convex hull facets, using only sidedness queries, must have depth $\Omega(f)$ when d is odd, and $\Omega(f^{1-2/d})$ when d is even, where f is the number of faces of the convex hull.*

Proof: We construct a modified degenerate polytope as follows. We start by constructing a degenerate polytope with f faces, exactly as described in the proof of Lemma 3.2. When d is odd, this polytope is the convex hull of $\Theta(f^{2/(d-1)})$ points on the wired moment curve, and has $\Omega(f)$ degenerate facets. When d is even, the polytope is the convex hull of $\Theta(f^{2/d})$ points and has $\Omega(f^{1-2/d})$ degenerate facets.

By introducing a new vertex extremely close to the relative interior of any facet of a simplicial polytope, we can split that facet into d smaller facets. Each such split increases the number of polytope faces by $2^d - 2$. To bring the number of vertices of our adversary polytope up to n, we choose some facet and repeatedly split it in this fashion, being careful not to introduce any new degenerate simplices. The augmented polytope has at most $f + (2^d - 2)n = O(f)$ faces.

To get a modified *adversary* polytope, we slide the original vertices of the degenerate polytope along the weird moment curve, just enough to remove the degeneracies, leaving the new vertices in place. Each of the degenerate facets becomes a collapsible simplex. As long as we don't slide the vertices too far, collapsing a simplex will not change the orientation of any simplex involving a new vertex. (In effect, we are treating sidedness queries involving new vertices as "allowable" queries; see below.) The lower bound now follows from the usual adversary argument. □

Finally, we note that our convex hull lower bounds still hold if we augment our model of computation with extra queries as in Section 2.3. Let X be the set of numbers described in the proof of Lemma 3.2. We call an algebraic query *allowable* if for some projective transformation ϕ, the configuration $\phi(\omega_d(X))$ is nondegenerate with respect to that query.

Theorem 3.6. *Any decision tree that decides whether the convex hull of a set of n points in $\mathbb{R}^d$ is simplicial, using only sidedness queries and a finite number of allowable queries, requires $\Omega(n^{\lceil d/2 \rceil - 1} + n \log n)$ sidedness queries in the worst case.*

The proof of this theorem follows the proof of Theorem 2.6 almost exactly. The only difference is that we must consider only projective transformations that preserve the convex hull structure of $\omega_d(X)$. Alternately, we can use Stolfi's two-sided projective model, in which projective maps preserve (or reverse) the orientation of every simplex in $\mathbb{R}^d$, and thus always preserve the combinatorial structure of convex hulls; see [140, Chapter 14].

3.3 Real Convex Hull Algorithms

A large number of convex hull algorithms rely (or can be made to rely) exclusively on sidedness queries. These include the "gift-wrapping" algorithms of Chand and Kapur [30] and Swart [142], the "beneath-beyond" method of Seidel [132], Clarkson and Shor's [50] and Seidel's [136] randomized incremental algorithms, Chazelle's worst-case optimal algorithm [36], and the recursive partial-order algorithm of Clarkson [49].

Seidel's "shelling" algorithm [134] and the space-efficient gift-wrapping algorithms of Avis and Fukuda[1] [10] and Rote [130] require only sidedness queries and "second-order" coordinate comparisons between vertices of the dual hyperplane arrangment. Matoušek [110] and Chan [29] improve the running times of these algorithms (in an output-sensitive sense), by finding the extreme points more quickly. Clarkson [49] describes a similar improvement to a randomized incremental algorithm. Since every point in our adversary configuration is extreme, our lower bound still holds even if the extremity of a point can be decided for free. We are not suggesting that the computational primitives used by these algorithms cannot be used to break our lower bounds; only that the ways in which these primitives are currently applied are inherently limited.

Chan [29] describes an improvement to the gift-wrapping algorithm, using ray shooting data structures of Agarwal and Matoušek [4] and Matoušek and Schwarzkopf [108] to speed up the pivoting step. In each pivoting step, the gift-wrapping algorithm finds a new facet containing a given ridge of the convex hull. In the dual, this is equivalent to shooting a ray from a vertex of the dual polytope along one of its outgoing edges. The dual vertex that the ray hits corresponds in the primal to the new facet. A single pivoting step tells us the orientation of $n - d$ simplices, all of which share the d vertices of the new facet. However, at most one of these simplices can be collapsible, since two collapsible simplices share at most $d/2$ vertices. Thus, even if we allow a pivoting step to be performed in constant time, our lower bound still holds.

There are a few convex hull algorithms which seem to fall outside our framework, most notably the divide-prune-and-conquer algorithm of Chan, Snoeyink, and Yap [28] and its improvement by Amato and Ramos [6]. The two-dimensional version of their algorithm uses sidedness queries, along with first-, second-, and even *third*-order comparisons; higher-dimensional versions use even more complex primitives.

3.4 Open Problems

Several open problems remain to be answered. While our lower bounds match existing upper bounds in odd dimensions, there is still a gap when the dimension is even. A first step in improving our lower bounds might be to improve the combinatorial bounds in Lemma 3.2. Is there a quasi-simplicial 4-polytope with n vertices and $\Omega(n^2)$ degenerate

[1] at least if Bland's pivoting rule is used

facets? Simple variations on the weird moment curve will not suffice, since an "evenness condition" like Lemma 3.1 always forces the number of degenerate facets to be linear. Arguments based on merging facets of cyclic or product polytopes also fail, as do variations of Amenta and Ziegler's deformed products [7, 8]. I conjecture that the answer is **no**, even for polyhedral 3-spheres.

A common application of convex hull algorithms is the construction of Delaunay triangulations and Voronoi diagrams. Are $\Omega(n^{\lceil d/2 \rceil})$ in-sphere queries required to decide if the Delaunay triangulation is simplicial (*i.e.*, really a triangulation)? Again, a first step might be to construct a three- or four-dimensional Delaunay triangulation with $\Omega(n^2)$ independent degenerate features. I conjecture, however, that no such triangulations exist.

Another similar problem is deciding, given a set of points, which ones are vertices of the set's convex hull. This problem can be decided in $O(n^2)$ time (using only sidedness queries!) by invoking a linear programming algorithm once for each point [48, 109, 113, 136]. This upper bound can be improved to $O(n^{2\lfloor d/2 \rfloor/(\lfloor d/2 \rfloor - 1)} \operatorname{polylog} n)$ using an algorithm due to Chan [29]. Except for the polylogarithmic term, this algorithm is almost certainly optimal. It seems unlikely that a collapsible simplex argument could be used to imply a reasonable lower bound for this problem. Perhaps the techniques we describe in Part II are more applicable.

The suggestions described at the end of the previous chapter apply to convex hull problems as well. Richter-Gebert's universality theorem for 4-polytopes [127, 129] implies that it is NP-hard to decide if a given combinatorial 3-sphere is realizable as a 4-polytope. (In contrast, Steinitz' Theorem [161, Chapter 4] implies that every 2-sphere is realizable as a 3-polytope.) Perhaps deciding if a 4-polytope is simplicial requires $\Omega(n^2)$ time because (1) deciding if a combinatorial 3-sphere is simplicial requires quadratic time, and (2) it is not possible for an efficient algorithm to know that its input is a 4-polytope and not an arbitrary combinatorial 3-sphere. Again, this suggestion needs to be formalized before there is any hope of proving or disproving it.

Everything should be made as simple as possible,
but no simpler.

— Albert Einstein

Chapter 4

Spherical Degeneracies

The *spherical degeneracy problem* asks, given n points in $\mathbb{R}^d$, if any $d+2$ lie on the same sphere. This problem can be transformed into the affine degeneracy problem one dimension higher by projecting the input vertically onto the paraboloid $x_{d+1} = x_1^2 + \cdots + x_d^2$. The images of cospherical points in $\mathbb{R}^d$ under this projection lie on a single hyperplane in $\mathbb{R}^{d+1}$. Furthermore, if the point q lies inside (resp. outside) the sphere defined by $d+1$ points $p_0, \ldots, p_d$ in $\mathbb{R}^d$, then the image of q lies below (resp. above) the hyperplane in $\mathbb{R}^{d+1}$ defined by the images of $p_0, \ldots, p_d$ [62]. Sidedness queries on the lifted point set are thus equivalent to *insphere queries* in the original d-dimensional point set. Two-dimensional insphere queries are also called *incircle queries*. Algebraically, the result of an insphere query is given by the sign of the following determinant.

$$\begin{vmatrix} 1 & p_{01} & p_{02} & \cdots & p_{0d} & \sum_i p_{0i}^2 \\ 1 & p_{11} & p_{12} & \cdots & p_{1d} & \sum_i p_{1i}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & p_{d1} & p_{d2} & \cdots & p_{dd} & \sum_i p_{di}^2 \\ 1 & q_1 & q_2 & \cdots & q_d & \sum_i q_i^2 \end{vmatrix}$$

A special case of the spherical degeneracy problem ignores $(d+2)$-tuples that lie on spheres of infinite radius (*i.e.*, hyperplanes). We refer to any $(d+2)$-tuple that lies on a sphere of finite radius as a *proper* spherical degeneracy.

In this chapter, we show that $\Omega(n^3)$ incircle queries are required to detect circular degeneracies in the plane, and $\Omega(n^{d+1})$ insphere queries are required to detect proper spherical degeneracies in $\mathbb{R}^d$. Both lower bounds are tight [65, 68, 69]. Like our previous results, the lower bounds in this chapter are based on adversary arguments using "collapsible tuples".

4.1 Circular Degeneracies

Theorem 4.1. *Any decision tree algorithm that detects proper circular degeneracies, using only incircle queries, must have depth* $\Omega(n^3)$.

Proof: The adversary presents the following set of points:

$$S \triangleq \bigcup_{i=1}^{n/6} \left\{ (2^{5i+2}, 0), (2^{5i-2}, 0), (0, 2^{5i}) \right\} \cup \bigcup_{i=1}^{n/2} \left\{ (0, 2^{5(i-n/6)+1}) \right\}$$

The set consists of four subsets of points, two contained in each positive coordinate axis. We easily verify that this set contains no proper circular degeneracies, using the fact that four points $(a, 0)$, $(b, 0)$, $(0, c)$, and $(0, d)$ are cocircular if and only if $ab = cd$.

For each $1 \leq i, j, k \leq n/6$, the following points are "almost" cocircular.

$$(2^{5i+2}, 0), (2^{5j-2}, 0), (0, 2^{5k}), (0, 2^{5(i+j-k)+1})$$

Each such set of points is a *collapsible* 4*-tuple*. The adversary can collapse any such tuple by changing the four points to the following.

$$(2^{5i+3/2}, 0), (2^{5j-3/2}, 0), (0, 2^{5k-1/2}), (0, 2^{5(i+j-k)+1/2})$$

We easily verify that this change does not introduce any other new circular degeneracies or change the result of any other incircle query. There are $n^3/216 = \Omega(n^3)$ collapsible 4-tuples, each of which must be checked by the algorithm. □

Since collapsing a 4-tuple preserves both the coordinate orders of the points and their order type, we immediately have the following stronger theorem.

Theorem 4.2. *Any decision tree algorithm that detects proper circular degeneracies, using only incircle queries, sidedness queries, and coordinate comparisons, must perform* $\Omega(n^3)$ *incircle queries in the worst case.*

A lower bound for the general problem follows from a very simple argument, similar to the weird moment curve argument used throughout the last two chapters. In this case, the "weird" curve we need is a parabola.

Theorem 4.3. *Any decision tree that decides whether n points in $\mathbb{R}^2$ is circularly degenerate, using only incircle queries, must have depth $\Omega(n^3)$.*

Proof: Four points $(a, a^2), (b, b^2), (c, c^2), (d, d^2)$ on the unit parabola are cocircular if and only if $a+b+c+d = 0$. (Indeed, the paraboloid lifting function $(x, y) \mapsto (x, y, x^2+y^2)$ maps the unit parabola to a skewed three-dimensional weird moment curve; see Figure 3.1!) Let X be the set of integers from $-n$ to n. There are clearly $\Theta(n^3)$ 4-tuples in X whose sums are zero. The adversary presents a set of points on the unit parabola with x-coordinates taken from the set $X + 1/8$. This set is nondegenerate and has $\Omega(n^3)$ collapsible 4-tuples. $\square$

We can extend the model of computation in a similar fashion as in Section 2.3, but with a different set of primitives. A *linear fractional transformation* of the plane (or more formally, of the Riemann sphere $\mathbb{CP}^1$) is any transformation that maps circles to circles. If we represent the points of $\mathbb{R}^2$ in complex homogeneous coordinates — representing $(x, y) \in \mathbb{R}^2$ by any complex multiple of $(1 + 0i, x + yi) \in \mathbb{C}^2$ — then a linear fractional transformation is equivalent to a linear transformation of $\mathbb{C}^2$.

We say that a query is *circularly allowable* if some linear fractional transformation of the set (X, X^2) is nondegenerate with respect to that query, where $X = \{-n, 1-n, \ldots, n\}$ is the set of numbers described in the proof of Theorem 4.3. Circularly allowable queries include first- and second-order coordinate comparisons and sidedness queries, but do not include comparisons between arbitrary incircle determinants.

Arguments similar to those in Section 2.3 give us the following theorem.

Theorem 4.4. *Any decision tree that decides whether n points in $\mathbb{R}^2$ is circularly degenerate, using only incircle queries and a finite number of circularly allowable queries, requires $\Omega(n^3)$ incircle queries in the worst case.*

4.2 Proper Spherical Degeneracies

In order to extend Theorem 4.1 to the d-dimensional case, we exhibit a set S of $O(n)$ points in $\mathbb{R}^d$ that contains $\Omega(n^{d+1})$ *collapsible* $(d+2)$*-tuples*: sets of $d+2$ non-cospherical points in S that can be moved so that they become cospherical, without changing the result of any other insphere query. The following construction is primarily due to Raimund Seidel [73].

The point set S in question is the union of $d+1$ smaller sets, $S_1 \cup \cdots \cup S_d \cup D$, where each S_i consists of $n/2$ even integer points on the positive x_i-axis, and D consists of about $(d+1)n/2$ "odd" points on the main diagonal $(t, \ldots, t)$. At the risk of confusing the reader, we let each subscripted variable t_i refer simultaneously to a point on the x_i-axis and that point's non-zero coordinate. Similarly, each unsubscripted variable t refers simultaneously to a point on the main diagonal and the value of all its coordinates.

To make our construction precise, the sets S_i include points t_i such that t_i is even and

$$a_i < t_i \le a_i + n,$$

where $a_1 = 0$, a_i is large for all $1 < i < d$ (say $a_i = n^3 + in$), and a_d is huge (say $a_d = 2^n$). The set D includes points t such that dt is odd and

$$A < dt \le A + (d+1)n,$$

where $A = \sum_{i=1}^{d} a_i$.

Lemma 4.5. *The set S contains no proper spherical degeneracies.*

Proof: For all $1 \le i \le d$, let t_i and t_i' be two distinct points in S_i, and let t and t' be two distinct points in D. Note that with our choice of values for a_i we have the following bounds.

$$\begin{array}{rcccl} -(d+1)n & < & t_1 + \cdots + t_d - dt & < & dn \\ 1/n & < & \dfrac{1}{t_1} + \cdots + \dfrac{1}{t_d} - \dfrac{1}{t} & < & 1 \\ \dfrac{d-2}{n^3} - o\left(\dfrac{1}{n^3}\right) & < & \dfrac{1}{t_2} + \cdots + \dfrac{1}{t_d} - \dfrac{1}{t} & < & \dfrac{d-2}{n^3} \end{array}$$

By examining the appropriate insphere determinants, we find that the cosphericity of any set of d+2 points from S is expressed by the vanishing of one of the following algebraic expressions.

- Two points from the x_i-axis, one from each of the other axes, and one from the main diagonal:

$$t_1 + \cdots + t_d - dt + t_i t_i' \left(\frac{1}{t_1} + \cdots + \frac{1}{t_d} - \frac{1}{t} \right) \tag{4.1}$$

- Two points from the main diagonal, and one from each axis:

$$t_1 + \cdots + t_d - dt + dtt' \left(\frac{1}{t_1} + \cdots + \frac{1}{t_d} - \frac{1}{t} \right) \tag{4.2}$$

- Two points from the x_i axis, two points from the x_j axis, and $d-2$ points elsewhere:

$$t_i t_i' - t_j t_j' \tag{4.3}$$

- Two points from the x_i axis, two points from the main diagonal, and $d-2$ points elsewhere:

$$t_i t_i' - dtt' \tag{4.4}$$

With t_i, t_i', t, t' chosen in the indicated ranges and with the indicated parities, expression (4.1) never vanishes, since the last term dominates when $i > 1$, and the whole expression differs from an odd integer by less than d/n when $i = 1$. Expression (4.2) never vanishes, since the last term always dominates. Expression (4.3) never vanishes, since the x_i-range and the x_j-range are disjoint. Finally, expression (4.4) never vanishes, since the second term dominates when $i < d$, and the first term dominates when $i = d$. □

Lemma 4.6. *The set S contains $\Omega(n^{d+1})$ collapsible $(d+2)$-tuples.*

Proof: For any choice of two distinct points t_1, t_1' from S_1 and one point t_i from each of the other S_i, we can choose the point from D with all coordinates equal to $(t_1 + \cdots + t_d + t_1' - 1)/d$, so that these points form a collapsible $(d+2)$-tuple. To collapse the tuple, the adversary decreases the non-zero coordinates of the axis points by $1/(2d+2)$ and increases each coordinate of the main diagonal point by just under $1/2d + 1/n$. □

Our previous adversary argument immediately implies the following lower bound.

Theorem 4.7. *Any decision tree algorithm that detects proper spherical degeneracies in $\mathbb{R}^d$, using only insphere queries, must have depth $\Omega(n^{d+1})$.*

As we did in the planar case, we can extend this lower bound to allow additional computational primitives.

Theorem 4.8. *Any decision tree algorithm that detects proper spherical degeneracies in $\mathbb{R}^d$, using only insphere queries, sidedness queries, and coordinate comparisons, must perform $\Omega(n^{d+1})$ insphere queries in the worst case.*

Proof: It suffices to show that collapsing a $(d+2)$-tuple does not change the result of any coordinate comparison or sidedness query. Coordinate comparisons don't change, since the order of the points is preserved within each subset, and the range of coordinates for the points on the main diagonal is disjoint from the range of coordinates for the points on any coordinate axis.

By examining the appropriate sidedness determinants, we find simple algebraic expressions giving the orientation of any simplex in S, similar to the expressions (4.1)–(4.4) describing cosphericity. There are only three nontrivial cases.

- Two points on one axis, and no points on one axis or the main diagonal:

$$t_i - t_i' \tag{4.5}$$

- Two points on the main diagonal, and no points on one axis:

$$t - t' \tag{4.6}$$

- One point on each axis, and one on the main diagonal:

$$\frac{1}{t_1} + \cdots + \frac{1}{t_d} - \frac{1}{t} \tag{4.7}$$

In every other case, the simplex is always degenerate.

In the first and second cases, sidedness queries reduce to coordinate comparisons. In the original configuration S, expression (4.7) is positive, and collapsing a tuple only makes it bigger, since each t_i is decreasing and t is increasing. Thus, no simplex in S changes orientation. □

4.3 Open Problems

We conjecture that $\Omega(n^{d+1})$ insphere queries are required to detect *arbitrary* spherical degeneracies. (I claimed this lower bound in [73], but my "proof" was incorrect.) A proof of this conjecture would follow immediately from the construction of a set of numbers having $\Omega(n^{d+1})$ $(d+2)$-tuples in the zeroset of a certain symmetric polynomial, by applying the same "sliding adversary" argument used to prove many of our previous lower bounds. For example, in three dimensions, we need $\Omega(n^4)$ 5-tuples in the zeroset of the polynomial

$$1 + \sum_{1 \le i \le j \le 5} t_i t_j.$$

Unlike all our previous constructions, the adversary set we used to prove Theorem 4.7 is not obtained by perturbing a highly degenerate point configuration. Is there a set of n points in $\mathbb{R}^d$ with $\Omega(n^{d+1})$ independent spherical degeneracies, for any $d \ge 3$? Such a set might lead to a lower bound for the general spherical degeneracy problem, and it might also allow us to define a general class of "spherically allowable" queries, strengthening Theorem 4.8.

"... In that blessed region of Four Dimensions, shall we linger on the threshold of the Fifth, and not enter therein? Ah, no! Let us rather resolve that our ambition shall soar with our corporal ascent. Then, yielding to our intellectual onset, the gates of the Sixth Dimension shall fly open; after that a Seventh, and then an Eighth —"

How long I should have continued I know not. In vain did the Sphere, in his voice of thunder, reiterate his command of silence, and threaten me with the direst penalties if I persisted.

— Edwin Abbott, *Flatland*, 1884

Chapter 5

Linear Satisfiability Problems

Many computational decision problems, particularly in computational geometry, can be reduced to questions of the following form: For some fixed multivariate polynomial ϕ, given a set of n real numbers, is any subset in the zero-set of ϕ? Examples include element uniqueness ($\phi = x - y$) and 3SUM ($\phi = x + y + z$). Higher dimensional examples include the affine and spherical degeneracy problems considered in Chapters 2 and 4, and Hopcroft's point-line incidence problem, which we will consider in Chapter 6.

In this chapter, we develop general techniques for proving lower bounds on the complexity of deciding problems of this type. In particular, we examine *linear satisfiability problems*, in which the polynomial ϕ is linear. Any r-variable linear satisfiability problem can be decided in $O(n^{(r+1)/2})$ time when r is odd, or $O(n^{r/2} \log n)$ time when r is even. These are the best known upper bounds; the algorithms that achieve them were described in Section 2.1.3 (with $r = d + 1$).

We consider these problems under two models of computation, both restrictions of the linear decision tree model. In the *direct query* model, each decision is based on the sign of an assignment to ϕ by r of the input variables. In the r-*linear decision tree* model, each decision is based on the sign of an arbitrary affine combination of at most r input variables. We show that in these models, any algorithm that solves any r-variable linear satisfiability problem must perform $\Omega(n^{\lceil r/2 \rceil})$ direct queries in the worst case. This matches known upper bounds when r is odd, and is within a logarithmic factor when r is even. Moreover, results of Fredman [80] establish the existence of *nonuniform* algorithms whose running times match our lower bounds exactly.

The adversary arguments we use to establish lower bounds for these require two new tricks. The first trick is to allow our adversary configurations to contain formal infinitesimals, instead of just real numbers. Tarski's Transfer Principle implies that for any algorithm, if there is a hard configuration with infinitesimals, then a corresponding real configuration exists with the same properties. Previously, Dietzfelbinger and Maass [56, 55] used a similar technique to prove lower bounds, using "inaccessible" numbers, or numbers having "different orders of magnitude". Unlike their technique, using infinitesimals makes it possible, and indeed sufficient, to derive a *single* adversary configuration for any problem, rather than explicitly constructing a different configuration for every algorithm.

The second trick is allowing our adversary configurations to be degenerate. That is, both the original configuration and the collapsed configuration contain tuples in the zero-set of ϕ. We show that such a configuration can always be perturbed into general position, so that the new configuration has just as many collapsible tuples as the original.

An $\Omega(n \log n)$ lower bound for any linear satisfiability problem follows easily from techniques of Dobkin and Lipton in the linear decision tree model [58], Steele and Yao in the algebraic decision tree model [138], and Ben-Or in the algebraic computation tree model [16]. The first better lower bound is due to Fredman [80], who proved an $\Omega(n^2)$ lower bound on the number of comparisons required to detect duplicate elements in the Minkowski sum $X + Y$ of two sets of real numbers; his proof relies on a simple adversary argument. Fredman's result was generalized by Dietzfelbinger [55], who derived an $\Omega(n^{r/2})$ lower bound on the depth of any comparison tree algorithm that determines, given a set of n reals, whether any two subsets of size $r/2$ have the same sum. In our terminology, he proves a lower bound for the specific r-variable linear satisfiability problem with

$$\phi = \sum_{i=1}^{r/2} t_i - \sum_{i=1}^{r/2} t_{i+r/2}$$

in the direct query model, for all even r. Dietzfelbinger's results imply a lower bound in the more general r-linear decision tree model as well.

Our lower bounds should be compared with the following result of Meyer auf der Heide [114]: For any fixed n, there exists a linear decision tree of depth $O(n^4 \log n)$ that decides the n-dimensional knapsack problem. This nonuniform algorithm can be adapted to solve any of the linear satisfiability problems we consider, in the same amount of time [56]. Thus, there is no hope of proving lower bounds bigger than $\Omega(n^4 \log n)$ for any of these problems in the linear decision tree model. We reiterate that our lower bounds apply only to linear decision trees where the number of terms in any query is bounded by a constant.

5.1 Preliminaries

An *ordered field* is a field with a strict linear ordering $<$ compatible with the field operations, or more abstractly, a field in which the equation $\sum_i a_1^2 = 0$ has no nontrivial solutions. A *real closed field* is an ordered field, no proper algebraic extension of which is also an ordered field. The *real closure* $\widetilde{K}$ of an ordered field K is the smallest real closed field that contains it. We refer the interested reader to [22] or [124] for further details and more formal definitions, and to [25, 26] for previous algorithmic applications of real closed fields.

An *elementary formula*[1] is a finite quantified boolean formula, each of whose clauses is a multivariate polynomial inequality with real coefficients. An elementary formula *holds in* an ordered field K if and only if the formula has no free variables, and the formula is true if we interpret each variable as an element of K and addition and multiplication as field operations in K.

The following principle was originally proven by Tarski [146], in a slightly different form. See [22] for a more recent proof.

The Transfer Principle: *Let $\widetilde{K}$ and $\widetilde{K'}$ be two real closed fields. An elementary formula holds in $\widetilde{K}$ if and only if it holds in $\widetilde{K'}$.*

In particular, this implies that if an elementary formula holds in *any* real closed field, then it must hold in the reals.

For any ordered field K, we let $K(\varepsilon)$ denote the ordered field of rational functions in ε with coefficients in K, where ε is positive but less than every positive element of K. In this case, we say that ε is *infinitesimal in* K. We use towers of such field extensions. In such an extension, the order of the infinitesimals is specified by the description of the field. For example, in the ordered field $\mathbb{R}(\varepsilon_1, \varepsilon_2, \varepsilon_3)$, ε_1 is infinitesimal in the reals, ε_2 is infinitesimal in $\mathbb{R}(\varepsilon_1)$, and ε_3 is infinitesimal in $\mathbb{R}(\varepsilon_1, \varepsilon_2)$. An important property of such a field (in fact, the only property we really need) is that the sign of any element $a_0 + a_1\varepsilon_1 + a_2\varepsilon_2 + a_3\varepsilon_3 \in \mathbb{R}(\varepsilon_1, \varepsilon_2, \varepsilon_3)$, where each of the coefficients a_i is real, is given by the sign of the first nonzero coefficient; in particular, the element is zero if and only if every

[1] or more formally, a formula in the first-order language of ordered fields with parameters in $\mathbb{R}$ [22]

a_i is zero. Infinitesimals have been used extensively in perturbation techniques [67, 71, 160], in algorithms dealing with real semialgebraic sets [25, 26], and in at least one other lower bound argument [93].

Let us now formally define our model of computation. Recall that a *linear decision tree* is a ternary tree in which each interior node v in the tree is labeled with a linear *query polynomial* $q_v \in \mathbb{R}[t_1, \ldots, t_n]$ and its branches labeled -1, 0, and $+1$. Each leaf is labeled with some value; for our purposes, these values are all either "true" or "false". We compute with such a tree as follows. Given an input $X \in \mathbb{R}^n$, the sign of $q_v(X)$ is computed, where v is the root of the tree, and the computation proceeds recursively in the appropriate subtree. When a leaf is reached, its label is returned as the output of the algorithm. (Compare [58, 138].) An r-*linear* decision tree is a linear decision tree, each of whose query polynomials has at most r terms.

Let K be any ordered field extension of the reals. Since K is ordered, and since any real polynomial can be thought of as a function from K to K, it is reasonable to talk about the behavior of any linear decision tree given input from K^n. (We emphasize that query polynomials always have real coefficients, even when we consider more general inputs.) For any ordered field K, we will refer to the space K^n of possible inputs as the *configuration space*, and its individual elements as *configurations*.

5.2 The Lower Bound

In this section, we prove the following lower bound.

Theorem 5.1. *Any r-linear decision tree that decides an r-variable linear satisfiability problem must have depth $\Omega(n^{\lceil r/2 \rceil})$.*

Throughout this section, let ϕ denote a fixed linear expression in r variables. We say that an r-tuple is *degenerate* if it lies in the zero-set of ϕ, and that a configuration X is degenerate if it contains any degenerate r-tuples. For any configuration X, we call an r-tuple of elements of X *collapsible* if the following properties are satisfied.

(1) The tuple is nondegenerate.

(2) There exists another *collapsed* configuration $\check{X}$, such that the corresponding tuple in $\check{X}$ is degenerate, but the sign of every other real affine combination of r or fewer elements is the same for both configurations.

In other words, the only way for an r-linear decision tree to distinguish between X and $\check{X}$ is to perform a direct query on the tuple. Our usual adversary argument implies that the number of collapsible tuples in any nondegenerate configuration is a lower bound on the depth of any r-linear decision tree.

Unfortunately, this approach seems to be doomed from the start. For *any* two sets X and Y of real numbers, there are an infinite number of query polynomials that are positive at X and negative at Y. It follows that real configurations cannot contain collapsible tuples. Moreover, for any set X of n real numbers, there is an algorithm which requires only n queries to decide whether X satisfies any fixed linear satisfiability problem. Thus, no single real configuration is hard for every algorithm.

To get around this problem, we allow our adversary configurations to contain elements of an ordered field of the form $K = \mathbb{R}(\varepsilon_1, \ldots, \varepsilon_m)$. Allowing the adversary to use infinitesimals lets us construct a configuration with several collapsible tuples (Lemma 5.3), even though such configurations are impossible if we restrict ourselves to the reals.

The algorithms we consider are only required to behave correctly when they are given real input. Therefore, before applying our adversary argument, we must first eliminate the infinitesimals. The second step in our proof (Lemma 5.4) is to derive, for each r-linear decision tree, a corresponding real configuration with several *relatively* collapsible tuples (defined below). This step follows from our infinitesimal construction by a straightforward application of Tarski's Transfer Principle.

Finally, the adversary configurations we construct in the first step (and by implication, the real configurations we get by invoking the Transfer Principle) contain several r-tuples in the zeroset of ϕ. Thus, the collapsible tuples do not immediately give us the lower bound, since both the original configuration and the collapsed configuration are degenerate. In the final step of the proof (Lemma 5.5), we show that these degenerate configurations can be perturbed into general position. The lower bound then follows from our usual adversary argument.

5.2.1 The Infinitesimal Adversary Configuration

Our construction relies on an integer matrix M satisfying the following lemma.

Lemma 5.2. *There exists an $r \times \lfloor r/2 \rfloor$ integer matrix M satisfying the following two conditions.*

(1) There are $\Omega(n^{\lceil r/2\rceil})$ vectors $v \in \{1, 2, \ldots, n\}^r$ such that $M^\top v = 0$.

(2) Every set of $\lfloor r/2 \rfloor$ rows of M forms a nonsingular matrix.

Proof: Let $M = (m_{ij})$ be the $r \times \lfloor r/2 \rfloor$ integer matrix whose first $\lceil r/2 \rceil$ rows form a Vandermonde matrix with $m_{ij} = i^{j-1}$, and whose last $\lfloor r/2 \rfloor$ rows form a negative identity matrix. We claim that this matrix satisfies conditions (1) and (2).

We construct a vector $v = (v_1, v_2, \ldots, v_r) \in \{1, 2, \ldots, n\}^r$ such that $M^\top v = 0$ as follows. Let $m_{\max} = \lceil r/2 \rceil^{\lfloor r/2 \rfloor - 1}$ denote the largest element in M. Fix the first $\lceil r/2 \rceil$ coordinates of v arbitrarily in the range

$$1 \leq v_i \leq \left\lfloor \frac{n}{\lceil r/2 \rceil m_{\max}} \right\rfloor .$$

Now assign the following values to the remaining $\lfloor r/2 \rfloor$ coordinates:

$$v_j = \sum_{i=1}^{\lceil r/2 \rceil} m_{i, j-\lceil r/2 \rceil} v_i .$$

Since each m_{ij} is a positive integer, the v_j are all positive integers in the range $\lceil r/2 \rceil \leq v_j \leq n$. We easily verify that $M^\top v = 0$. There are

$$\left\lfloor \frac{n}{\lceil r/2 \rceil m_{\max}} \right\rfloor^{\lceil r/2 \rceil} = \left\lfloor \frac{n}{\lceil r/2 \rceil^{\lfloor r/2 \rfloor}} \right\rfloor^{\lceil r/2 \rceil} = \Omega(n^{\lceil r/2 \rceil})$$

different ways to choose the vector v. Thus, M satisfies condition (1).

Let M' be a matrix consisting of $\lfloor r/2 \rfloor$ arbitrary rows of M. Using elementary row and column operations, we can write

$$M' = W \begin{bmatrix} V & 0 \\ 0 & -I \end{bmatrix},$$

where W is a matrix with determinant ± 1, V is a square minor of a nonnegative Vandermonde matrix, and I is an identity matrix. Since W, V', and I are all nonsingular, so is M'. Thus, M satisfies condition (2). $\square$

Lemma 5.3. *There exists a configuration $X \in K^n$ with $\Omega(n^{\lceil r/2 \rceil})$ collapsible tuples, for some ordered field K.*

Proof: We explicitly construct a configuration $X \in \mathbb{R}(\Delta_1, \ldots, \Delta_{r-1}, \delta_1, \ldots, \delta_{\lfloor r/2 \rfloor}, \varepsilon_1, \ldots, \varepsilon_r)$ that satisfies the lemma. We assume without loss of generality that n is a multiple of r.

Write $\phi = \sum_{i=1}^{r} a_i t_i$ with real coefficients a_i and formal variables t_i. Let the matrix $M = (m_{ij})$ be given by the previous lemma. Our configuration X is the union of r smaller sets X_i, each containing n/r elements x_{ij} defined as follows.

$$x_{ij} = \frac{1}{a_i}\left((-1)^i(\Delta_{i-1} + \Delta_i) + \sum_{k=1}^{\lfloor r/2 \rfloor} m_{ik}\delta_k j + \varepsilon_i j^2\right)$$

For notational convenience, we define $\Delta_0 = \Delta_r = 0$.

We claim that any tuple $(x_{1p_1}, \ldots, x_{rp_r})$ satisfying the equation $M^\top(p_1, \ldots, p_r) = 0$ is collapsible. By condition (1), there are $\Omega((n/r)^{\lceil r/2 \rceil}) = \Omega(n^{\lceil r/2 \rceil})$ such tuples. The adversary collapses the tuple by replacing X with $\check{X}$, with elements

$$\check{x}_{ij} = \frac{1}{a_i}\left((-1)^i(\Delta_{i-1} + \Delta_i) + \sum_{k=1}^{\lfloor r/2 \rfloor} m_{ik}\delta_k j + \varepsilon_i (j - p_i)^2\right),$$

or more succinctly, $\check{x}_{ij} = x_{ij} + \varepsilon_i(p_i^2 - 2jp_i)/a_i$.

For example, in the simplest nontrivial case $r = 3$, our adversary configuration X lies in the field $\mathbb{R}(\Delta_1, \Delta_2, \delta_1, \varepsilon_1, \varepsilon_2, \varepsilon_3)$. If we take $M = (1, 1, -1)^\top$, then X contains the following elements, for all $1 \le j \le n/3$:

$$\begin{aligned}
x_{1j} &= (-\Delta_1 \phantom{{}+\Delta_2} + \delta_1 j + \varepsilon_1 j^2)/a_1,\\
x_{2j} &= (\Delta_1 + \Delta_2 + \delta_1 j + \varepsilon_2 j^2)/a_2,\\
x_{3j} &= (- \Delta_2 + \delta_1 j + \varepsilon_3 j^2)/a_3,
\end{aligned}$$

The indices of each allegedly collapsible tuple satisfy the equation $p_1 + p_2 = p_3$, and the corresponding collapsed configuration $\check{X}$ has the following elements:

$$\begin{aligned}
\check{x}_{1j} &= (-\Delta_1 \phantom{{}+\Delta_2} + \delta_1 j + \varepsilon_1 (j - p_1)^2)/a_1,\\
\check{x}_{2j} &= (\Delta_1 + \Delta_2 + \delta_1 j + \varepsilon_2 (j - p_2)^2)/a_2,\\
\check{x}_{3j} &= (- \Delta_2 + \delta_1 j + \varepsilon_3 (j - p_3)^2)/a_3.
\end{aligned}$$

Fix a tuple $(x_{1p_1}, \ldots, x_{rp_r})$ where $M^\top(p_1, \ldots, p_r) = 0$, and let $\check{X}$ be the corresponding collapsed configuration. We easily confirm that the collapsed tuple $(\check{x}_{1p_1}, \ldots, \check{x}_{rp_r})$ is degenerate. It remains to show that every other r-linear query expression has the same sign in both X and $\check{X}$.

To distinguish between the query polynomials and their value at a particular input, let t_{ij} be the formal variable corresponding to each element x_{ij} in the configuration X above.

Consider the query polynomial $Q = \sum_{i=1}^{r} Q_i$, where for each i,

$$Q_i = a_i \sum_{j=1}^{n/r} \alpha_{ij} t_{ij},$$

and at most r of the coefficients α_{ij} are not zero. We refer to t_{ij} as a *query variable* if its coefficient α_{ij} is not zero. We define A_i and J_i as

$$A_i = \sum_{j=1}^{n/r} \alpha_{ij} \qquad \text{and} \qquad J_i = \sum_{j=1}^{n/r} \alpha_{ij} j.$$

We can rewrite the query expression $Q(X)$ as a real linear combination of the infinitesimals as follows.

$$\begin{aligned}
Q(X) &= \sum_{i=1}^{r} \sum_{j=1}^{n/r} \alpha_{ij} \left((-1)^i (\Delta_{i-1} + \Delta_i) + \sum_{k=1}^{\lfloor r/2 \rfloor} m_{ik} \delta_k j + \varepsilon_i j^2 \right) \\
&= \sum_{i=1}^{r} \left(\left(\sum_{j=1}^{n/r} \alpha_{ij} \right) (-1)^i (\Delta_{i-1} + \Delta_i) + \left(\sum_{j=1}^{n/r} \alpha_{ij} j \right) \left(\sum_{k=1}^{\lfloor r/2 \rfloor} m_{ik} \delta_k \right) + \left(\sum_{j=1}^{n/r} \alpha_{ij} j^2 \right) \varepsilon_i \right) \\
&= \sum_{i=1}^{r-1} (-1)^i (A_i - A_{i+1}) \Delta_i + \sum_{k=1}^{\lfloor r/2 \rfloor} \left(\sum_{i=1}^{r} m_{ik} J_i \right) \delta_k + \sum_{i=1}^{r} \left(\sum_{j=1}^{n/r} \alpha_{ij} j^2 \right) \varepsilon_i
\end{aligned}$$

Let us define

$$D_i = (-1)^i (A_i - A_{i+1}), \qquad d_k = \sum_{i=1}^{r} m_{ik} J_i, \quad \text{and} \qquad e_i = \sum_{j=1}^{n/r} \alpha_{ij} j^2,$$

for each i and k, so that

$$Q(X) = \sum_{i=1}^{r-1} D_i \Delta_i + \sum_{k=1}^{\lfloor r/2 \rfloor} d_k \delta_k + \sum_{i=1}^{r} e_i \varepsilon_i.$$

The sign of $Q(X)$ is the sign of the first nonzero coefficient in this expansion; in particular, $Q(X) = 0$ if and only if *every* coefficient is zero. Similarly, we can write

$$Q(\check{X}) = \sum_{i=1}^{r-1} D_i \Delta_i + \sum_{k=1}^{\lfloor r/2 \rfloor} d_k \delta_k + \sum_{i=1}^{r} \check{e}_i \varepsilon_i,$$

where for each i,

$$\check{e}_i = \sum_{j=1}^{n/r} \alpha_{ij} (j - p_i)^2 = e_i - 2 p_i J_i + p_i^2 A_i.$$

If any of the coefficients D_i or d_k is nonzero, then the first such coefficient determines the sign of both $Q(X)$ and $Q(\check{X})$. Thus, it suffices to consider only queries for which every $D_i = 0$ and every $d_k = 0$. Note that in this case, all the A_i's are equal. There are three cases to consider.

Case 1. Suppose no subset X_i contains exactly one of the query variables. (This includes the case where all query variables belong to the same subset.) Then at most $\lfloor r/2 \rfloor$ of the Q_i's are not identically zero, and it follows that $A_i = 0$ for all i. The vector J consisting of the $\lfloor r/2 \rfloor$ (or fewer) nonzero J_i's must satisfy the matrix equation $(M')^\top J = 0$, where M' is a square minor of the matrix M. By condition (2) above, M' is nonsingular, so *all* the J_i's must be zero. It follows that $\check{e}_i = e_i$ for all i, which implies that $Q(X) = Q(\check{X})$.

Case 2. Suppose some subset X_i contains exactly one query variable t_{ij} and some other subset $X_{i'}$ contains none. Then $A_i = \alpha_{ij}$ and $A_{i'} = 0$. Since A_i and $A_{i'}$ are equal, we must have $\alpha_{ij} = 0$, but this contradicts the assumption that x_{kj} is a query variable. Thus, this case never happens.

Case 3. Finally, suppose each query variable comes from a different subset. (This includes the case of a direct query on what we claim is a collapsible tuple.) Recall that all the A_i's are equal. Since we are only interested in the sign of the query, we can assume without loss of generality that $A_i = \alpha_{ij} = 1$ for each query variable t_{ij}. Thus, each of the coefficients e_i is positive, which implies that $Q(X)$ is positive. Furthermore, unless the query variables are exactly x_{ip_i} for all i, each of the coefficients $\check{e}_i$ is also positive, which means $Q(\check{X})$ is also positive.

This completes the proof of Lemma 5.3. $\square$

5.2.2 Moving Back to the Reals

The configurations we construct are not directly usable in an adversary argument, because the algorithms we consider are only required to be correct when given real input. Thus, before we can apply our adversary argument, we must eliminate the infinitesimals. Since we know that a single real configuration cannot be hard for *every* algorithm, we are forced to derive, for each algorithm, a corresponding real configuration that is hard for that particular algorithm. Rather than constructing such configurations explicitly in terms of the coefficients of the query polynomials, as was done in [56, 55], we nonconstructively derive their existence from our infinitesimal construction.

Let $\mathcal{Q}_{\mathcal{A}}$ denote the set of query polynomials used by any r-linear decision tree $\mathcal{A}$. (We assume, without loss of generality, that $\mathcal{Q}_{\mathcal{A}}$ includes all $\Theta(n^r)$ direct queries, since otherwise the algorithm cannot correctly detect all possible degenerate tuples.) For any input configuration X, we call an r-tuple of elements in X *relatively collapsible* if the following properties are satisfied.

(1) The tuple is nondegenerate.

(2) There exists another *collapsed* configuration $\check{X}$, such that the corresponding tuple in $\check{X}$ is degenerate, but the sign of every other polynomial in $\mathcal{Q}_{\mathcal{A}}$ is the same for both configurations.

Clearly, any collapsible tuple is also relatively collapsible. To prove a lower bound, it suffices to prove, for each r-linear decision tree $\mathcal{A}$, the existence of a corresponding nondegenerate input configuration with a large number of relatively collapsible tuples.

Lemma 5.4. *For any r-linear decision tree $\mathcal{A}$, there exists a real configuration $X_{\mathcal{A}} \in \mathbb{R}^n$ with $\Omega(n^{\lceil r/2 \rceil})$ relatively collapsible tuples.*

Proof: Fix $\mathcal{A}$, and let $X \in K^n$ be the configuration given by Lemma 5.3. Each of the collapsible tuples in X is clearly also relatively collapsible. Each relatively collapsible tuple Y in X corresponds to a polynomial ϕ_Y, such that $\phi_Y(X) = \phi(Y)$. Call the set of these polynomials Φ.

It follows directly from the definitions that the following elementary formula holds in K.

$$\exists X \bigwedge_{\phi_Y \in \Phi} \left(\phi_Y(X) \neq 0 \ \wedge \ \exists \check{X} \left(\phi_Y(\check{X}) = 0 \ \wedge \bigwedge_{q \in \mathcal{Q}_{\mathcal{A}} \setminus \{\phi_Y\}} \operatorname{sgn} q(X) = \operatorname{sgn} q(\check{X}) \right) \right)$$

This is just a convenient shorthand for the actual formula. Each reference to $\phi_Y(X)$ or $q(X)$ should be expanded into an explicit polynomial in X. The equation $\operatorname{sgn} a = \operatorname{sgn} b$ into the boolean formula $((ab > 0) \vee (a = 0 \wedge b = 0))$. Since the sets Φ and $\mathcal{Q}_{\mathcal{A}}$ are finite, the expanded formula is also finite and therefore elementary.

Since K is a subset of its real closure $\widetilde{K}$, and the formula is only existentially quantified, the formula also holds in $\widetilde{K}$. Thus, by the Transfer Principle, it also holds in $\mathbb{R}$. The lemma follows immediately. $\square$

With a little more care, we can show that the real configurations are derived by replacing the infinitesimals by sufficiently small and sufficiently well-separated real values, but this is not necessary to prove our lower bounds.

5.2.3 Perturbing into General Position

One final problem remains. The adversary configurations we construct (and by implication, the real configurations we get by invoking the previous lemma) are degenerate. In simple cases, we can construct nondegenerate adversary configurations, but this becomes considerably more difficult as we consider larger values of r. Instead, we show nonconstructively that the existing degenerate configurations can be perturbed into general position.

Lemma 5.5. *For any r-linear decision tree $\mathcal{A}$, there exists a nondegenerate real configuration $X^*_{\mathcal{A}} \in \mathbb{R}^n$ with $\Omega(n^{\lceil r/2 \rceil})$ relatively collapsible tuples.*

Proof: As before, let $\mathcal{Q}_{\mathcal{A}}$ denote the set of query polynomials used by $\mathcal{A}$. Each query in $\mathcal{Q}_{\mathcal{A}}$ induces a hyperplane in the configuration space $\mathbb{R}^n$, and these hyperplanes define a cell complex, called the *arrangement* [62]. Color each hyperplane "red" if it corresponds to a direct query, and "green" otherwise.

Each input configuration corresponds to a point in some cell $\mathcal{C}$ in this arrangement. Nondegenerate configurations correspond to points in n-dimensional cells; degenerate configurations correspond to points in cells of lower dimension. As long as we never change the result of any query in $\mathcal{Q}_{\mathcal{A}}$, changing the entries in a configuration corresponds to moving the configuration point within $\mathcal{C}$. Collapsing a collapsible tuple moves the configuration point onto a boundary facet of $\mathcal{C}$ uniquely spanned by a red hyperplane. (That is, the red hyperplane is the only hyperplane that contains the facet but not the entire cell.) To prove the lemma, it suffices to find a full-dimensional cell with $\Omega(n^{\lceil r/2 \rceil})$ red boundary facets.

Let $\mathcal{C}$ be an arbitrary cell, and let $\mathcal{C}'$ be one of the cells in its boundary. Any hyperplane that uniquely spans a facet of $\mathcal{C}'$ also uniquely spans a facet of $\mathcal{C}$. Thus, if there is a cell of *any* dimension with $\Omega(n^{\lceil r/2 \rceil})$ red boundary facets, then there must be an n-dimensional cell with $\Omega(n^{\lceil r/2 \rceil})$ red boundary facets. Since relatively collapsible tuples correspond to red boundary facets, it suffices to show that there exists a (possibly degenerate) real configuration with $\Omega(n^{\lceil r/2 \rceil})$ relatively collapsible tuples. Such a configuration is guaranteed by Lemma 5.4. □

This lemma, together with our usual adversary argument, completes the proof of Theorem 5.1.

5.3 Matching Nonuniform Upper Bounds

Our lower bound matches known upper bounds when r is odd, but is a logarithmic factor away when r is even and greater than 2. We use the following result of Fredman [80] to show that our lower bounds cannot be improved in this case.

Lemma 5.6 (Fredman). *Let Γ be a subset of the $n!$ orderings of $\{1, \ldots, n\}$ for some fixed n. There exists a comparison tree of depth at most $\log_2(|\Gamma|) + 2n$ that sorts any sequence of n numbers with order type in Γ.*

Theorem 5.7. *For any n and $r > 2$, there exists an r-linear decision tree with depth $O(n^{\lceil r/2 \rceil})$ that solves any r-linear satisfiability problem with n inputs.*

Proof: It suffices to consider the case when r is even, since for any odd r there is a simple uniform algorithm with running time $O(n^{(r+1)/2})$. Suppose we are trying to satisfy the equation $\sum_{i=1}^{r} a_i t_i = 0$ for some fixed coefficients $a_i \in \mathbb{R}$. Given a configuration $X = (x_1, \ldots, x_n) \in \mathbb{R}^n$, we (implicitly) construct sets J and K of $n^{\underline{r/2}}$ real numbers each[2], as follows:

$$J = \left\{ \sum_{i=1}^{r/2} a_i x_{j_i} \;\middle|\; \{j_1, j_2, \ldots, j_{r/2}\} \subset \{1, 2, \ldots, n\} \right\}$$

$$K = \left\{ \sum_{i=1}^{r/2} -a_{i+r/2} x_{k_i} \;\middle|\; \{k_1, k_2, \ldots, k_{r/2}\} \subset \{1, 2, \ldots, n\} \right\}$$

Then X is degenerate if and only if the sets J and K share an element defined by tuples whose index sets $\{j_i\}$ and $\{k_i\}$ are disjoint. We can detect this condition by sorting $J \cup K$ using Fredman's "comparison" tree, which is really a r-linear decision tree.

Every pair of elements of $J \cup K$ induces a hyperplane in the configuration space $\mathbb{R}^n$. There is a one-to-one correspondence between the cells in the resulting hyperplane arrangement and the possible orderings of $J \cup K$. Since an arrangement of N hyperplanes

[2] For any integer $a \geq 0$, the falling factorial power $n^{\underline{a}}$ is defined as $n(n-1)\cdots(n-a+1) = n!/(n-a)!$ [92].

in $\mathbb{R}^D$ has at most $\sum_{i=0}^{D}\binom{N}{i} = O(N^D)$ cells [62], there are at most $O((2n^{\underline{r/2}})^{2n}) = O((2n)^{rn})$ possible orderings. It follows that the depth of Fredman's decision tree is at most $4n^{\underline{r/2}} + O(rn\log n) = O(n^{r/2})$. □

Of course, this result does not imply the existence of a single $O(n^{\lceil r/2\rceil})$-time algorithm that works for *all* values of n. Closing the logarithmic gap between these upper and lower bounds, even for the special case of sorting the Minkowski sum $X+Y$ of two sets, is a long-standing and very difficult open problem. The closest result is an algorithm of Steiger and Streinu [139] that sorts $X+Y$ in $O(n^2\log n)$ time using only $O(n^2)$ comparisons.

5.4 Conclusions and Open Problems

We have developed a new general technique for proving lower bounds in decision tree models of computation. We show that it suffices to construct a single input configuration, possibly degenerate and possibly containing infinitesimals, containing several collapsible tuples. Using this technique, we have proven $\Omega(n^{\lceil r/2\rceil})$ lower bounds on the depth of any r-linear decision tree that decides an r-variable linear satisfiability problem. This is the best possible lower bound in this model.

An immediate open problem is to improve our lower bounds to stronger models of computation. It seems "obvious" that linear queries with more variables or higher-degree queries almost never give us useful information, and therefore can almost always be added to our model of computation with impunity. Can we define a general class of "allowable" queries for linear satisfiability problems, as we did in the previous chapters?

Can the techniques developed in this chapter be applied to higher-order polynomial satisfiability problems?

There are simple reductions from linear satisfiability problems to many other higher-dimensional geometric problems. For example, finding a d-tuple in the zeroset of the polynomial $\sum_{i=1}^{d} t_i$ can be reduced to the d-dimensional affine degeneracy problem. (See Lemma 2.3.) Several more good examples can be found in Gajentaan and Overmars' collection of 3SUM-hard problems [86]. Unfortunately, these reductions use primitives disallowed by the models of computation in which our lower bounds hold. Consequently, our linear satisfiability lower bounds do *not* imply lower bounds for these geometric problems.

Can the techniques of this chapter be applied directly to higher-dimensional problems? (My original presentation of these results [74] claimed to do just that, but the proofs were flawed; Lemma 4.1 in [74] is actually false.)

Ultimately, we would like to prove a lower bound larger than $\Omega(n \log n)$ for *any* non-NP-hard polynomial satisfiability problem, in some general model of computation such as linear decision trees, algebraic decision trees, or even algebraic computation trees. Linear satisfiability problems seem to be good candidates for study.

"Now, even though their jumping is blind and wholly random, there are billions upon billions of atoms in every interstice, and as a consequence of this great number, their little skips and scamperings give rise to, among other things—and purely by accident—to significant configurations.... Do you know what a configuration is, blockhead?"

"No insults, please!" said Pugg. "For I am not your usual uncouth pirate, but refined and with a Ph.D., and therefore extremely high-strung."

— Stanislaw Lem (translated by Michael Kandel), "The Sixth Sally, or How Trurl and Klapaucius Created a Demon of the Second Kind to Defeat the Pirate Pugg", *The Cyberiad*, 1974

Part II

Divide-and-Conquer Lower Bounds

He who can properly define and divide is to be considered a god.

— Plato, c. 400 BC

Great wits are sure to madness near allied,
And thin partitions do their bounds divide.

— John Dryden, *Absalom and Achitophel*, 1681

Chapter 6

Hopcroft's Problem

In the early 1980's, John Hopcroft posed the following problem to several members of the computer science community.

Given a set of n points and n lines in the plane, does any point lie on a line?

Hopcroft's problem arises as a special case of many other geometric problems, including collision detection, ray shooting, and range searching.

The earliest sub-quadratic algorithm for Hopcroft's problem, due to Chazelle [32], runs in time $O(n^{1.695})$. (Actually, this algorithm counts intersections among a set of n line segments in the plane, but it can easily be modified to count point-line incidences instead.) A very simple algorithm, attributed to Hopcroft and Seidel [51], described in [62, p. 350], runs in time $O(n^{3/2}\log^{1/2} n)$. Cole, Sharir, and Yap [51] combined these two algorithms, achieving a running time of $O(n^{1.412})$. Edelsbrunner, Guibas, and Sharir [66] developed a randomized algorithm with expected running time $O(n^{4/3+\varepsilon})$ [1] ; see also [64]. A somewhat simpler algorithm with the same running time was developed by Chazelle, Sharir, and Welzl [43]. Further research replaced the n^{ε} term in this upper bound with a succession of smaller and smaller polylogarithmic factors. The running time was improved by Edelsbrunner, Guibas, Hershberger, Seidel, Sharir, Snoeyink, and Welzl [63] to $O(n^{4/3}\log^{4} n)$ (expected); then by Agarwal [1] to $O(n^{4/3}\log^{1.78} n)$; then by Chazelle [35] to $O(n^{4/3}\log^{1/3} n)$; and most recently by Matoušek [111] to $n^{4/3}2^{O(\log^* n)}$. This is currently the fastest algorithm known. Matoušek's algorithm can be tuned to detect incidences among

[1] In time bounds of this form, ε refers to an arbitrary positive constant. For any fixed value of ε, the algorithm can be tuned to run within the stated time bound. However, the multiplicative constants hidden in the big-Oh notation depend on ε, and tend to infinity as ε approaches zero.

n points and m lines in the plane in time $O(n \log m + n^{2/3} m^{2/3} 2^{O(\log^*(n+m))} + m \log n)$ [54], or more generally among n points and m hyperplanes in $\mathbb{R}^d$ in time

$$O\left(n \log m + n^{d/(d+1)} m^{d/(d+1)} 2^{O(\log^*(n+m))} + m \log n\right).$$

The lower bound history is much shorter. The only previously known lower bound is $\Omega(n \log m + m \log n)$, in the algebraic decision tree and algebraic computation tree models, by reduction from the problem of detecting an intersection between two sets of real numbers [138, 16].

In this chapter, we establish new lower bounds on the complexity of Hopcroft's problem. We formally define a general class of *partitioning algorithms*, which includes most (if not all) of the algorithms mentioned above, and prove that any partitioning algorithm can be forced to take time $\Omega(n \log m + n^{2/3} m^{2/3} + m \log n)$ in two dimensions, or $\Omega(n \log m + n^{5/6} m^{1/2} + n^{1/2} m^{5/6} + m \log n)$ in three or more dimensions. We improve this lower bound slightly in dimensions four and higher for the counting version of Hopcroft's problem, where we want to know the number of incident point-hyperplane pairs.

Informally, a partitioning algorithm covers space with a constant number of (not necessarily disjoint) connected regions, determines which points and hyperplanes intersect each region, and recursively solves each of the resulting subproblems. The algorithm may apply projective duality to reverse the roles of the points and hyperplanes, that is, to partition the input according to which dual hyperplanes and dual points intersect each region. The algorithm is also allowed to merge subproblems arbitrarily before partitioning. For purposes of proving lower bounds, we assume that partitioning the points and hyperplanes requires only linear time, regardless of the complexity of the regions or how they depend on the input. We give a more formal definition of partitioning algorithms in Section 6.3.

To develop lower bounds in this model, we first define a combinatorial representation of the relative order type of a set of points and hyperplanes, called a *monochromatic cover*, and derive lower bounds on its worst case complexity. A monochromatic cover is a partition of the sign matrix induced by the relative orientations of the points and hyperplanes into (not necessarily disjoint) minors, such that all entries in each minor are equal. The size of a cover is the total number of rows and columns in the minors. The main result of this chapter (Theorem 6.19) is that the running time of any partitioning algorithm is bounded below by the size of some monochromatic cover of its input.

Some related results deserve to be mentioned here. Erdős constructed a set of n points and n lines in the plane with $\Omega(n^{4/3})$ incident point-line pairs [62, p. 112]. It follows immediately that any algorithm that reports all incident pairs requires time $\Omega(n^{4/3})$ in the worst case. Of course, we cannot apply this argument to either the decision version or the counting version of Hopcroft's problem, since the output size for these problems is constant. Our planar lower bounds are ultimately based on the Erdős configuration.

Chazelle [33, 38] has established lower bounds for the closely related *simplex range searching* problem: Given a set of points and a set of simplices, how many points are in each simplex? For example, any data structure of size s that supports triangular range queries among n points in the plane requires $\Omega(n/\sqrt{s})$ time per query [33]. It follows that answering n queries over n points requires $\Omega(n^{4/3})$ time in the worst case. For the offline version of the same problem, where all the triangles are known in advance, Chazelle establishes a slightly weaker bound of $\Omega(n^{4/3}/\log^{4/3} n)$ [38], although an $\Omega(n^{4/3})$ lower bound follows immediately from the Erdős construction using Chazelle's methods. In higher dimensions, Chazelle's results imply lower bounds of $\Omega(n^{2d/(d+1)}/\log^{d/(d+1)} n)$ and $\Omega(n^{2d/(d+1)}/\log^{5/2-\gamma} n)$ in the online and offline cases, respectively, where $\gamma > 0$ is a small constant that depends on d. All these lower bounds hold in the Fredman/Yao semigroup arithmetic model; see Section 1.3. For related results, see also [23, 44].

Lower bounds in the semigroup model are based on the existence of configurations of points and ranges, such as the planar Erdős configuration, whose incidence graphs have (a subgraph with) no large complete bipartite subgraphs. Our lower bounds have a similar basis. In Section 6.2, we develop point-hyperplane configurations of this type, naturally generalizing the Erdős configuration to arbitrary dimensions. These configurations also allow us to extend Chazelle's offline lower bounds to a counting version of Hopcroft's problem.

6.1 Easy Quadratic Lower Bounds

In light of the results of Part I, it is natural to ask whether we can prove lower bounds for Hopcroft's problem in a model of computation that allows only simple queries. The appropriate primitive to consider is the *relative orientation query*: Given a point and a hyperplane, does the point lie above, on, or below the hyperplane? Algebraically, the result of a relative orientation query is given by the inner product of the homogeneous

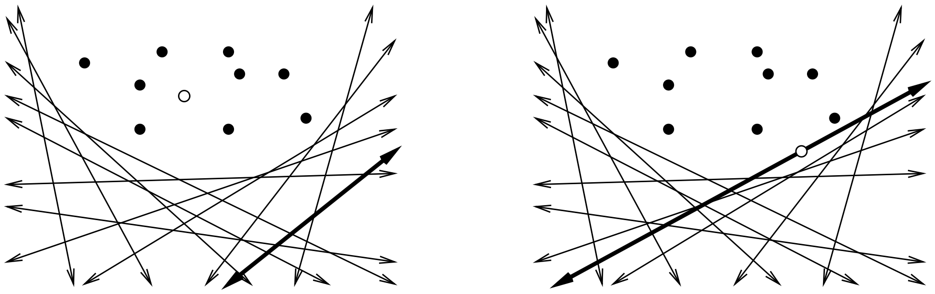


Figure 6.1. An easy adversary construction for Hopcroft's problem. (a) The original adversary configuration. (b) A collapsed configuration.

coordinate vectors of the point and the hyperplane [140]. Surprisingly, we can easily establish a *quadratic* lower bound for Hopcroft's problem if this is the only primitive we are allowed.

Theorem 6.1. *Any decision tree algorithm that decides Hopcroft's problem in* $\mathbb{R}^d$ *for any* $d \geq 1$*, using only relative orientation queries, must have depth at least* nm*.*

Proof: The lower bound follows from a simple adversary argument. The adversary presents the algorithm with a set of n points and m hyperplanes in which every point is above every hyperplane. If the algorithm does not perform a relative orientation query for some point-hyperplane pair, the adversary can move that point onto that hyperplane without changing the relative orientation of any other pair. See Figure 6.1. The algorithm cannot tell the two configurations apart, even though one has an incidence and the other does not. Thus, in order to be correct, the algorithm must perform a relative orientation query for every point-hyperplane pair. □

Obviously, this lower bound is tight.

In dimensions higher than one, we can considerably strengthen the model of computation in which this quadratic lower bound holds. We will explicitly describe only the two-dimensional case; generalization to higher dimensions is straightforward. Our new model of computation is a decision tree with three types of primitives: relative orientation queries, point queries, and line queries. A point query is any decision that is based exclusively on the coordinates the input points. Line queries are defined analogously. We emphasize that point queries can combine information from any number of points, and line

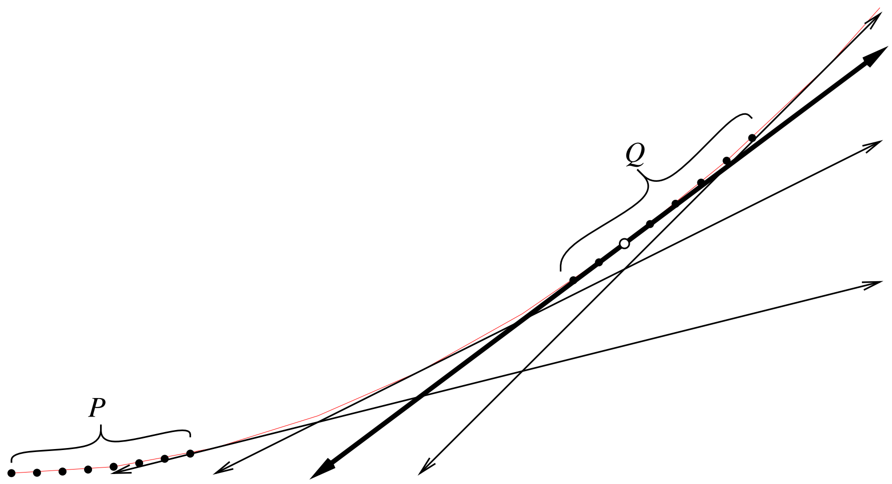


Figure 6.2. A harder adversary construction for Hopcroft's problem. The white point in Q is on the dark line; otherwise, every point is above every line.

queries from any number of lines. We call a point query *algebraic* if the result is given by the sign of a multivariate polynomial, not necessarily of bounded degree, evaluated at the point coordinates.

Theorem 6.2. *Any decision tree algorithm that solves Hopcroft's problem in the plane, using only relative orientation queries, algebraic point queries, and (arbitrary) line queries, must make at least* nm *relative orientation queries in the worst case.*

Proof: For any real number x_0, let $P(x_0)$ denote the set of n points

$$\left\{ \left(x_0, x_0^2\right), \left(x_0+1, (x_0+1)^2\right), \ldots, \left(x_0+n-1, (x_0+n-1)^2\right) \right\},$$

and let $L(x_0)$ denote the set of m lines tangent to the unit parabola $y = x^2$ at the points

$$\left\{ \left(x_0+n, (x_0+n)^2\right), \left(x_0+2n, (x_0+2n)^2\right), \ldots, \left(x_0+mn, (x_0+mn)^2\right) \right\}.$$

As before, our lower bound follows from an adversary argument. The adversary initially presents the points $P = P(x_0)$ and the lines $L = L(x_0)$, for some real value x_0 to be specified later. If the algorithm does not perform a relative orientation query for the ith point and the jth line, then the adversary replaces the points with the new set $Q = P(x_0 + in - j + 1)$. We easily verify that in the new configuration, the ith point and the jth line are incident, but otherwise, every point is above every line. See Figure 6.2.

Since the adversary does not change the lines, no line query can distinguish between the two configurations. It remains only to consider the point queries. The result of any algebraic point query in $P(x)$ is given by the sign of a polynomial in the single variable x. Let $r_{\max}$ be the largest root of all the point query polynomials used by the algorithm, ignoring those that are identically zero. If x_1 and x_2 are both larger than $r_{\max}$, then every point query polynomial has the same sign at both $P(x_1)$ and $P(x_2)$. Thus, if the adversary fixes $x_0 > r_{\max}$, then the algorithm cannot distinguish between the original point set P and any of the collapsed point sets Q using point queries.

It follows that the algorithm cannot be correct unless it performs a relative orientation query for every pair. $\square$

Our restriction to algebraic point queries is actually stronger than the result requires; it suffices that for all x in some sufficiently large interval, the result of every point query on the set $P(x)$ is constant. If we rephrase this argument in the dual space, we get a quadratic lower bound in a model that allows arbitrary point queries but requires line queries to be algebraic.

These adversary arguments actually give us a quadratic lower bound for the much easier *halfspace emptiness* problem "Is every point above every hyperplane?". We consider this problem in greater detail in the next chapter. Our arguments can easily be modified to apply to almost any range searching problem, and a wide range of other related problems; see [78]. We leave the details and further generalizations as exercises for the reader.

Of course, none of the sub-quadratic algorithms listed previously follow the models of computation considered in this section. Unlike the degeneracy problems considered in Part I, there does not appear to be a small fixed set of primitives that are used by all known algorithms for Hopcroft's problem. Many algorithms define several levels of higher-order geometric objects, and some of their decisions are based on large fractions of the input.

In light of our results, it is clear that higher-order primitives that involve both points and lines, such as "Is this point to the left or right of the intersection of these two lines?", are necessary to achieve nontrivial upper bounds. If we allow any primitives of this type, however, it seems unlikely that our earlier adversary techniques can be used to derive nontrivial lower bounds, either for Hopcroft's problem or for other range searching problems. We leave the development of such lower bounds as an interesting open problem.

6.2 Incidences and Monochromatic Covers

Let $P = \{p_1, p_2, \ldots, p_n\}$ be a set of points and $H = \{h_1, h_2, \ldots, h_m\}$ be a set of hyperplanes in $\mathbb{R}^d$. These two sets naturally induce a *relative orientation matrix* $M(P, H) \in \{+, 0, -\}^{n \times m}$. Each row of $M(P, H)$ corresponds to one of the points in P; each column corresponds to a hyperplane in H; and the (i, j)th entry denotes whether the point p_i is above $(+)$, on (0), or below $(-)$ the hyperplane h_j. Any minor of the matrix $M(P, H)$ is itself a relative orientation matrix $M(P', H')$, for some $P' \subseteq P$ and $H' \subseteq H$.

We call a sign matrix *monochromatic* if all its entries are equal. A *minor cover* of a matrix is a set of minors whose union is the entire matrix. If every minor in the cover is monochromatic, we call it a *monochromatic cover*. The *size* of a minor is the number of rows plus the number of columns, and the size of a minor cover is the sum of the sizes of the minors in the cover. Given a set of points and hyperplanes, a monochromatic cover of its relative orientation matrix provides a succinct combinatorial representation of its relative order type.

We similarly define a succinct representation for the incidence structure of a set of points and hyperplanes. A *zero cover* of P and H is a collection of monochromatic minors that covers all (and only) the zeros in the relative orientation matrix $M(P, H)$. A zero cover can also be interpreted as a partition of the incidence graph induced by P and H into (not necessarily disjoint) complete bipartite subgraphs.

Monochromatic covers for 0-1 matrices have been previously used to prove lower bounds for communication complexity problems [106]. Typically, however, these results make use of the number of minors in the cover, not the size of the cover as we define it here.[2] Covers of bipartite graphs by complete subgraphs were introduced by Tarján [145] in the context of switching theory. Tuza [149], and independently Chung, Erdős, and Spencer [45], showed that every $n \times m$ bipartite graph has such a cover of size $O(nm/\log(\max(m, n)))$ and that this bound is tight in the worst case, up to constant factors. These results apply immediately to monochromatic covers of arbitrary sign matrices. See also [2] for a geometric application of bipartite clique covers.

Relative orientation matrices are defined in terms of a fixed (projective) coordinate system, which determines what it means for a point to be "above" or "below" a hyperplane.

[2] Since any row or column can be split into three or fewer monochromatic minors, any sign matrix can be covered by $3\min(m, n)$ such minors. Furthermore, there are sets of n points and m lines in the plane whose relative orientation matrices require $3\min(m, n)$ monochromatic minors to cover them.

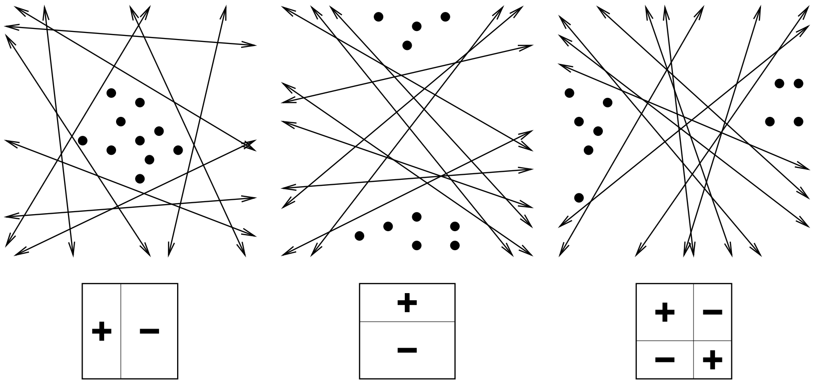


Figure 6.3. Collections of points and lines with simple relative orientation matrices.

This coordinate system determines which minors of the relative orientation matrix are monochromatic, and therefore determine the minimum monochromatic cover size. However, we easily observe that the minimum monochromatic cover size is independent of any choice of coordinate system, up to a factor of two, as follows. Call a relative orientation matrix *simple* if it can be changed into a monochromatic matrix by inverting some subset of the rows and columns. Projective transformations preserve simple minors. See Figure 6.3. Every monochromatic minor is simple, and every simple minor can be partitioned into four monochromatic minors, whose total size is twice that of the original minor.

We will use the following notation throughout the rest of the chapter. Given a set P of points and H of hyperplanes, let $I(P, H)$ denote the number of point-hyperplane incidences, $\zeta(P, H)$ the minimum size of their smallest zero cover, and $\mu(P, H)$ the size of their smallest monochromatic cover. Let $I_d(n, m)$ denote the maximum of $I(P, H)$ over all sets P of n points and all sets H of m hyperplanes in $\mathbb{R}^d$, and define $\zeta_d(n, m)$ and $\mu_d(n, m)$ similarly. Finally, let $\mu_d^\circ(n, m)$ denote the maximum of $\mu(P, H)$ over all sets of n points and m hyperplanes in $\mathbb{R}^d$ with no incidences. In the remainder of this section, we develop asymptotic lower bounds for $\mu_d^\circ(n, m)$ and $\zeta_d(n, m)$, which in turn imply lower bounds for $\mu_d(n, m)$.

6.2.1 One Dimension

In one dimension, points and hyperplanes are both just real numbers. We can always permute the rows and columns of the relative orientation matrix of two sets of numbers, by sorting the sets, so that the number of $+$ (resp. $-$) entries in successive rows or columns is nonincreasing (resp. nondecreasing). The matrix can then be split into two "staircase" matrices, one positive and one negative, and a collection of zero minors of total size at most $n+m$. We immediately observe that $\zeta_1(n,m) = n+m$.

Theorem 6.3. $\mu_1^\circ(m,n) = \begin{cases} \Theta(n \log_{n/m} m) & \textit{if } n > m \\ \Theta(n \log n) & \textit{if } n = m \\ \Theta(m \log_{m/n} n) & \textit{if } n < m \end{cases}$

Proof: Without loss of generality, we assume n and m are both powers of two. It suffices to bound the cover size of a monochromatic staircase with n rows and m columns.

First consider the simplest case, $n = m$. To prove the upper bound, we construct a cover of an arbitrary $n \times n$ staircase matrix by partitioning the staircase into an $n/2 \times k$ monochromatic minor and two smaller staircases, where k is the number of entries in the $n/2$th row of the original matrix. The total size $C(n,n)$ of this cover is bounded by the recurrence

$$C(n,n) \leq \max_{0 \leq k \leq n} (n/2 + k + C(n/2,k) + C(n/2, n-k))$$

whose solution is $C(n,n) \leq (3n/2)\lg n$.

To prove the matching lower bound, it suffices to consider a triangular matrix, where for all i, the ith row has i entries. We claim that any cover for this matrix must have size at least $(n/2)\lg n$. Fix a cover. Partition the staircase into an $n/2 \times n/2$ rectangle and two $n/2 \times n/2$ staircases. If a minor in the cover intersects the lower triangle, call it a lower minor; otherwise, call it an upper minor. The upper (resp. lower) minors induce a cover of the upper (resp. lower) triangle, of size at least $(n/4)\lg(n/2)$, by induction. It remains to bound the contribution of the rectangle to the total cover size.

If some row in the rectangle is completely contained in lower minors, then those lower minors have (altogether) $n/2$ more columns than we accounted for in the induction step. Otherwise, every row contains an element of an upper minor, and those upper minors have (altogether) $n/2$ more rows than we accounted for in the induction step. Thus, the rectangle contributes at least $n/2$ to the total cover size. This completes the proof for the case $n = m$.

Now suppose $n > m$. An explicit recursive construction gives us a cover of size $O(n \log_{n/m} m)$ for any $n \times m$ staircase. An inductive counting argument implies that any cover of the $n \times m$ triangular matrix, whose ith row has $\lceil im/n \rceil$ entries, must have size at least $(n/2)\log_{n/m} m$. In both arguments, we start by dividing the n rows of the matrix into n/m slabs of m rows each, and cutting each slab into a maximal rectangle and a smaller staircase. We leave the details as an easy exercise.

The final case $n < m$ is handled symmetrically. □

This bound simplifies to $\Theta(n+m)$ when either $n = O(m^k)$ or $m = O(n^k)$ for any constant $k < 1$, and to $\Theta(n \log n)$ when $n = \Theta(m)$. In the special case $n = m$, our upper bound proof is nothing more than an application of quicksort. The connection between the size of the monochromatic cover and the running time of the divide-and-conquer algorithm is readily apparent in this case. In Section 6.3, we generalize this connection to higher dimensions.

6.2.2 Two Dimensions

To derive lower bounds for $\mu_2^\circ(n, m)$ and $\zeta_2(n, m)$, we use the following combinatorial result of Erdős. (See [82] or [62, p.112] for proofs.)

Lemma 6.4 (Erdős). *For all n and m, there is a set of n points and m lines in the plane with $\Omega(n + n^{2/3}m^{2/3} + m)$ incident pairs. Thus, $I_2(n, m) = \Omega(n + n^{2/3}m^{2/3} + m)$.*

Fredman [82] uses Erdős' construction to prove lower bounds for dynamic range query data structures in the plane.[3] This lower bound is asymptotically tight. The corresponding upper bound was first proven by Szemerédi and Trotter [143]. A simpler proof, with better constants, was later given by Clarkson, Edelsbrunner, Guibas, Sharir, and Welzl [46]. A completely elementary proof has recently discovered by Székely [119].

Theorem 6.5. $\zeta_2(n, m) = \Omega(n + n^{2/3}m^{2/3} + m)$

Proof: It is not possible for two distinct points to both be adjacent to two distinct lines; any mutually incident set of points and lines has either exactly one point or exactly one line. It follows that for any set P of points and H of lines in the plane, $\zeta(P, H) \geq I(P, H)$. The theorem now follows from Lemma 6.4. □

Theorem 6.6. $\mu_2^\circ(n, m) = \Omega(n + n^{2/3}m^{2/3} + m)$

[3] Perhaps it is more interesting that Chazelle's static lower bounds [33, 38] do *not* use this construction.

Proof: Consider any configuration of n points and $m/2$ lines with $\Omega(n + n^{2/3}m^{2/3} + m)$ point-line incidences, as given by Lemma 6.4. Replace each line ℓ in this configuration with a pair of lines, parallel to ℓ and at distance ε on either side, where ε is chosen sufficiently small that all point-line distances in the new configuration are at least ε. The resulting configuration of n points and m lines clearly has no point-line incidences. We call a point-line pair in this configuration *close* if the distance between the point and the line is ε. There are $\Omega(n + n^{2/3}m^{2/3} + m)$ such pairs.

Now consider a single monochromatic minor in the relative orientation matrix of these points and lines. Let P' denote the set of points and H' the set of lines represented in this minor. We claim that the number of close pairs between P' and H' is small.

Without loss of generality, we can assume that all the points are above all the lines. If a point is close to a line, the point must be on the convex hull of P', and the line must support the upper envelope of H'. Thus, we can assume that both P' and H' are in convex position. In particular, we can order both the points and lines from left to right.

Either the leftmost point is close to at most one line, or the leftmost line is close to at most one point. It follows inductively that the number of close pairs is at most $|P'|+|H'|$, which is exactly the size of the minor. The theorem follows immediately. □

6.2.3 Three Dimensions

The technique we used in the plane does not generalize immediately to higher dimensions. Even in three dimensions, there are collections of points and planes where every points is incident to every plane. See Figure 6.4. In order to derive a lower bound for either $\zeta_3(m, n)$ or $\mu_3^\circ(n, m)$, we need a configuration of points and planes with many incidences, but without large sets of mutually incident points and planes. In the following lemma, we construct such a configuration, naturally generalizing Erdős' planar construction.

We use the notation $[n]$ to denote the set of integers $\{1, 2, \ldots, n\}$; $i \perp j$ to mean that i and j are relatively prime; and $\varphi(n)$ to denote the Euler totient function $\varphi(n)$, the number of positive integers less than or equal to n that are relatively prime to n.

Our construction relies on the following lemma, whose (simple) proof we omit. We refer the reader to [98] or [92] for relevant background.

Lemma 6.7. $\sum_{i=1}^{n} i^k \varphi(i) = \Theta(n^{k+2})$ *for any nonnegtive integer* k.

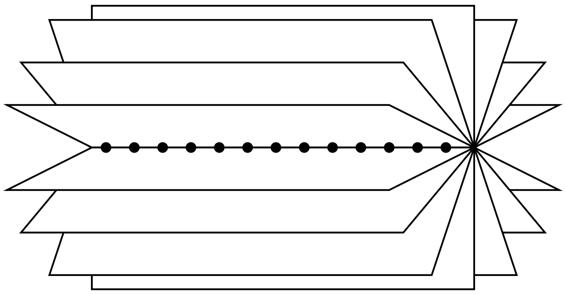


Figure 6.4. $I_3(n, m) = mn$. Every point lies on every plane.

Lemma 6.8. *For all* n *and* m *such that* $\lfloor n^{1/3} \rfloor < m$*, there exists a set* P *of* n *points and a set* H *of* m *planes, such that* $I(P, H) = \Omega(n^{5/6} m^{1/2})$ *and any three planes in* H *intersect in at most one point.*

Proof: Fix sufficiently large n and m such that $\lfloor n^{1/3} \rfloor < m$. Let $h(a, b, c; i, j)$ denote the plane passing through the points (a, b, c), $(a+i, b+j, c)$, and $(a+i, b, c+i-j)$. Let $p = \lfloor n^{1/3} \rfloor$ and $q = \lfloor \alpha(m/p)^{1/4} \rfloor$ for some suitable constant $\alpha > 0$. (Note that with n sufficiently large and m in the indicated range, p and q are both positive integers.)

Now consider the points $P = [p]^3 = \{(x, y, z) \mid x, y, z \in [p]\}$ and the hyperplanes

$$H = \left\{ h(a, b, c; i, j) \;\middle|\; i \in [q], j \in [i], i \perp j, a \in [i], b \in [j], c \in \Big[\lfloor p/2 \rfloor\Big] \right\}$$

The number of planes in H is

$$\left\lfloor \frac{p}{2} \right\rfloor \sum_{i=1}^{q} i \sum_{\substack{j=1 \\ j \perp i}}^{i} j = \left\lfloor \frac{p}{2} \right\rfloor \sum_{i=1}^{q} \frac{i^2 \varphi(i)}{2} = O(pq^4) = O(m).$$

By choosing the constant α appropriately and possibly adding in $o(m)$ extra planes, we can ensure that H contains exactly m planes. We claim that this collection of points and planes satisfies the lemma.

Since any rational plane can be represented in the form $h(a, b, c; i, j)$ in an infinite number of different ways, we must first check that the planes in H are actually distinct! Consider a single plane $h = h(a, b, c; i, j) \in H$. Since i, j, and $i - j$ are pairwise relatively prime, h intersects exactly one point (x, y, z) such that $x \in [i]$ and $y \in [j]$, namely, the point (a, b, c). Thus, for each fixed i and j we use, the planes $h(a, b, c; i, j) \in H$ are distinct. Since planes with different "slopes" are clearly different, H contains m distinct planes.

For all $k \in [\lfloor p/2i \rfloor]$, the intersection of $h(a,b,c;i,j) \in H$ with the plane $x = a + ki$ contains at least k points of P. It follows that

$$\left| P \cap h(a,b,c;i,j) \right| \geq \sum_{k=1}^{\lfloor p/2i \rfloor} k > \frac{1}{2} \left\lfloor \frac{p}{2i} \right\rfloor^2 .$$

Thus, the total number of incidences between P and H can be calculated as follows.

$$\begin{aligned} I(P,H) &\geq \left\lfloor \frac{p}{2} \right\rfloor \sum_{i=1}^{q} i \sum_{\substack{j=1 \\ j \perp i}}^{i} \frac{j}{2} \left\lfloor \frac{p}{2i} \right\rfloor^2 \\ &\geq \left\lfloor \frac{p}{2} \right\rfloor^3 \sum_{i=1}^{q} \sum_{\substack{j=1 \\ j \perp i}}^{i} \frac{j}{2i} \\ &= \left\lfloor \frac{p}{2} \right\rfloor^3 \sum_{i=1}^{q} \frac{\varphi(i)}{4} \\ &= \Omega(p^3 q^2) \\ &= \Omega(n^{5/6} m^{1/2}) \end{aligned}$$

Finally, If H contains three planes that intersect in a line, the intersection of those planes with the plane $x = 0$ must consist of three concurrent lines. It suffices to consider only the planes passing through the point $(1,1,1)$, since for any other triple of planes in H there is a parallel triple passing through that point. The intersection of $h(1,1,1;i,j)$ with the plane $x = 0$ is the line through $(0, 1 - j/i, 1)$ and $(0, 1, j/i)$. Since $i \perp j$, each such plane determines a unique line. Furthermore, since all these lines are tangent to a parabola, no three of them are concurrent. It follows that the intersection of any three planes in H consists of at most one point. $\square$

Edelsbrunner, Guibas, and Sharir [64] prove an upper bound of $O(n \log m + n^{4/5+2\varepsilon} m^{3/5-\varepsilon} + m)$ on the maximum number of incidences between n points and m planes, where no three planes contain a common line. The probabilistic counting techniques of Clarkson *et al.* [46] imply the better upper bound $O(n + n^{4/5} m^{3/5} + m)$. We omit further details.

Theorem 6.9. $\zeta_3(n,m) = \Omega(n + n^{5/6} m^{1/2} + n^{1/2} m^{5/6} + m)$

Proof: Consider the case $n^{1/3} < m \leq n$. Fix a set P of n points and a set H of m hyperplanes satisfying Lemma 6.8. Any mutually incident subsets of P and H contain either at most one point or at most two planes. Thus, the number of entries in any zero minor of $M(P, H)$ is at most twice the size of the minor. It follows that any zero cover of $M(P, H)$ must have size $\Omega(I(P, H)) = \Omega(n^{5/6}m^{1/2})$. The dual construction gives us a lower bound of $\Omega(n^{1/2}m^{5/6})$ for all m in the range $n \leq m < n^3$, and the trivial lower bound $\Omega(n + m)$ applies for other values of m. □

Lemma 6.10. *Let P be a set of n points and H a set of m planes in $\mathbb{R}^3$, such that every point in P is either on or above every plane in H, and any three planes in H intersect in at most one point. Then $I(P, H) \leq 2(m + n)$.*

Proof: Call any point (resp. plane) *lonely* if it is incident to less than three planes (resp. points). Without loss of generality, we can assume that none of the points in P or planes in H is lonely, since each lonely point and plane contributes at most two incidences.

No point in the interior of of the convex hull of P can be incident to a plane in H. Any point in the interior of a facet of the convex hull can be on at most one plane in H. Consider any point $p \in P$ in the interior of an edge of the convex hull. Any plane containing p also contains the two endpoints of the edge. There cannot be more than two such planes in H, so p must be lonely. It follows that every point in P is a vertex of the convex hull of P.

No plane can contain a point unless it touches the upper envelope of H. Any plane that only contains a vertex of the upper envelope must be lonely. For any plane h that contains only an edge of the envelope, two other planes also contain that edge, and any points on h must also be on the other two planes. Then h must be lonely, since any three planes in H intersect in at most one point. It follows that every plane in H spans a facet of the upper envelope of H. Furthermore, every point in P is a vertex of this upper envelope.

Construct a bipartite graph with vertices P and H and edges corresponding to incident pairs. This bipartite graph is clearly planar, and thus has at most $2(m + n)$ edges. □

Theorem 6.11. $\mu_3^\circ(n, m) = \Omega(n + n^{5/6}m^{1/2} + n^{1/2}m^{5/6} + m)$

Proof: Consider the case $2n^{1/3} < m \leq n$. Fix a set P of n points and a set H of $m/2$ hyperplanes satisfying Lemma 6.8. Replace each plane $h \in H$ with a pair of parallel planes at distance ε on either side of h, for some suitably small constant $\varepsilon > 0$. Call the resulting

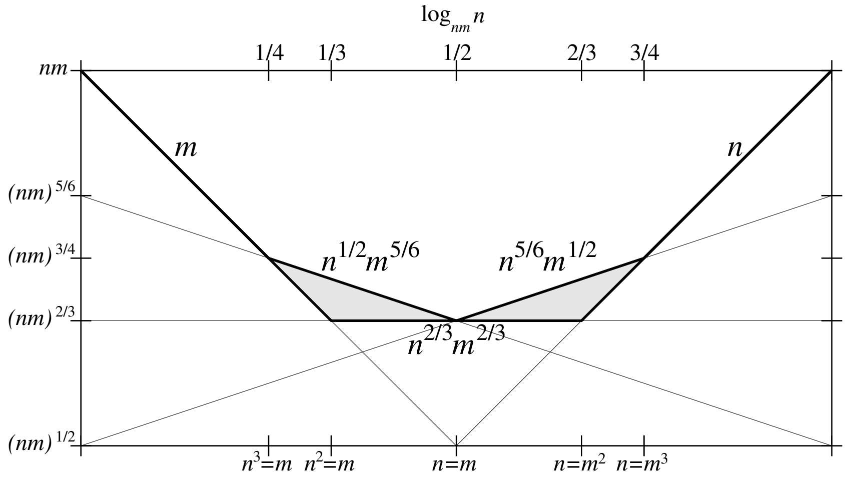


Figure 6.5. Comparison of lower bounds for $\mu_2^\circ(n,m)$ and $\mu_3^\circ(n,m)$. See Theorems 6.6 and 6.11.

set of m planes H_ε. We say that a point is *close* to a plane if the distance between them is exactly ε. There are $\Omega(n^{5/6}m^{1/2})$ close pairs between P and H_ε, and no incidences.

Call a sign matrix *loosely monochromatic* if either none of its entries is $+$ or none of its entries is $-$. For any subsets $P' \subseteq P$ and $H' \subseteq H$, Lemma 6.10 implies that if $M(P', H')$ is loosely monochromatic, then $I(P', H') = O(|P'| + |H'|)$.

For every monochromatic minor of the matrix $M(P, H_\varepsilon)$, the corresponding minor of $M(P, H)$ is loosely monochromatic. Furthermore, there is a one-to-one correspondence between the close pairs in the first minor and the incident pairs in the second. It follows that any monochromatic minor of $M(P, H_\varepsilon)$ orients only a linear number of close pairs. Thus, any monochromatic cover for P and H_ε must have size $\Omega(n^{5/6}m^{1/2})$.

Similar arguments apply to other values of m. □

For the special case $m = \Theta(n)$, this theorem does not improve the $\Omega(n^{4/3})$ bound we derived earlier for the planar case. For all other values of m between $\Omega(n^{1/3})$ and $O(n^3)$, however, the new bound is an improvement. See Figure 6.5.

6.2.4 Higher Dimensions

In order to generalize Lemma 6.8 to arbitrary dimensions, we need the following rather technical lemma. Let us define two series $f_i(t)$ and $F_i(t)$ of polynomials as $f_1(t) = 1$, $f_i(t) = t + i - 2$ for all $i > 1$, and $F_i(t) = \prod_{j=1}^{i-1} f_j(t)$ for all i.

Lemma 6.12. *Let $t_1, t_2, \ldots, t_d$ be distinct real numbers such that $f_j(t_i) \neq 0$ for all $1 \leq i, j \leq d$. The $d \times d$ matrix M, whose (i,j)th entry is $1/f_j(t_i)$, is nonsingular.*

Proof: Let V be the $d \times d$ Vandermonde matrix whose (i,j)th entry is t_i^{j-1}. Since the t_i are distinct, V is nonsingular. We prove the lemma by converting M into V using elementary row and column operations. We transform the matrix inductively, one column at a time. Transforming the first column is trivial.

The inductive step is somewhat easier to understand if we focus on a single row of M, and think of it as a vector of rational functions in some formal variable t, instead of a vector of real numbers. Suppose we have already transformed the first $d-1$ entries inductively, and we are now ready to transform the last entry. The first step is to multiply the entire vector by $f_d(t)$; this ensures that every entry in the vector is a polynomial. By induction, the dth entry is now $F_d(t)$, and for all $j < d$, the jth entry is now $f_d(t) \cdot t^{j-1} = t^j + (d-2)t^{j-1}$. It remains to show that we can transform this vector of polynomials into the vector $(1, t, t^2, \ldots, t^{d-1})$.

Write the coefficients of the polynomials into a $d \times d$ matrix C, whose (i,j)th entry $c_{i,j}$ is the coefficient of t^{i-1} in the jth polynomial. The only nonzero entries in C are the coefficients of $F_d(t)$ in the last column, $(d-2)$'s in rest of the main diagonal, and ones in the next lower diagonal. For example, when $d = 4$, our vector of polynomials is $(t+2, t^2+2t, t^3+2t^2, t^2+t)$, and

$$C = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Recall that the determinant of C is defined as follows.

$$\det C \triangleq \sum_{\pi \in S_d} \left(\operatorname{sgn}(\pi) \prod_{i=1}^{d} c_{i,\pi(i)} \right)$$

The only permutations that contribute to the determinant are those that start down the main diagonal, jump to the last column, and then finish along the lower diagonal. It follows that $\det C = (-1)^d F_d(2-d)$. Since $2-d$ is not a root of $F_d(t)$, we conclude that C is nonsingular.

Thus, there is a series of column operations that convert C into the identity matrix. Since each column of C contains the coefficients of a polynomial in the corresponding column of M, the *same* column operations complete the transformation of M into V. □

Lemma 6.13. *For any* $\lfloor n^{1/d} \rfloor < m$*, there exists a set* P *of* n *points and a set* H *of* m *hyperplanes in* $\mathbb{R}^d$*, such that* $I(P,H) = \Omega(n^{1-2/d(d+1)} m^{2/(d+1)})$ *and any* d *hyperplanes in* H *intersect in at most one point.*

Proof (sketch): First consider the case $d = 4$. Let $h(a,b,c,d;i,j)$ denote the hyperplane passing through the four points

$$(a,b,c,d) \quad (a+i, b+j, c, d) \quad (a+i, b, c+i+j, d) \quad (a+i, b, c, d+2i+j).$$

Let $p = \lfloor n^{1/4} \rfloor$ and $q = \lfloor \alpha (m/p)^{1/5} \rfloor$ for some suitable constant $\alpha > 0$. Then $P = [p]^4$ and H is the set of hyperplanes $h(a,b,c,d;i,j)$ satisfying the following set of conditions.

$$\begin{gathered} i \in [q], \quad j \in [i], \quad j \text{ is odd}, \quad i \perp j \\ a \in [i], \quad b \in [j], \quad c \in [i+j], \quad d \in \Big[\lfloor p/2 \rfloor\Big] \end{gathered}$$

Note that j is odd and relatively prime with i if and only if i, j, $i+j$, and $2i+j$ are pairwise relatively prime. This condition is necessary to establish that the hyperplanes in H are distinct. It follows from straightforward algebraic manipulation, similar to that used in the proof of Lemma 6.8, that $|H| = O(pq^5) = O(m)$ and $I(P,H) = \Omega(p^4 q^2) = \Omega(n^{9/10} m^{2/5})$.

To establish that no four hyperplanes in H intersect in a common line, we examine the intersection of each hyperplane $h(1,1,1,1;i,j) \in H$ with the hyperplane $x_1 = 0$. This intersection is the plane

$$\frac{1}{i} + \frac{x_2 - 1}{j} + \frac{x_3 - 1}{i+j} + \frac{x_4 - 1}{2i+j} = 0.$$

It follows from Lemma 6.12, by setting $t_k = j_k / i_k$ for all k, that no four of these planes are concurrent.

Now consider the d-dimensional case. Let e_i denote the unit vector whose ith coordinate is 1 and whose other coordinates are 0. For any point $x \in \mathbb{R}^d$, let $h(x;i,j)$ denote the affine hull of x and the points $x + ie_1 + ((k-2)i + j)e_k$ for all $2 \le k \le d$. Let $p = \lfloor n^{1/d} \rfloor$ and $q = \lfloor \alpha (m/p)^{1/(d+1)} \rfloor$ for some suitable constant $\alpha > 0$. Then $P = [p]^d$; and H is the set of hyperplanes $h(x;i,j)$ such that $i, j, i+j, 2i+j, \ldots, (d-2)i+j$ are pairwise relatively prime; $x_1 \in [i]$; $x_k \in [(k-2)i+j]$ for all $2 \le k \le d$; and $x_d \in [\lfloor p/2 \rfloor]$.

The relative primality conditions on i and j imply that the hyperplanes in H are distinct. The bounds $|H| = O(pq^{d+1}) = O(m)$ and

$$I(P, H) = \Omega(p^d q^2) = \Omega(n^{1-2/d(d+1)} m^{2/(d+1)})$$

follow from straightforward algebraic manipulation. Finally, Lemma 6.12 implies that any d hyperplanes in H intersect in a unique point. We omit further details. □

Note that the lower bound for dimension d only improves the bound for dimension $d-1$ when $n = \Omega(m^{(d-1)/2})$. Again, using probabilistic counting techniques [46], we can prove an upper bound of $I(P, H) = O(n + n^{(2d-2)/(2d-1)} m^{d/(2d-1)} + m)$ if any d hyperplanes in H intersect in at most one point.

The previous lemma immediately gives us the following lower bound for $\zeta_d(n, m)$.

Theorem 6.14. $\zeta_d(n, m) = \Omega\left(\sum_{i=1}^{d} \left(n^{1-2/i(i+1)} m^{2/(i+1)} + n^{2/(i+1)} m^{1-2/i(i+1)}\right)\right)$

Since our d-dimensional lower bound only improves our $(d-1)$-dimensional lower bound for certain values of n and m, we have combined the lower bounds from all dimensions $1 \le i \le d$ into a single expression. If the relative growth rates of n and m are fixed, the entire sum can be reduced to a single term.

Unfortunately, we are unable to generalize Lemma 6.10 even into four dimensions. Consequently, the best lower bound we can derive for $\mu_d^\circ(n, m)$ for any $d > 3$ derives trivially from Theorem 6.11. The best upper bound we can prove for the number of incidences between n points and m hyperplanes in $\mathbb{R}^4$, where every point is above or on every hyperplane and no four hyperplanes contain a line, is $O(n + n^{2/3} m^{2/3} + m)$. (See [70] for the derivation of a similar upper bound.) No superlinear lower bounds are known in any dimension, so there is some hope for a linear upper bound.

However, we can achieve a superlinear number of incidences in five dimensions, under a weaker combinatorial general position requirement. Thus, unlike in lower dimensions, some sort of *geometric* general position requirement is necessary to keep the number of incidences small.

Lemma 6.15. *For all n and m, there exists a set P of n points and a set H of m hyperplanes in $\mathbb{R}^5$, such that every point is on or above every hyperplane, no two hyperplanes in H contain more than one point of P in their intersection, and $I(P, H) = \Omega(n + n^{2/3} m^{2/3} + m)$.*

Proof: Define the function $\sigma : \mathbb{R}^3 \to \mathbb{R}^6$ as follows.

$$\sigma(x, y, z) = (x^2, y^2, z^2, \sqrt{2}\,xy, \sqrt{2}\,yz, \sqrt{2}\,xz)$$

For any $v, w \in \mathbb{R}^3$, we have $\langle \sigma(v), \sigma(w) \rangle = \langle v, w \rangle^2$, where $\langle \cdot, \cdot \rangle$ denotes the usual inner product of vectors. In a more geometric setting, σ maps points and lines in the plane, represented in homogeneous coordinates, to points and hyperplanes in $\mathbb{R}^5$, also represented in homogeneous coordinates [140]. For any point p and line ℓ in the plane, the point $\sigma(p)$ is incident to the hyperplane $\sigma(\ell)$ if and only if p is incident to ℓ; otherwise, $\sigma(p)$ lies above $\sigma(\ell)$. Thus, we can take P and H to be the images under σ of any sets of n points and m lines with $\Omega(n + n^{2/3}m^{2/3} + m)$ incidences, as given by Lemma 6.4. □

6.2.5 A Lower Bound in the Semigroup Model

Our results immediately imply a lower bound for a variant of the counting version of Hopcroft's problem, in the Fredman/Yao semigroup arithmetic model. The lower bound follows from the following result of Chazelle [38, Lemma 3.3]. (Chazelle's lemma only deals with the case $n = m$, but his proof generalizes immediately to the more general case.)

Lemma 6.16 (Chazelle). *If A is an $n \times m$ incidence matrix with I ones and no $p \times q$ minor of ones, then the complexity of computing Ax over a semigroup is $\Omega(I/pq - n/p)$.*

Theorem 6.17. *Given n weighted points and m hyperplanes in $\mathbb{R}^d$,*

$$\Omega\left(\sum_{i=1}^{d}\left(n^{1-2/i(i+1)}m^{2/(i+1)} + n^{2/(i+1)}m^{1-2/i(i+1)}\right)\right)$$

semigroup operations are required to determine the sum of the weights of the points on each hyperplane, in the worst case.

Proof: The lower bound follows immediately from Lemma 6.13. □

As in Theorem 6.14, we have combined the best lower bounds from several dimensions into a single expression. When $m = \Theta(n)$, this bound simplifies to $\Omega(n^{4/3})$, which already follows immediately from Chazelle's lemma and the Erdős construction. For all other values of m between $\Omega(n^{1/d})$ and $O(n^d)$, however, the new bound is an improvement over any previously known lower bounds for this problem. The best known upper bound is given by Matoušek's algorithm [111].

6.3 Partitioning Algorithms

A *partition graph* is a directed acyclic graph, with one source, called the *root*, and several sinks, or *leaves*. Associated with each non-leaf node v is a set $\mathcal{R}_v$ of *query regions*, satisfying three conditions.

(1) The cardinality of $\mathcal{R}_v$ is at most some constant $\Delta \geq 2$.

(2) Each region in $\mathcal{R}_v$ is connected.

(3) The union of the regions in $\mathcal{R}_v$ is $\mathbb{R}^d$.

(We do *not* require the query regions to be disjoint, convex, simply connected, semi-algebraic, or of constant combinatorial complexity.) In addition, every non-leaf note v is either a *primal node* or a *dual node*, depending on whether its query regions $\mathcal{R}_v$ should be interpreted as a partition of primal or dual space. Each query region in $\mathcal{R}_v$ corresponds to an outgoing edge of v. Thus, the outdegree of the graph is at most Δ.

Given sets P of points and H of hyperplanes as input, a *partitioning algorithm* constructs a partition graph, which can depend arbitrarily on the input, and uses it to drive the following divide-and-conquer process. The algorithm starts at the root and proceeds through the graph in topological order. At every node except the root, points and hyperplanes are passed in along incoming edges from preceding nodes. For each node v, let $P_v \subseteq P$ denote the points and $H_v \subseteq H$ the hyperplanes that reach v; at the root, we have $P_{\text{root}} = P$ and $H_{\text{root}} = H$. At every non-leaf node v, the algorithm partitions the sets P_v and H_v into (not necessarily disjoint) subsets by the query regions $\mathcal{R}_v$ and sends these subsets out along outgoing edges to succeeding nodes. If v is a primal node, then for every query region $R \in \mathcal{R}_v$, the points in P_v that are contained in R and the hyperplanes in H_v that intersect R traverse the outgoing edge corresponding to R. If v is a dual node, then for every query region $R \in \mathcal{R}_v$, the points $p \in P_v$ whose dual hyperplanes p^* intersect R and the hyperplanes $h \in H_v$ whose dual points h^* are contained in R traverse the corresponding outgoing edge. Note that a single point or hyperplane may enter or leave a node along several different edges.

For the purpose of proving lower bounds, the entire running time of the algorithm is given by charging unit time whenever a point or hyperplane traverses an edge. In particular, we do not charge for the construction of the partition graph or its query regions, nor for the time that would be required in practice to decide if a point or hyperplane intersects a query region. As a result, partitioning algorithms are effectively nondeterministic. In principle, the algorithm has "time" to compute the optimal partition graph for its input, and even very similar inputs might result in radically different partition graphs.

To solve Hopcroft's problem, the algorithm reports an incidence if and only if some leaf in the partition graph is reached by both a point and a hyperplane. It is easy to see that if some point and hyperplane are incident, then there is at least one leaf in every partition graph that is reached by both the point and the hyperplane. Thus, for any set P of points and set H of hyperplanes, a partition graph in which no leaf is reached by both a point and a hyperplane provides a *proof* that there are no incidences between P and H.

In this section, we derive lower bounds for the worst-case running time of partitioning algorithms that solve Hopcroft's problem. With the exception of the basic lower bound of $\Omega(n \log m + m \log n)$, which in light of Theorem 6.3 we must prove directly, our lower bounds are derived from the cover size bounds in Section 6.2. In the Section 6.4, we will describe how existing algorithms for Hopcroft's problem fit into our computational framework.

6.3.1 The Basic Lower Bound

Theorem 6.18. *Any partitioning algorithm that solves Hopcroft's problem in any dimension must take time* $\Omega(n \log m + m \log n)$ *in the worst case.*

Proof: It suffices to consider the following configuration, where n is a multiple of m. P consists of n points on some vertical line in $\mathbb{R}^d$, say the x_d-axis, and H consists of m hyperplanes normal to that line, placed so that n/m points lie between each hyperplane and the next higher hyperplane, or above the top hyperplane. (We implicitly used a one-dimensional version of this configuration to prove the lower bound in Theorem 6.3.) For each point, call the hyperplane below it its *partner*. Each hyperplane is a partner of n/m points.

Let G be the partition graph generated by some partitioning algorithm. Recall that the out-degree of every node in G is at most Δ. The *level* of any node in G is the length of the shortest path from the root to that node. There are at most Δ^k nodes at level k. We say that a node v *splits* a point-hyperplane pair if both the point and the hyperplane reach v, and none of the outgoing edges of v is traversed by both the point and the hyperplane. In order for the algorithm to be correct, every point-hyperplane pair must be split. Finally, we say that a hyperplane h is *active at level* k if none of the nodes in the first k levels split h from any of its partners.

Suppose v is a primal node. For each hyperplane h that v splits from one of its partner points p, mark some query region in $\mathcal{R}_v$ that contains p but misses h. The marked region lies completely above h, but not completely above any hyperplane higher than h. It follows that the same region cannot be marked more than once. Since there are at most Δ regions, at most Δ hyperplanes become inactive. By similar arguments, if v is a dual node, then v splits at most Δ points from their partners.

Thus, the number of hyperplanes that are inactive at level k is less than Δ^{k+2}. In particular, at level $\lfloor \log_\Delta m \rfloor - 3$, at least $m(1-1/\Delta)$ hyperplanes are still active. It follows that at least $n(1-1/\Delta)$ points each traverse at least $\lfloor \log_\Delta m \rfloor - 3$ edges. We conclude that the total running time of the algorithm is at least

$$n(1-1/\Delta)(\lfloor \log_\Delta m \rfloor - 3) = \Omega(n \log m).$$

Similar arguments establish a lower bound of $\Omega(m \log n)$ when $n < m$. $\square$

6.3.2 The Lower Bound for the Decision Problem

Let $T_{\mathcal{A}}(P, H)$ denote the running time of an algorithm $\mathcal{A}$ that solves Hopcroft's problem in $\mathbb{R}^d$ for some d, given points P and hyperplanes H as input.

Theorem 6.19. *Let $\mathcal{A}$ be a partitioning algorithm that solves Hopcroft's problem, and let P be a set of points and H a set of hyperplanes such that $I(P, H) = 0$. Then $T_{\mathcal{A}}(P, H) = \Omega(\mu(P, H))$.*

Proof: Recall that the running time $T_{\mathcal{A}}(P, H)$ is defined in terms of the edges of the partition graph as follows.

$$T_{\mathcal{A}}(P, H) \triangleq \sum_{\text{edge } e} \Big(\#\text{ points traversing } e + \#\text{ hyperplanes traversing } e\Big)$$

We say that a point or hyperplane *misses* an edge from v to w if it reaches v but does not traverse the edge. (It might still reach w by traversing some other edge.) For every edge that a point or hyperplane traverses, there are at most $\Delta - 1$ edges that it misses.

$$\Delta \cdot T_{\mathcal{A}}(P, H) \geq \sum_{\text{edge } e} \big(\#\text{ points traversing } e + \#\text{ hyperplanes traversing } e$$
$$+ \#\text{ points missing } e + \#\text{ hyperplanes missing } e\big)$$

Call any edge that leaves a primal node a primal edge, and any edge that leaves a dual node a dual edge.

$$\Delta \cdot T_{\mathcal{A}}(P, H) \geq \sum_{\text{primal edge } e} \Big(\#\text{ points traversing } e + \#\text{ hyperplanes missing } e\Big)$$
$$+ \sum_{\text{dual edge } e} \Big(\#\text{ hyperplanes traversing } e + \#\text{ points missing } e\Big)$$

Consider, for some primal edge e, the set P_e of points that traverse e and the set H_e of hyperplanes that miss e. The edge e is associated with some query region R, such that every point in P_e is contained in R, and every hyperplane in H_e is disjoint from R. Since R is connected, it follows immediately that the relative orientation matrix $M(P_e, H_e)$ is simple. Similarly, for any dual edge e, the relative orientation matrix of the set of points that miss e and hyperplanes that traverse e is also simple.

Now consider any point $p \in P$ and hyperplane $h \in H$. Since $\mathcal{A}$ correctly solves Hopcroft's problem, no leaf is reached by both p and h. It follows that some node v splits p and h. If v is a primal node, then h misses the outgoing primal edges that p traverses. If v is a dual node, then p misses the outgoing dual edges that h traverses.

Thus, we can associate a simple minor with every edge in the partition graph, and this collection of minors covers the relative orientation matrix $M(P, H)$. Furthermore, the size of this simple cover is exactly the lower bound we have for $\Delta \cdot T_{\mathcal{A}}(P, H)$ above. Splitting each simple minor into monochromatic minors at most doubles the size of the cover. Since the size of the resulting monochromatic cover must be at least $\mu(P, H)$, we conclude that $T_{\mathcal{A}}(P, H) \geq \mu(P, H)/2\Delta$. □

Corollary 6.20. *The worst-case running time of any partitioning algorithm that solves Hopcroft's problem in* $\mathbb{R}^d$ *is* $\Omega(n \log m + n^{2/3}m^{2/3} + m \log n)$ *for* $d = 2$ *and* $\Omega(n \log m + n^{5/6}m^{1/2} + n^{1/2}m^{5/6} + m \log n)$ *for all* $d \geq 3$.

Proof: Theorems 6.18 and 6.19 imply that the worst case running time is $\Omega(n \log m + \mu_d^\circ(n, m) + n \log m)$. Thus, Theorem 6.6 gives the planar lower bound, and Theorem 6.11 gives us the lower bound in higher dimensions. □

We emphasize that the condition $I(P, H) = 0$ is necessary for this lower bound to hold. If there is an incidence, then the trivial partitioning algorithm "detects" it. The partition graph consists of a single leaf, and since that leaf is reached by every point and every hyperplane, the algorithm correctly reports an incidence.

6.3.3 The Lower Bound for the Counting Problem

Every partitioning algorithm assumes that a point and hyperplane are incident if they reach the same leaf in its partition graph. Thus, the number of incidences associated with a leaf is the product of the number of points that reach it and the number of hyperplanes that reach it. To solve the counting version of Hopcroft's problem, a partitioning algorithm returns as its output the sum of these products over all leaves in its partition graph. In order for this output to be correct, the algorithm must ensure that every non-incident point-hyperplane pair is separated and that every incident pair reaches exactly one leaf. Since every incident point-hyperplane pair is guaranteed to reach at least one leaf, it is not possible for a partitioning algorithm to count too few incidences.

Theorem 6.21. *Let $\mathcal{A}$ be a partitioning algorithm that solves the counting version of Hopcroft's problem, and let P be a set of points and H a set of hyperplanes. Then $T_{\mathcal{A}}(P, H) = \Omega(\mu(P, H))$.*

Proof: We follow the proof for the decision lower bound almost exactly. We associate a simple minor with every edge just as before. We also associate a monochromatic minor with every leaf, consisting of all points and hyperplanes that reach the leaf. Every non-incident point-hyperplane pair is represented in some edge minor, and every incident pair in exactly one leaf minor. Thus, the minors form a simple cover. The total size of the leaf minors is certainly less than $T_{\mathcal{A}}(P, H)$, since every point and hyperplane that reaches a leaf must traverse one of the leaf's incoming edges. The total size of the edge minors is at most $\Delta \cdot T_{\mathcal{A}}(P, H)$, as established previously. Splitting each edge minor into monochromatic minors at most doubles their size. Thus, we get a monochromatic cover of size at most $(2\Delta + 1)T_{\mathcal{A}}(P, H)$, which implies $T_{\mathcal{A}}(P, H) \geq \mu(P, H)/(2\Delta + 1)$. $\square$

Corollary 6.22. *The worst-case running time of any partitioning algorithm that solves the counting version of Hopcroft's problem in* $\mathbb{R}^d$ *is*

$$\Omega\left(n\log m + \sum_{i=2}^{d}\left(n^{1-2/i(i+1)}m^{2/(i+1)} + n^{2/(i+1)}m^{1-2/i(i+1)}\right) + m\log n\right).$$

See the remark after Theorem 6.14.

We can prove the following stronger lower bound by only paying attention to the minors induced at the leaves. We define an *unbounded partition graph* to be just like a partition graph except that we place no restrictions on the number of query regions associated with each node. Call the resulting class of algorithms *unbounded partitioning algorithms*. Note that such an algorithm can solve the decision version of Hopcroft's problem in linear time.

Theorem 6.23. *Let* $\mathcal{A}$ *be an unbounded partitioning algorithm that solves the counting version of Hopcroft's problem, and let* P *be a set of points and* H *a set of hyperplanes. Then* $T_{\mathcal{A}}(P, H) = \Omega(\zeta(P, H))$.

Proof: We associate a zero minor with every leaf, and these minors form a zero cover. The total size of the leaf minors is less than $T_{\mathcal{A}}(P, H)$, since every point and hyperplane that reaches a leaf must traverse one of the leaf's incoming edges. □

The following corollary is now immediate.

Corollary 6.24. *The worst-case running time of any unbounded partitioning algorithm that solves the counting version of Hopcroft's problem in* $\mathbb{R}^d$ *is*

$$\Omega\left(\sum_{i=1}^{d}\left(n^{1-2/i(i+1)}m^{2/(i+1)} + n^{2/(i+1)}m^{1-2/i(i+1)}\right)\right).$$

6.3.4 Containment Shortcuts Don't Help

We might consider adding the following containment shortcut to our model. Suppose that while partitioning points and hyperplanes at a primal node, the algorithm discovers that a query region R is completely contained in some hyperplane h. Then we know immediately that any point contained in R is incident to h. Rather than sending h down the edge corresponding to R, the algorithm could increment a running counter for each

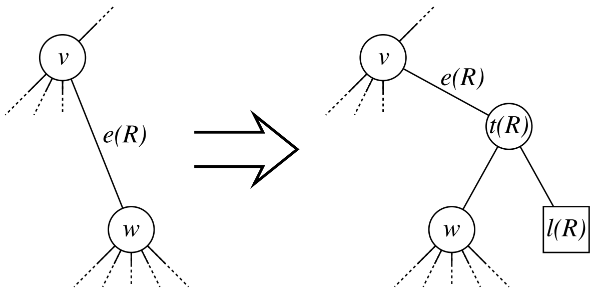


Figure 6.6. Eliminating containment shortcuts.

point in R. We can apply a symmetric shortcut at each dual node, potentially reducing the number of points traversing each dual edge. In addition to charging for edge traversals, we now also charge unit time whenever an algorithm discovers that a hyperplane (either primal or dual) contains a query region.

Clearly, adding this shortcut can only decrease the running time of any partitioning algorithm. However, for any algorithm that uses this shortcut, we can derive an equivalent algorithm without shortcuts that is slower by only a small constant factor, as follows.

If a hyperplane h contains a query region R, then it must also contain $\mathrm{aff}(R)$, the affine hull of R. We can reverse the containment relation by applying a duality transformation — the dual point h^* is contained in the dual flat $(\mathrm{aff}(R))^*$. Similarly, if a point p is contained in R, then $(\mathrm{aff}(R))^* \in p^*$.

For each node v in the partition graph of the shortcut algorithm, and each query region $R \in \mathcal{R}_v$, we modify the graph as follows. Let $e(R)$ be the edge of the partition graph corresponding to R, and let w be the destination of this edge. We introduce two new nodes, a "test" node $t(R)$ and a leaf $\ell(R)$. If v is a primal node, then $t(R)$ is a dual node, and vice versa. The query subdivision $\mathcal{R}_{t(R)}$ consists of exactly two regions: $(\mathrm{aff}(R))^*$ and $\mathbb{R}^d \setminus (\mathrm{aff}(R))^*$, whose corresponding edges point to $\ell(R)$ and w, respectively. Finally, we redirect $e(R)$ so that it points to $t(R)$. See Figure 6.6. The new algorithm $\mathcal{A}'$ uses this modified partition graph, without explicitly checking for containments.

The new node $t(R)$ separates the hyperplanes that contain R from the hyperplanes that merely intersect it. Any point contained in R reaches both w and $\ell(R)$. Thus, the new algorithm reports or counts exactly the same incidences as the original shortcut algorithm. We easily verify that the running time of the new algorithm is at most three times the running time of the shortcut algorithm.

6.4 Real Algorithms for Hopcroft's Problem

Real algorithms for Hopcroft's problem all employ roughly the same divide-and-conquer strategy. Each algorithm divides space into a number of regions, determines which points and hyperplanes intersect each region, and recursively solves the resulting subproblems. In some cases [32, 66, 43], the number of regions used at each level of recursion is a constant, and these algorithms fit naturally into the partitioning algorithm framework.

For most algorithms, however, the number of regions is a small polynomial function of the input size, and not a constant as required by the definition of the partitioning model. However, we can still model most, if not all, of these algorithms by partitioning algorithms.

In order to determine which points lie in which regions, each of these algorithms constructs a (possibly trivial) point location data structure. Each node in this data structure partitions space into a constant number of simple regions, and for each region, there is a pointer to another node in the data structure. Each leaf in the data structure corresponds to one of the high-level query regions. Composing all the point location data structures used by the algorithm in all recursive subproblems gives us the algorithm's partition graph. Many of these algorithms alternate between primal and dual spaces at various levels of recursion [35, 111]. The data structures used in primal space give us the primal nodes in the partition graph, and the data structures used in dual space give us the dual nodes.

What about the hyperplanes? Many algorithms also use the point location data structures to determine the regions hit by each hyperplane. Algorithms of this type fit into our model perfectly. In particular, Matoušek's algorithm [111], which is based on Chazelle's hierarchical cuttings [35] and is the fastest algorithm known, can be modeled this way. Matoušek's algorithm and Theorem 6.21 immediately give us the following theorem.

Theorem 6.25. $\mu_d(n, m) = O\left(m \log n + n^{d/(d+1)} m^{d/(d+1)} 2^{O(\log^*(n+m))} + n \log m\right)$

However, other algorithms do not use the point location data structure to locate hyperplanes, at least not at all levels of recursion. In these algorithms [63, 1], the query regions form a decomposition of space into cells of constant complexity, typically simplices or trapezoids. The algorithms determine which cells a given hyperplane hits by iteratively "walking" through the cells. At each cell that the hyperplane intersects, the algorithm can determine in constant time which of the neighboring cells are also intersected, by checking each of the boundary facets.

In many cases, modifying such an algorithm to directly use the point location data structure instead of the iterative procedure increases the running time by only a constant factor. If the current point location data structure locates the hyperplanes too slowly, we may be able to replace it with a different data structure that supports fast hyperplane location, again without increasing the asymptotic running time. We could use, for example, the randomized incremental construction of Seidel [135] in the plane, or the hierarchical cuttings data structure of Chazelle [35] in higher dimensions. The modified algorithm can then be described as a partitioning algorithm.

Other algorithms construct a point location data structure for the arrangement of the *entire* set of hyperplanes [51, 66, 35]. Usually, this is done only when the number of hyperplanes is much smaller than the number of points. In this case, the algorithm doesn't need to locate the hyperplanes at all! Again, however, we can modify the algorithm so that it uses a point location data structure that allows efficient hyperplane location as well, and artificially locates the hyperplanes. If we use an appropriate data structure, the running time will only increase by a constant factor.

These arguments are admittedly ad hoc. Modifying the partitioning model to *naturally* include algorithms that use different strategies for point and hyperplane location, or strengthening our lower bounds to a similar model that does not require constant-degree partitioning, is an interesting open problem.

Finally, a few algorithms partition the points or the hyperplanes *arbitrarily* into subsets, without using geometric information of any kind [62, p. 350],[51]. In this case, every hyperplane becomes part of every subproblem. In order to take algorithms of this kind into account, we must strengthen our model of computation by adding a new type of node that partitions either the points or the hyperplanes (but not both!) into arbitrary subsets at no cost. Lemmas 6.19 and 6.21 still hold in this stronger model, since the new nodes cannot separate any point from any hyperplane. However, Theorem 6.18 does not hold in this model; for example, we can solve Hopcroft's problem in $\mathbb{R}^d$ in time $O(n+m^{d+1})$ by "arbitrarily" partitioning the points so that each subset is contained in a single cell of the hyperplane arrangement.

6.5 Conclusions and Open Problems

We have proven new lower bounds on the complexity of Hopcroft's problem that apply to a broad class of geometric divide-and-conquer algorithms. Our lower bounds were developed in two stages. First, we derived lower bounds on the minimum size of a monochromatic cover in the worst case. Second, we showed that the running time of any partitioning algorithm is bounded below by the size of some monochromatic cover of its input.

A number of open problems remain to be solved. The most obvious is to improve the lower bounds, in particular for the case $n = m$. The true complexity almost certainly increases with the dimension, but the best lower bound we can achieve in higher dimensions comes trivially from the two-dimensional case. Is there a configuration of n points and n planes in $\mathbb{R}^3$ whose minimum monochromatic cover size is $\Omega(n^{3/2})$?

One possible approach is to consider restrictions of the partitioning model. Can we achieve better bounds if we only consider algorithms whose query regions are convex? What if the query regions at every node are distinct? What if the running time depends on the complexity of the query regions? In the next chapter, we prove slightly better lower bounds for Hopcroft's problem in higher dimensions by restricting the query regions to convex polyhedra with constant complexity.

The class of partitioning algorithms is general enough to directly include many, but not all, existing algorithms for solving Hopcroft's problem. The model requires that a single data structure be used to determine which points and hyperplanes intersect each query region, but many algorithms use a tree-like structure to locate the points and an iterative procedure to locate the hyperplanes. We can usually modify such algorithms so that they do fit our model, at the cost of only a constant factor in their running time, but this is a rather ad hoc solution. Any extension of our lower bounds to a more general model, which would explicitly allow different strategies for locating points and hyperplanes, would be interesting.

Our techniques imply lower bounds for several other problems similar to Hopcroft's problem [78]. Unfortunately, the partitioning algorithm model is specifically tailored to detect intersections or containments between pairs of objects, and there are a number of similar geometric problems for which the model simply does not apply. We mention one specific example, the *cyclic overlap problem*. Given a set of nonintersecting line segments in $\mathbb{R}^3$, does any subset form a cycle with respect to the "above" relation? The fastest known algorithm for this problem, due to de Berg, Overmars, and Schwarzkopf [53], runs in time $O(n^{4/3+\varepsilon})$, using a divide-and-conquer strategy very similar to algorithms for Hopcroft's problem. In the algebraic decision tree model, the cyclic overlap problem is at least as hard as Hopcroft's problem [75]. Apparently, however, this problem cannot even be solved by a partitioning algorithm, since the answer might depend on arbitrarily large tuples of segments, arbitrarily far apart. Extending our lower bounds into more traditional models of computation remains an important and very difficult open problem.

"Here's to the new golden age of mathematics," Lord Vickers cried suddenly.
There was a chorus of approving remarks.
"That was the real thing!"
"Plenty of logic."
"And so many symbols!"
Lord Vickers smiled at me from across the room.
"There'll be a place for you at my new institute, Fletcher."
I took a glass of sherry.

— Rudy Rucker, "A New Golden Age", 1981

Chapter 7

Halfspace Emptiness

The halfspace emptiness problem asks, given a set of points and a set of halfspaces, whether any halfspace contains a point. In this chapter, we will consider the following formulation of the problem: Given a set of points and hyperplanes, is every point above every hyperplane? Using linear programming [48, 109, 113, 136], we can decide in linear time whether the union of a set of halfspaces is $\mathbb{R}^d$. If it is, then *every* input point must lie in a halfspace; if not, then by an appropriate projective transformation, we can ensure that the halfspaces miss the point $(0, 0, \ldots, 0, \infty)$. If we use the duality transformation $(a_1, a_2, \ldots, a_d) \longleftrightarrow \sum_{i=1}^{d-1} a_i x_i = a_d + x_d$, then a point p is above a hyperplane h if and only if the dual point h^* is above the dual point p^*. Thus, in this formulation, the halfspace emptiness problem is self-dual.

The best known algorithms for this problem were developed for its online version: Given a set of n points, preprocess it to answer halfspace emptiness (or reporting) queries. In two and three dimensions, we can easily build a linear-size data structure, in $O(n \log n)$ time, that allows halfspace emptiness queries to be answered in logarithmic time [5, 39, 57]. In higher dimensions, a randomized algorithm due to Clarkson [47] answers halfspace emptiness queries in time $O(\log n)$ after $O(n^{\lfloor d/2 \rfloor + \varepsilon})$ preprocessing time. Matoušek [107] describes two halfspace emptiness data structures, one answering queries in time $O(n^{1-1/\lfloor d/2 \rfloor} \operatorname{polylog} n)$ time after $O(n \log n)$ preprocessing time, and the other answering queries in time $O(n^{1-1/\lfloor d/2 \rfloor} 2^{O(\log^* n)})$ after $O(n^{1+\varepsilon})$ preprocessing time. Combining Clarkson's and Matoušek's data structures, for a fixed parameter $n \le s \le n^{\lfloor d/2 \rfloor}$, one can answer queries in time $O((n \log n)/s^{1/\lfloor d/2 \rfloor})$ after $O(s \operatorname{polylog} n)$ preprocessing time [107, 3, 29]. For extensions and applications of halfspace range reporting, see [3, 4, 27, 29, 110, 108].

Given n points and m halfspaces, we can solve the offline halfspace emptiness problem in time

$$O\left(n \log m + (nm)^{\lfloor d/2 \rfloor / (\lfloor d/2 \rfloor + 1)} \operatorname{polylog}(n+m) + m \log n\right),$$

using either Clarkson's data structure or one of Matoušek's data structures, depending on the relative growth rates of n and m. In two and three dimensions, the time bound simplifies to $O(n \log m + m \log n)$. If $n > m$, we actually solve the problem in the dual, by building a data structure to report if any halfspace contains a query point.

The only lower bound previously known for this problem is $\Omega(n \log m + m \log n)$, in the algebraic decision tree or algebraic computation tree models, by reduction from the set intersection problem [138, 16]. Thus, the two- and three-dimensional algorithms are optimal, but there is still a large gap in dimensions four and higher.

In this chapter, we develop a lower bound of $\Omega(n \log m + n^{2/3} m^{2/3} + m \log n)$ on the complexity of the halfspace emptiness problem in $\mathbb{R}^5$, matching known upper bounds up to polylogarithmic factors. We obtain marginally better bounds in dimensions 9 and higher. Using similar techniques, we also prove slightly better bounds for Hopcroft's problem in dimensions four and higher. Our lower bounds apply to *polyhedral partitioning algorithms*, a restriction of the class of partitioning algorithms introduced in the previous chapter. Informally, a polyhedral partitioning algorithm covers space with a constant number of constant-complexity polyhedra, determines which points and halfspaces intersect which polyhedra, and recursively solves the resulting subproblems.

The basic approach is the same as in the previous chapter. We first define *polyhedral covers*, and develop lower bounds on their combinatorial complexity. The main result of this chapter (Theorem 7.9) states that the running time of a polyhedral partitioning algorithm is bounded below by the polyhedral cover size of its input. The $\Omega(n^{4/3})$ lower bound then follows from the construction of a set of points and hyperplanes in $\mathbb{R}^5$, with all the points above all the hyperplanes, whose every polyhedral cover is that large.

7.1 Projective Polyhedra

Our lower bound argument relies heavily on certain properties of convex polytopes and polyhedra. Many of these properties are more easily proved, and have fewer special cases, if we state and prove them in projective space rather than affine Euclidean space. In particular, developing these properties in projective space allows us to more easily deal with unbounded and degenerate polyhedra and duality transformations. Everything we describe in this section can be formalized algebraically in the language of polyhedral cones and linear subspaces one dimension higher; we will give a much less formal, purely geometric treatment. For more technical details, we refer the reader to Chapters 1 and 2 of Ziegler's lecture notes [161].

The projective space $\mathbb{RP}^d$ can be defined as the set of lines through the origin in $\mathbb{R}^{d+1}$. Every k-dimensional linear subspace of $\mathbb{R}^{d+1}$ induces a $(k-1)$-dimensional *flat* f in $\mathbb{RP}^d$, and its orthogonal complement induces the *dual flat* f^*.

A *projective polyhedron* is a single closed cell, not necessarily of full dimension, in the arrangement of a finite number of hyperplanes in $\mathbb{RP}^d$. A *projective polytope* is a simply-connected projective polyhedron, or equivalently, a projective polyhedron that is disjoint from some hyperplane (not necessarily in its defining arrangement). Every projective polyhedron is (the closure of) the image of a convex polyhedron under some projective transformation, and every projective polytope is the image of a convex polytope. Every flat is also a projective polyhedron.

The *projective span* (or projective hull) of any subset $X \subseteq \mathbb{RP}^d$, denoted $\mathrm{span}(X)$, is the projective subspace of minimal dimension that contains it. The *relative interior* of a projective polyhedron is its interior in the subspace topology of its projective hull. A hyperplane *supports* a polyhedron if it intersects the polyhedron but not its relative interior. A flat has no supporting hyperplanes.

A *proper face* of a projective polyhedron is the intersection of the polyhedron and one or more of its supporting hyperplanes. Every proper face of a polyhedron is a lower-dimensional polyhedron. A *face* of a polyhedron is either a proper face or the polyhedron itself. We write $\Phi \leq \Pi$ to denote that a polyhedron Φ is a face of another polyhedron Π. The *dimension* of a face is the dimension of its projective hull. The dimension of the empty set is taken to be -1. The faces of a polyhedron form a lattice under inclusion. Every projective polyhedron has a face lattice isomorphic to that of a convex polytope, possibly of lower dimension.

The *apex* of a polyhedron Π is the intersection of all its supporting hyperplanes, or equivalently, its unique face of minimum dimension. The apex is empty if and only if the polyhedron is a polytope but not a single point; the apex is the whole polyhedron if and only if the polyhedron is a flat.

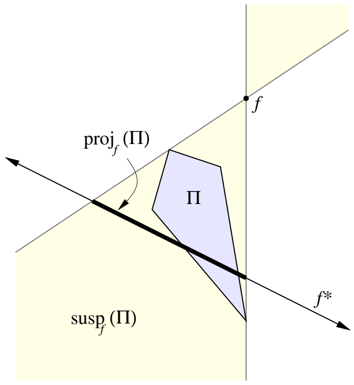


Figure 7.1. The suspension and projection of a polygon by a point.

The *dual* of a polyhedron Π, denoted Π^*, is defined to be the set of points whose dual hyperplanes intersect Π in one of its faces:

$$\Pi^* \triangleq \{p \mid (p^* \cap \Pi) \leq \Pi\}.$$

In other words, $h^* \in \Pi^*$ if and only if h either contains Π, supports Π, or completely misses Π. This definition generalizes both the polar of a convex polytope and the projective dual of a flat. We easily verify that Π^* is a projective polyhedron whose face lattice is the inverse of the face lattice of Π. In particular, Π and Π^* have the same number of faces. See [161, pp. 59–64] and [140, pp. 143–150] for similar definitions and more technical details.

For any subset $X \subseteq \mathbb{RP}^d$ and any flat f, the *suspension of* X *by* f, denoted $\mathrm{susp}_f(X)$, is formed by replacing each point in X by the span of that point and f:

$$\mathrm{susp}_f(X) \triangleq \bigcup_{p \in X} \mathrm{span}(p \cup f).$$

The suspension of a subset of projective space corresponds to an infinite cylinder over a subset of an affine space, at least when the apex of suspension is "at infinity".[1] The *projection of* X *by* f, denoted $\mathrm{proj}_f(X)$, is the intersection the suspension and the dual flat f^*:

$$\mathrm{proj}_f(X) \triangleq f^* \cap \mathrm{susp}_f(X),$$

In particular, $\mathrm{susp}_f(X)$ is the set of all points in $\mathbb{RP}^d$ whose projection by f is in $\mathrm{proj}_f(X)$. The projection of a subset of projective space corresponds to the orthogonal projection of a subset of affine space onto an affine subspace. See Figure 7.1.

[1] Ziegler [161, p. 33] calls this the *elimination* of X.

7.2 Polyhedral Separation

Let P be a set of points, let H be a set of hyperplanes, and let Π be a projective polyhedron in $\mathbb{RP}^d$. We say that Π *separates* P and H if Π contains P and the dual polyhedron Π^* contains the dual points H^*; that is, any hyperplane in H either contains Π, supports Π, or misses Π entirely. Both P and H may intersect the relative boundary of Π. We say that P and H are r-*separable* if there is a polyhedron with at most r faces that separates them.

The proofs of Theorems 6.6 and 6.11 implicitly relied on the following trivial observation: if we perturb a configuration of points and hyperplanes just enough to remove any incidences, and the resulting configuration is monochromatic, then the original configuration must have been loosely monochromatic. The following technical lemma establishes the corresponding, but no longer trivial, property of r-separable configurations. Informally, if a configuration is not r-separable, then arbitrarily small perturbations cannot make it r-separable. First-time readers are encouraged to skip the proof.

Technical Lemma 7.1. *Let H be a set of m hyperplanes in $\mathbb{RP}^d$. For all r, the set of point configurations $P \in (\mathbb{RP}^d)^n$ such that P and H are r-separable is topologically closed.*

Proof: There are two cases to consider: either the hyperplanes in H do not have a common intersection, or they intersect in a common flat. The proof of the second case relies on the first.

Case 1 ($\bigcap \mathbf{H} = \emptyset$):

Any polyhedron that separates P and H must be completely contained in a closed d-cell $\mathcal{C}$ of the arrangement of H. Thus, it suffices to show, for each cell $\mathcal{C}$, that the set of n-point configurations contained in $\mathcal{C}$ and r-separable from H is topologically closed. Our approach is to show that this set is actually a compact semialgebraic set.

Fix a cell $\mathcal{C}$. Since every hyperplane in H passes through the apex of $\mathcal{C}$, both $\mathcal{C}$ and any polyhedra it contains must be polytopes. By choosing an appropriate hyperplane "at infinity" that misses $\mathcal{C}$, we can treat $\mathcal{C}$ and any polytopes it contains as *convex* polytopes in $\mathbb{R}^d$.

Let $A = \{a_1, a_2, \ldots, a_\nu\}$ and $B = \{b_1, b_2, \ldots, b_\nu\}$ be two indexed sets of points in $\mathbb{R}^d$, for some integer ν. We say that A *is simpler than* B, written $A \sqsubseteq B$, if for any subset of B contained in a facet of conv(B), the corresponding subset of A is contained in a facet of conv(A).[2] Equivalently, $A \sqsubseteq B$ if and only if for $d+1$ points in B, d of whose vertices lie on a facet of conv(B), the corresponding simplex in A either has the same orientation or is degenerate. Simpler point sets have less complex convex hulls — if $A \sqsubseteq B$, then conv(A) has no more vertices, facets, or faces than conv(B). If both $A \sqsubseteq B$ and $B \sqsubseteq A$, then the convex hulls of A and B are combinatorially equivalent.

If B is fixed, then the relation $A \sqsubseteq B$ can be encoded as the conjunction of at most $O(\nu^{\lfloor d/2 \rfloor + 1})$ algebraic inequalities of the form

$$\begin{vmatrix} a_{i_0 0} & a_{i_0 1} & \cdots & a_{i_0 d} \\ a_{i_1 0} & a_{i_1 1} & \cdots & a_{i_1 d} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i_d 0} & a_{i_d 1} & \cdots & a_{i_d d} \end{vmatrix} \Diamond\, 0,$$

where $(a_{i_j 0}, a_{i_j 1}, \ldots, a_{i_j d})$ are the homogeneous coordinates of the point $a_{i_j} \in A$, and $\Diamond$ is either $\geq$, $=$, or $\leq$. In every such inequality, the corresponding points $b_{i_1}, b_{i_0}, \ldots, b_{i_d}$ (but not necessarily b_{i_0}) all lie on a facet of conv(B). For every d-tuple of points in B contained in a facet of conv(B), there are $\nu - d$ such inequalities, one for every other point. (If we replace the loose inequalities $\leq, \geq$ with strict inequalities $<, >$, the resulting expression encodes the combinatorial equivalence of conv(A) and conv(B).)

We can encode the statement "P is contained in $\mathcal{C}$ and is r-separable from H" as the following elementary formula:

$$\bigvee_{\nu=1}^{r} \quad \bigvee_{\substack{B \in (\mathbb{R}^d)^\nu \\ \text{conv}(B) \text{ has at most } r \text{ faces}}} \left\{ \begin{array}{l} \exists\, a_1, a_2, \ldots, a_\nu \in \mathcal{C} : \\ \exists\, \lambda_1, \lambda_2, \ldots, \lambda_n \in [0,1]^\nu : \\ \qquad (A \sqsubseteq B) \;\wedge\; \displaystyle\bigwedge_{i=1}^{n} \left(\sum_{j=1}^{\nu} a_j \lambda_{ij} = p_i \;\wedge\; \sum_{j=1}^{\nu} \lambda_{ij} = 1 \right) \end{array} \right\} \tag{7.1}$$

Equivalently, in English:

> For some integer ν, and for some set B of ν points whose convex hull has at most r faces, there exists a set A of ν points in $\mathcal{C}$, such that A is simpler than B (so conv(A) has at most r faces) and every point in P is a convex combination of points in A (with barycentric coordinates Λ).

[2] Every set of points is simpler than itself. It would be more correct, but also unwieldier, to say "A is at least as simple as B".

Since there are only a finite number of combinatorial equivalence classes of convex polytopes with ν vertices [89], the formula is finite, and therefore defines a semi-algebraic set. It remains only to show that this set is closed.

For any fixed ν and B, the set of configurations $P \times A \times \Lambda \in (\mathbb{R}^d)^n \times \mathcal{C}^\nu \times ([0,1]^\nu)^n$ that satisfy the subexpression

$$(A \sqsubseteq B) \ \wedge \ \bigwedge_{i=1}^{n} \left(\sum_{j=1}^{\nu} a_j \lambda_{ij} = p_i \ \wedge \ \sum_{j=1}^{\nu} \lambda_{ij} = 1 \right)$$

is the intersection of the closed convex polytope $\mathcal{C}^{n+\nu} \times [0,1]^{\nu n}$, at most $(\nu - d)r$ closed algebraic halfspaces, νn quadratic surfaces, and νn hyperplanes, and is therefore closed and bounded. It follows that the set of point configurations P that satisfy the subexpression of (7.1) in braces is the projection of a compact set, and is therefore also compact. Finally, the set of configurations P satisfying the entire formula (7.1) is the union of a finite number of compact sets, and therefore must be compact.

This completes the proof of Case 1.

Case 2 ($\bigcap H \neq \emptyset$):

The previous argument will not work in this case, because the cells in the arrangement of H are not simply connected, and thus are not polytopes.

Let P be an arbitrary set of n points in $\mathbb{RP}^d$, such that P and H are *not* r-separable. To prove the lemma, it suffices to show that there is an open neighborhood $\mathcal{U} \in (\mathbb{RP})^d$ with $P \in \mathcal{U}$, such that, for all $Q \in \mathcal{U}$, Q and H are not r-separable.

Let $f = \bigcap H$, and let f^* be its dual flat. Without loss of generality, suppose the points $p_1, p_2, \ldots, p_m \in P$ are disjoint from f, and the points $p_{m+1}, \ldots, p_n \in P$ are contained in f. Denote these two subsets of P by $P \setminus f$ and $P \cap f$, respectively. Either subset may be empty. Note that $\mathrm{proj}_f(P) = \mathrm{proj}_f(P \setminus f)$, since by definition $\mathrm{proj}_f(f)$ is empty.

If any polyhedron Π separates P and H, then its projection $\mathrm{proj}_f(\Pi)$ separates the projected points $\mathrm{proj}_f(P)$ and the lower dimensional hyperplanes $H \cap f^*$. Conversely, if any polyhedron $\Pi \subseteq f^*$ separates $\mathrm{proj}_f(P)$ and $H \cap f^*$ then its suspension $\mathrm{susp}_f(\Pi)$ separates P and H. Thus, P and H are r-separable if and only if $\mathrm{proj}_f(P)$ and $H \cap f^*$ are r-separable.

Since P and H are *not* r-separable, neither are $\mathrm{proj}_f(P)$ and $H \cap f^*$. The lower-dimensional hyperplanes $H \cap f^*$ do not have a common intersection. Thus, Case 1 implies that the set of configurations $P' \in (f^*)^m$ such that P' and $H \cap f^*$ are r-separable is closed. It follows that there is an open set $\mathcal{U}' \subseteq (f^*)^m$, with $\mathrm{proj}_f(P) \in \mathcal{U}'$, such that for all $Q' \in \mathcal{U}'$, Q' and $H \cap f^*$ are not r-separable.

Let $\mathcal{U}'' \subseteq (\mathbb{RP}^d)^m$ be the set of m-point configurations P'' such that $\mathrm{proj}_f(P'') \in \mathcal{U}'$. Clearly, $\mathcal{U}''$ is an open neighborhood of $P \setminus f$, and no configuration in $Q'' \in \mathcal{U}''$ is r-separable from H.

Finally, if Q'' and H are not r-separable, then no superset of Q'' is r-separable from H. Let $\mathcal{U} = \mathcal{U}'' \times (\mathbb{RP}^d)^{n-m}$. Then $\mathcal{U}$ is an open subset of $(\mathbb{RP}^d)^n$ containing P. Since every configuration $Q \in \mathcal{U}$ has a subset Q'' that is not r-separable from H, we conclude that no $Q \in \mathcal{U}$ is r-separable from H, as claimed.

This completes the proof of Case 2, and thus the entire technical lemma. $\square$

The method we used to encode the condition "$\mathrm{conv}(A)$ has at most r faces" may seem somewhat convoluted. If we replace $A \sqsubseteq B$ with "$\mathrm{conv}(A)$ is combinatorially equivalent to $\mathrm{conv}(B)$", we get exactly the same semi-algebraic set, without needing to define the partial order $\sqsubseteq$. Unfortunately, testing whether two convex polytopes are combinatorially equivalent requires *strict* inequalities, whose corresponding semi-algebraic sets are *open*.

7.3 Polyhedral Covers

A r-*polyhedral cover* of a set P of points and a set H of hyperplanes is an indexed set of subset pairs $\{(P_i, H_i)\}$, where $P_i \subseteq P$ and $H_i \subseteq H$ for all i, such that

(1) For each index i, P_i and H_i are r-separable.

(2) For every point $p \in P$ and hyperplane $h \in H$, there is some index i such that $p \in P_i$ and $h \in H_i$.

We emphasize that the subsets P_i are not necessarily disjoint, nor are the subsets H_i. We refer to each subset pair (P_i, H_i) in an r-polyhedral cover as a r-*polyhedral minor*. The *size* of a polyhedral cover is the sum of the sizes of the subsets P_i and H_i.

Let $\pi_r(P, H)$ denote the size of the smallest r-polyhedral cover of P and H. Let $\pi^\circ_{d,r}(n, m)$ denote the maximum of $\pi_r(P, H)$ over all sets P of n points and H of m hyperplanes in $\mathbb{RP}^d$ with no incidences. When the subscript r is omitted, we take it to be a constant. Finally, recall that $I(P, H)$ denotes the number of point-hyperplane incidences between P and H.

Lemma 7.2. *Let P be a set of n points and H a set of m hyperplanes, such that no subset of s hyperplanes contains t points in its intersection. If P and H are r-separable, then $I(P,H) \leq r(s+t)(n+m)$.*

Proof: Let Π be a polyhedron with r faces that separates P and H. For any point $p \in P$ and hyperplane $h \in H$ such that p lies on h, there is some face $f \leq \Pi$ that contains p and is contained in h. For each face f of Π, let P_f denote the points in P that are contained in f, and let H_f denote the hyperplanes in H that contain f.

Since no set of s hyperplanes can all contain the same t points, it follows that for all f, either $|P_f| < t$ or $|H_f| < s$. Thus, we can bound $I(P,H)$ as follows.

$$I(P,H) \leq \sum_{f \leq \Pi} I(P_f, H_f) = \sum_{f \leq \Pi} (|P_f| \cdot |H_f|) \leq (s+t) \sum_{f \leq \Pi} (|P_f| + |H_f|)$$

Since Π has r faces, the last sum counts each point and hyperplane at most r times. $\square$

The next lemma shows that sufficiently small perturbations of a configuration cannot decrease its polyhedral cover size.

Lemma 7.3. *Let P be a set of n points and H a set of m hyperplanes in $\mathbb{RP}^d$. For all point configurations $Q \in (\mathbb{RP}^d)^n$ sufficiently close to P, $\pi_r(Q,H) \geq \pi_r(P,H)$.*

Proof: Let P' be a subset of P, and for any other set Q of $|P|$ points, let Q' be the corresponding subset of Q. Let H' be a subset of H. Lemma 7.1 implies that there is an open set $\mathcal{U}(P',H') \subseteq (\mathbb{RP}^d)^n$ such that if $Q \in \mathcal{U}(P,H)$ and Q' and H' are r-separable, then P' and H' are r-separable.

Let $\mathcal{U}$ be the intersection of these $2^n 2^m$ open sets:

$$\mathcal{U} = \bigcap_{P' \subseteq P} \bigcap_{H' \subseteq H} \mathcal{U}(P',H').$$

For all $Q \in \mathcal{U}$, every r-polyhedral minor of Q and H corresponds to a r-polyhedral minor of P and H. Thus, for any r-polyhedral *cover* of Q and H, there is a corresponding r-polyhedral cover of P and H with exactly the same size. $\square$

Theorem 7.4. $\pi_2^\circ(m,n) = \Omega(n + n^{2/3}m^{2/3} + m)$.

Proof: Let P be a set of n points and H a set of m lines in the plane with $I(P,H) = \Omega(n + n^{2/3}m^{2/3} + m)$, as described by Lemma 6.4.

Consider subsets $P_i \subseteq P$ and $H_i \subseteq H$ such that P_i and H_i are r-separable. Since two distinct lines in the plane intersect in a single point, Lemma 7.2 implies that $I(P_i, H_i) \leq 4r(|P_i| + |H_i|)$. It follows that any collection of r-polyhedral minors that includes every incidence between P and H must have size at least $I(P,H)/4r$. Thus, $\pi_{2,r}(P,H) = \Omega(n + n^{2/3}m^{2/3} + m)$ for any constant r.

Finally, Lemma 7.3 implies that we can perturb P slightly, removing all the incidences, without decreasing the polyhedral cover size. $\square$

A similar argument derives the following lower bound from Lemma 6.13. As usual, the best lower bounds from each dimension have been combined into a single expression.

Theorem 7.5. $\pi^\circ_d(n,m) = \Omega\left(\sum_{i=1}^{d} \left(n^{1-2/i(i+1)}m^{2/(i+1)} + n^{2/(i+1)}m^{1-2/i(i+1)}\right)\right)$.

Following the terminology in the previous chapter, we call say that a point-hyperplane configuration in $\mathbb{R}^d$ is *monochromatic* if every point lies above every hyperplane. Monochromatic configurations have no incidences. Let $\hat{\pi}_{d,r}(m,n)$ denote the maximum of $\pi_r(P,H)$ over all monochromatic configurations of n points and of m hyperplanes in $\mathbb{R}^d \subset \mathbb{RP}^d$.

Lemma 6.15 and the arguments in Theorem 7.4 immediately imply the following lower bound.

Theorem 7.6. $\hat{\pi}_5(n,m) = \Omega(n + n^{2/3}m^{2/3} + m)$.

We can improve this bound very slightly in higher dimensions. Define the family of functions $\sigma_d : \mathbb{R}^{d+1} \to \mathbb{R}^{\binom{d+2}{2}}$ as follows.

$$\sigma_d(x_0, x_1, \ldots, x_d) \triangleq (x_0^2, x_1^2, \ldots, x_d^2, \sqrt{2}\,x_0x_1, \sqrt{2}\,x_0x_2, \ldots, \sqrt{2}\,x_{d-1}x_d)$$

For any two vectors $u, v \in \mathbb{R}^{d+1}$, we have $\langle \sigma_d(u), \sigma_d(v)\rangle = \langle u, v\rangle^2$, where $\langle\cdot,\cdot\rangle$ is the standard vector inner product. In a more geometric setting, σ_d maps points and hyperplanes in $\mathbb{R}^d$, represented in homogeneous coordinates, to points and hyperplanes in $\mathbb{R}^D$, also in homogeneous coordinates, where $D = \binom{d+2}{2} - 1 = d(d+3)/2$. If the point p is incident to the hyperplane h, then $\sigma_d(p)$ is also incident to $\sigma_d(h)$; otherwise, $\sigma_d(p)$ is above $\sigma_d(h)$.

Lemma 6.13 now immediately implies:

Theorem 7.7. *For all* $D \geq d(d+3)/2$,

$$\hat{\pi}_D(n,m) = \Omega\left(\sum_{i=1}^{d}\left(n^{1-2/i(i+1)}m^{2/(i+1)} + n^{2/(i+1)}m^{1-2/i(i+1)}\right)\right).$$

It is clear that $\hat{\pi}_3(n,m) = \Theta(n+m)$, since both the convex hull of any set of points and the upper envelope of any set of planes have linear size triangulations. We conjecture that $\hat{\pi}_d(n,n) = \Omega(n^{\lfloor d/2\rfloor/(\lfloor d/2\rfloor+1)})$ for all d, but are unable to prove this when $d = 4$ or $d \geq 6$.

7.4 Polyhedral Partitioning Algorithms

A *polyhedral partition graph* is a partition graph in which every query region is a projective polyhedron with at most r faces, for some fixed constant r. A typical value for r might be 2^{d+1} (every query region is a simplex) or 3^d+1 (every query region is a combinatorial cube). We still do not require the query regions to be disjoint. A *polyhedral partitioning algorithm* is a partitioning algorithm whose partition graph is polyhedral.

Given sets P of points and H of hyperplanes in $\mathbb{R}^d$ as input, a polyhedral partitioning algorithm for the halfspace emptiness problem constructs a polyhedral partition graph and uses it to drive the following divide-and-conquer process, which is slightly different from that used to solve Hopcroft's problem. As before, the algorithm starts at the root and proceeds through the graph in topological order, and at every node except the root, points and hyperplanes are passed in along incoming edges from preceding nodes. For each node v, let $P_v \subseteq P$ denote the points and $H_v \subseteq H$ the hyperplanes that reach v; at the root, we have $P_{root} = P$ and $H_{root} = H$. If v is a primal node, then for every query region $\Pi \in \mathcal{R}_v$, the points in P_v that are contained in Π and the hyperplanes in H_v *whose lower halfspaces* intersect Π traverse the corresponding outgoing edge. If v is a dual node, then for every $\Pi \in \mathcal{R}_v$, the points $p \in P_v$ whose dual hyperplanes p^* intersect *or lie above* Π and the hyperplanes $h \in H_v$ whose dual points h^* are contained in Π traverse the corresponding outgoing edge.[3]

[3]Alternately, we could let the points whose dual hyperplanes intersect Π and the hyperplane whose dual points intersect or lie below Π traverse the edge. Using this alternate formulation has no effect on our results. In fact, we can allow our partition graphs to have *four* types of non-leaf nodes — primal or dual; point/halfspace or ray/hyperplane — without changing our results, or even significantly altering their proofs.

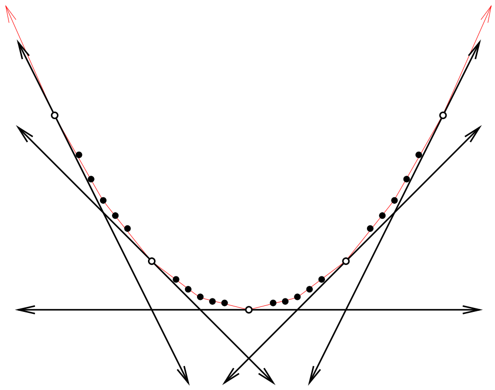


Figure 7.2. Worst-case configuration for halfspace emptiness. Tangent points are shown in white.

To solve the halfspace emptiness problem, a partitioning algorithm reports that all the points are above the hyperplanes if and only if no leaf in the partition graph is reached by both a point and a hyperplane. Clearly, if such an algorithm reports that every point is above every hyperplane, it must be correct.

Theorem 7.8. *Any polyhedral partitioning algorithm that solves the halfspace emptiness problem in* $\mathbb{R}^d$*, for any* $d \geq 2$*, must take time* $\Omega(n \log m + m \log n)$ *in the worst case.*

Proof: It suffices to consider the following configuration, where n is a multiple of m. P consists of n points on the unit parabola $x_d = x_1^2/2$ in $\mathbb{R}^d$, and H consists of m hyperplanes tangent to the parabola and orthogonal to the (x_1, x_d) plane, placed so that n/m points lie between adjacent points of tangency. All the points in P are above all the hyperplanes in H. The dual points H^* also lie on the parabola $x_d = x_1^2/2$, and the dual hyperplanes P^* are also tangent to that parabola.

Following the proof of Theorem 6.18, for any point, we call the hyperplane whose tangent point is closest in the positive x_1-direction the point's *partner*. Every hyperplane is the partner of n/m points. A node v *splits* a point-hyperplane pair if both the point and the hyperplane reach v, and none of the outgoing edges of v is traversed by both the point and the hyperplane. A hyperplane h is *active at level* k if no node in the first k levels splits h from any of its partners.

Suppose v is a primal node. For each hyperplane h that v splits from one of its partner points p, mark some query polyhedron $\Pi \in \mathcal{R}_v$ that contains p but misses h. Since Π has at most r faces, the intersection of Π and the parabola consists of at most r arcs, so

Π can be marked at most r times. Since there are at most Δ polyhedra in $\mathcal{R}_\nu$, at most $r\Delta$ hyperplanes become inactive at ν. Similarly, if ν is a dual node, then ν splits at most $r\Delta$ points from their partners.

Thus, the number of hyperplanes that are inactive at level k is less than $r\Delta^{k+2}$. In particular, at level $\lfloor \log_\Delta(m/r) \rfloor - 3$, at least $m(1 - 1/\Delta)$ hyperplanes are still active. It follows that at least $n(1 - 1/\Delta)$ points each traverse at least $\lfloor \log_\Delta(m/r) \rfloor - 3$ edges. We conclude that the total running time of the algorithm is at least

$$n(1 - 1/\Delta)(\lfloor \log_\Delta(m/r) \rfloor - 3) = \Omega(n \log m).$$

Symmetric arguments establish a lower bound of $\Omega(m \log n)$ when $n < m$. $\square$

The restriction to polyhedral partitioning algorithms is necessary for the lower bound to hold, since the problem can be solved in *linear* time in the generic partitioning algorithm model. The partition graph consists of a single primal node with two query regions: the convex hull of the points and its complement. If every point is above every hyperplane, then no hyperplane intersects the convex hull of the points.

This lower bound is tight, up to constant factors, in two and three dimensions.

Theorem 7.9. *Let $\mathcal{A}$ be a polyhedral partitioning algorithm that solves the halfspace emptiness problem, and let P be a set of points and H a set of hyperplanes, such that every point is above every hyperplane. Then $T_\mathcal{A}(P, H) = \Omega(\pi(P, H))$.*

Proof: From the proof of Theorem 6.19, we immediately have the following inequality:

$$\Delta \cdot T_\mathcal{A}(P, H) \geq \sum_{\text{primal edge } e} \Big(\#\text{ points traversing } e + \#\text{ hyperplanes missing } e\Big) + \sum_{\text{dual edge } e} \Big(\#\text{ hyperplanes traversing } e + \#\text{ points missing } e\Big)$$

For each primal edge e, let P_e be the set of points that traverse e, and let H_e be the set of hyperplanes that miss e. The edge e is associated with a query polyhedron Π. Every point in P_e is contained in Π, and every hyperplane in H_e is disjoint from Π. Since Π has at most r faces, P_e and H_e are r-separable.

Similarly, for each dual edge e, let H_e be the hyperplanes that traverse it, and P_e the points that miss it. The associated query polyhedron Π separates the dual points H_e^* and the dual hyperplanes P_e^*. By the definition of dual polyhedra, Π^* separates P_e and H_e.

For every point $p \in P$ and hyperplane $h \in H$, there is node that splits them (since otherwise the algorithm would return the wrong answer) and thus some edge e with $p \in P_e$ and $h \in H_e$. It follows that the collection of subset pairs $\{(P_e, H_e)\}$ is an r-polyhedral cover of P and H whose size is at least $\Delta \cdot T_{\mathcal{A}}(P, H)$ and, by definition, at most $\pi_r(P, H)$. □

We emphasize that every point must be above every hyperplane for this lower bound to hold. If some point lies below a hyperplane, then the trivial partitioning algorithm, whose partition graph consists of a single leaf, correctly "detects" the pair.

Corollary 7.10. *The worst-case running time of any polyhedral partitioning algorithm that solves the halfspace emptiness problem in* $\mathbb{R}^D$ *is* $\Omega(n \log m + n^{2/3}m^{2/3} + m \log n)$ *for all* $D \geq 5$ *and*

$$\Omega\left(n \log m + \sum_{i=2}^{D}\left(n^{1-2/i(i+1)}m^{2/(i+1)} + n^{2/(i+1)}m^{1-2/i(i+1)}\right) + m \log n\right)$$

for all $D \geq d(d+3)/2$.

Proof: Theorems 7.8 and 7.9 together imply that the worst case running time is $\Omega(n \log m + \hat{\pi}_d(n, m) + n \log m)$. The lower bounds then follow immediately from Theorem 7.6 and 7.7. □

Partitioning algorithms for the halfspace emptiness problem can (and do [47, 107]) apply a version of the "containment shortcut" described in Section 6.3.4. If some query region lies entirely in a hyperplane's lower halfspace, then the hyperplane need not traverse the corresponding edge. Instead, if any point lies in that region, we immediately halt and report that some point is below a hyperplane. Although this shortcut decreases the running time of the algorithm, we easily verify that Theorem 7.9 still applies in the faster model.

Our techniques also allow us to slightly improve earlier lower bounds for Hopcroft's problem in higher dimensions, matching our lower bounds for the counting problem in Corollary 6.22.

Theorem 7.11. *Let* $\mathcal{A}$ *be a polyhedral partitioning algorithm that solves Hopcroft's problem, and let* P *be a set of points and* H *a set of hyperplanes such that* $I(P, H) = 0$. *Then* $T_{\mathcal{A}}(P, H) = \Omega(\pi(P, H))$.

Combining this theorem with Theorem 7.5, we conclude:

Corollary 7.12. *The worst-case running time of any polyhedral partitioning algorithm that solves Hopcroft's problem in* $\mathbb{R}^d$ *is*

$$\Omega\left(n \log m + \sum_{i=2}^{d}\left(n^{1-2/i(i+1)} m^{2/(i+1)} + n^{2/(i+1)} m^{1-2/i(i+1)}\right) + m \log n\right).$$

7.5 Conclusions and Open Problems

We have proven a lower bound of $\Omega(n^{4/3})$ on the complexity of the offline halfspace emptiness problem in five dimensions. Our lower bounds apply to a broad class of geometric divide-and-conquer algorithms that recursively partition their input by a division of space into constant-complexity polyhedra.

The most obvious open problem is to improve our results. The correct complexity in d dimensions is almost certainly $\Theta(n^{2-2/\lfloor d/2 \rfloor})$, but we achieve this bound only when $d = 5$. In particular, the four dimensional case is wide open. It is not even known whether the four-dimensional halfspace emptiness problem is harder, or easier, than Hopcroft's problem in the plane [75].

The inner product doubling maps σ_d can be used to reduce Hopcroft's problem in $\mathbb{R}^d$ to halfspace emptiness in $\mathbb{R}^{d(d-3)/2}$ in linear time. Is there an efficient reduction from Hopcroft's problem to halfspace emptiness that only increases the dimension by a constant factor (preferably two)?

Our lower bounds are ultimately based on the construction of point-hyperplane configurations whose incidence graphs have several edges but no large complete bipartite subgraphs. Better such configurations would immediately lead to better lower bounds. Lower bounds in the Fredman/Yao semigroup arithmetic model have a similar basis. For example, Chazelle's lower bounds for offline simplex range searching [38] is based on a similar configuration of points and slabs. (See also [44].) Can we derive better polyhedral cover size bounds for points and hyperplanes from these configurations?

Another open problem is to prove tight lower bounds for *online* halfspace range query problems. Brönnimann, Chazelle, and Pach [23] have proven time-space tradeoffs for halfspace counting data structures in the Fredman/Yao semigroup model. Specifically, they prove that any data structure that uses space $n \le s \le n^d$ has worst-case query time

$$\Omega\left(\frac{(n/\log n)^{1-\frac{d-1}{d(d+1)}}}{s^{1/d}}\right).$$

Results of Matoušek [111] imply the upper bound $O((n/s^{1/d}) \operatorname{polylog} n)$, which is almost certainly optimal (except possibly for the polylog factor), so the lower bounds have significant room for improvement. Chazelle and Rosenberg [44] have developed quasi-optimal tradeoffs for simplex reporting data structures in Tarjan's pointer machine model, but no lower bounds are known for *halfspace* reporting. No lower bounds are known for online halfspace emptiness queries in any model of computation. One possible approach, suggested by Pankaj Agarwal (personal communication), is to model range query data structures with partition graphs and to prove tradeoffs between the total size of the graph (space) and the size of the subgraph induced by a query range (time).

A problem closely related to halfspace range searching is linear programming. The best known data structures of linear programming queries are based on data structures for halfspace emptiness [110] and halfspace reporting queries [27]. However, no nontrivial lower bounds are known for linear programming queries in any model of computation. One application of particular interest is deciding, given a set of points, whether every point is a vertex of the set's convex hull. Bounds for this problem closely match the best known bounds for halfspace emptiness [29], but the best known lower bound is $\Omega(n \log n)$. It seems unlikely that a lower bounds can be derived for this problem in the partitioning algorithm model, since the extremity of a point depends on several other points arbitrarily far away. Perhaps the techniques developed in Part I are more applicable.

Finally, extending our lower bounds into more traditional models of computation, such as algebraic decision trees or algebraic computation trees, is an important and extremely difficult open problem. A lower bound bigger than $\Omega(n \log m + m \log n)$ for *any* offline range searching problem in these models would be a major breakthrough.

> *Here, however, a word of warning may be in order: do* not *try to visualize n-dimensional objects for $n \geq 4$. Such an effort is not only doomed to failure—it may be dangerous to your mental health. (If you do succeed, then you are in trouble.) To speak of n-dimensional geometry with $n \geq 4$ simply means to speak of a certain part of algebra.*
>
> — Vašek Chvátal, *Linear Programming*, 1983

> *This is wrong, and even Chvátal acknowledges the fact that the correspondence between intuitive geometric terms and algebraic machinery can be used in both ways.*
>
> — Günter Ziegler, *Lectures on Polytopes*, 1995

Bibliography

[1] Pankaj K. Agarwal. Partitioning arrangements of lines: II. Applications. *Discrete Comput. Geom.*, 5:533–573, 1990.

[2] Pankaj K. Agarwal, N. Alon, B. Aronov, and S. Suri. Can visibility graphs be represented compactly? In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 338–347, 1993.

[3] Pankaj K. Agarwal, D. Eppstein, and J. Matoušek. Dynamic half-space reporting, geometric optimization, and minimum spanning trees. In *Proc. 33rd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 80–89, 1992.

[4] Pankaj K. Agarwal and J. Matoušek. Ray shooting and parametric search. *SIAM J. Comput.*, 22(4):794–806, 1993.

[5] A. Aggarwal, M. Hansen, and T. Leighton. Solving query-retrieval problems by compacting Voronoi diagrams. In *Proc. 22nd Annu. ACM Sympos. Theory Comput.*, pages 331–340, 1990.

[6] Nancy M. Amato and Edgar A. Ramos. On computing Voronoi diagrams by divide-prune-and-conquer. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 166–175, 1996.

[7] Nina Amenta and Günter Ziegler. Deformed products and maximal shadows of polytopes. Report 502-1996, Technische Univesität Berlin, May 1996. Available electronically at `ftp://ftp.math.tu-berlin.de/pub/Preprints/combi/Report-502-1996.ps.Z`.

[8] Nina Amenta and Günter Ziegler. Shadows and slices of polytopes. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 10–19, 1996.

[9] Arne Anderson, Torben Hagerup, Stefan Nilsson, and Rajeev Raman. Sorting in linear time? In *Proc. 27th Annu. ACM Sympos. Theory Comput.*, pages 427–436, 1995.

[10] D. Avis and K. Fukuda. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Discrete Comput. Geom.*, 8:295–313, 1992.

[11] David Avis and David Bremner. How good are convex hull algorithms? In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 20–28, 1995.

[12] David Avis, David Bremner, and Raimund Seidel. How good are convex hull algorithms? *Comput. Geom. Theory Appl.*, 1996. To appear. Full version of [11]. Available electronically at `ftp://mutt.cs.mcgill.ca/pub/doc/hgch.ps.gz`.

[13] Saugata Basu. On bounding the Betti numbers and computing the Euler characteristic of semi-algebraic sets. In *Proc. 28th Annu. ACM Sympos. Theory Comput.*, pages 408–417, 1996.

[14] Saugata Basu, Richard Pollack, and Marie-Fran coise Roy. On the number of cells defined by a family of polynomials on a variety. *Mathematika*. To appear. Available electronically at `http://www.math.nyu.edu/faculty/pollack/finalvariety.ps`.

[15] Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theor. Comput. Sci.*, 22:317–330, 1983.

[16] M. Ben-Or. Lower bounds for algebraic computation trees. In *Proc. 15th Annu. ACM Sympos. Theory Comput.*, pages 80–86, 1983.

[17] Samuel W. Bent and John W. John. Finding the median requires $2n$ comparisons. In *Proc. 17th ACM Sympos. Theory Comput.*, pages 213–216, 1985.

[18] A. Björner, M. Las Vergnas, N. White, B. Sturmfels, and G. Ziegler. *Oriented Matroids.* Cambridge University Press, Cambridge, 1993.

[19] Anders Björner, László Lovász, and Andrew C. C. Yao. Linear decision trees: Volume estimates and topological bounds. In *Proc. 24th Annu. ACM Sympos. Theory. Comput.*, pages 170–177, 1992.

[20] S. Bloch, J. Buss, and J. Goldsmith. How hard are n^2-hard problems? *SIGACT News*, 25(2):83–85, 1994.

[21] Manuel Blum, Robert W. Floyd, Vaughan Pratt, Ronald R. Rivest, and Robert E. Tarjan. Time bounds for selection. *J. Comput. System Sci.*, 7(4):448–461, 1973.

[22] J. Bochnak, M. Coste, and M.-F. Roy. *Géométrie algébraique réelle*, volume 12 of *Ergebnisse der Mathematik und ihrer Grenzgebeite 3*. Springer-Verlag, 1987.

[23] H. Brönnimann, B. Chazelle, and J. Pach. How hard is halfspace range searching? *Discrete Comput. Geom.*, 10:143–155, 1993.

[24] Stefan A. Burr, Branko Grünbaum, and N. J. A. Sloane. The orchard problem. *Geom. Dedicata*, 2:397–424, 1974.

[25] J. Canny. Some algebraic and geometric configurations in PSPACE. In *Proc. 20th Annu. ACM Sympos. Theory Comput.*, pages 460–467, 1988.

[26] J. Canny. Computing roadmaps in general semialgebraic sets. *Comput. J.*, 36:409–418, 1994.

[27] Timothy M. Chan. Fixed-dimensional linear programming queries made easy. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 284–290, 1996.

[28] Timothy M. Chan, Jack Snoeyink, and Chee-Keng Yap. Output-sensitive construction of polytopes in four dimensions and clipped Voronoi diagrams in three. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms (SODA '95)*, pages 282–291, 1995.

[29] Timothy M. Y. Chan. Output-sensitive results on convex hulls, extreme points, and related problems. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 10–19, 1995.

[30] D. R. Chand and S. S. Kapur. An algorithm for convex polytopes. *J. ACM*, 17:78–86, 1970.

[31] R. Chandrasekaran, Santosh N. Kabadi, and Katta G. Murty. Some NP-complete problems in linear programming. *Oper. Res. Lett.*, 1:101–104, 1982.

[32] B. Chazelle. Reporting and counting segment intersections. *J. Comput. Syst. Sci.*, 32:156–182, 1986.

[33] B. Chazelle. Lower bounds on the complexity of polytope range searching. *J. Amer. Math. Soc.*, 2:637–666, 1989.

[34] B. Chazelle. Lower bounds for orthogonal range searching, I: the reporting case. *J. ACM*, 37:200–212, 1990.

[35] B. Chazelle. Cutting hyperplanes for divide-and-conquer. *Discrete Comput. Geom.*, 9(2):145–158, 1993.

[36] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete Comput. Geom.*, 10:377–409, 1993.

[37] B. Chazelle. A spectral approach to lower bounds. In *Proc. 35th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 674–682, 1994.

[38] B. Chazelle. Lower bounds for off-line range searching. In *Proc. 27th Annu. ACM Sympos. Theory Comput.*, pages 733–740, 1995.

[39] B. Chazelle, L. J. Guibas, and D. T. Lee. The power of geometric duality. *BIT*, 25:76–90, 1985.

[40] B. Chazelle and J. Matoušek. Derandomizing an output-sensitive convex hull algorithm in three dimensions. Technical report, Dept. Comput. Sci., Princeton Univ., 1992.

[41] B. Chazelle and B. Rosenberg. Computing partial sums in multidimensional arrays. In *Proc. 5th Annu. ACM Sympos. Comput. Geom.*, pages 131–139, 1989.

[42] B. Chazelle and B. Rosenberg. The complexity of computing partial sums off-line. *Internat. J. Comput. Geom. Appl.*, 1(1):33–45, 1991.

[43] B. Chazelle, M. Sharir, and E. Welzl. Quasi-optimal upper bounds for simplex range searching and new zone theorems. *Algorithmica*, 8:407–429, 1992.

[44] Bernard Chazelle and Burton Rosenberg. Simplex range reporting on a pointer machine. *Comput. Geom. Theory Appl.*, 5:237–247, 1996.

[45] F. R. K. Chung, P. Erdős, and J. Spencer. On the decomposition of graphs into complete bipartite subgraphs. In Paul Erdős, editor, *Studies in pure mathematics*, pages 95–101. Birkhäuser, 1983.

[46] K. Clarkson, H. Edelsbrunner, L. Guibas, M. Sharir, and E. Welzl. Combinatorial complexity bounds for arrangements of curves and spheres. *Discrete Comput. Geom.*, 5:99–160, 1990.

[47] K. L. Clarkson. New applications of random sampling in computational geometry. *Discrete Comput. Geom.*, 2:195–222, 1987.

[48] K. L. Clarkson. A Las Vegas algorithm for linear programming when the dimension is small. In *Proc. 29th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 452–456, 1988.

[49] K. L. Clarkson. More output-sensitive geometric algorithms. In *Proc. 35th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 695–702, 1994.

[50] K. L. Clarkson and P. W. Shor. Applications of random sampling in computational geometry, II. *Discrete Comput. Geom.*, 4:387–421, 1989.

[51] R. Cole, M. Sharir, and C. K. Yap. On k-hulls and related problems. *SIAM J. Comput.*, 16:61–77, 1987.

[52] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* The MIT Press, Cambridge, Mass., 1990.

[53] M. de Berg, M. Overmars, and O. Schwarzkopf. Computing and verifying depth orders. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 138–145, 1992.

[54] M. de Berg and O. Schwarzkopf. Cuttings and applications. Report RUU-CS-92-26, Dept. Comput. Sci., Utrecht Univ., Utrecht, Netherlands, August 1992.

[55] M. Dietzfelbinger. Lower bounds for sorting of sums. *Theoret. Comput. Sci.*, 66:137–155, 1989.

[56] Martin Dietzfelbinger and Wolfgang Maass. Lower bound arguments with "inaccessible" numbers. *J. Comput. Syst. Sci.*, 36:313–335, 1988.

[57] D. P. Dobkin and D. G. Kirkpatrick. Determining the separation of preprocessed polyhedra – a unified approach. In *Proc. 17th Internat. Colloq. Automata Lang. Program.*, volume 443 of *Lecture Notes in Computer Science*, pages 400–413. Springer-Verlag, 1990.

[58] D. P. Dobkin and R. J. Lipton. On the complexity of computations under varying sets of primitives. *J. Comput. Syst. Sci.*, 18:86–91, 1979.

[59] David Dobkin and Richard J. Lipton. A lower bound of $\frac{1}{2}n^2$ on linear search programs for the knapsack problem. *J. Comput. Sys. Sci.*, 16(3):413–417, 1978.

[60] Charles Lutwidge Dodgson. *St. James Gazette*, August 1, 1883, pages 5–6.

[61] M. E. Dyer. The complexity of vertex enumeration methods. *Math. Oper. Res.*, 8:381–402, 1983.

[62] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, Heidelberg, West Germany, 1987.

[63] H. Edelsbrunner, L. Guibas, J. Hershberger, R. Seidel, M. Sharir, J. Snoeyink, and E. Welzl. Implicitly representing arrangements of lines or segments. *Discrete Comput. Geom.*, 4:433–466, 1989.

[64] H. Edelsbrunner, L. Guibas, and M. Sharir. The complexity of many cells in arrangements of planes and related problems. *Discrete Comput. Geom.*, 5:197–216, 1990.

[65] H. Edelsbrunner and L. J. Guibas. Topologically sweeping an arrangement. *J. Comput. Syst. Sci.*, 38:165–194, 1989. Corrigendum in 42 (1991), 249–251.

[66] H. Edelsbrunner, L. J. Guibas, and M. Sharir. The complexity and construction of many faces in arrangements of lines and of segments. *Discrete Comput. Geom.*, 5:161–196, 1990.

[67] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graph.*, 9:66–104, 1990.

[68] H. Edelsbrunner, J. O'Rourke, and R. Seidel. Constructing arrangements of lines and hyperplanes with applications. *SIAM J. Comput.*, 15:341–363, 1986.

[69] H. Edelsbrunner, R. Seidel, and M. Sharir. On the zone theorem for hyperplane arrangements. *SIAM J. Comput.*, 22(2):418–429, 1993.

[70] H. Edelsbrunner and M. Sharir. A hyperplane incidence problem with applications to counting distances. In P. Gritzman and B. Sturmfels, editors, *Applied Geometry and Discrete Mathematics: The Victor Klee Festschrift*, volume 4 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 253–263. AMS Press, 1991.

[71] I. Emiris and J. Canny. A general approach to removing degeneracies. In *Proc. 32nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 405–413, 1991.

[72] P. Erdős and G. Purdy. Two combinatorial problems in the plane. *Discrete Comput. Geom.*, 13(3–4):441–443, 1995.

[73] J. Erickson and R. Seidel. Better lower bounds on detecting affine and spherical degeneracies. *Discrete Comput. Geom.*, 13:41–57, 1995.

[74] Jeff Erickson. Lower bounds for linear satisfiability problems. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms (SODA '95)*, pages 388–395, 1995.

[75] Jeff Erickson. On the relative complexities of some geometric problems. In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 85–90, 1995.

[76] Jeff Erickson. New lower bounds for convex hull problems in odd dimensions. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 1–9, 1996.

[77] Jeff Erickson. New lower bounds for halfspace emptiness. In *Proc. 37th Annu. IEEE Sympos. Found. Comput. Sci.*, page to appear, 1996.

[78] Jeff Erickson. New lower bounds for Hopcroft's problem. *Disc. Comput. Geom.*, 1996. Special issue of papers from the 11th ACM Sympos. Comput. Geom., to appear.

[79] Stefan Felsner. On the number of arrangements of pseudolines. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 30–37, 1996.

[80] M. L. Fredman. How good is the information theory bound in sorting? *Theoret. Comput. Sci.*, 1:355–361, 1976.

[81] M. L. Fredman. A lower bound on the complexity of orthogonal range queries. *J. ACM*, 28:696–705, 1981.

[82] M. L. Fredman. Lower bounds on the complexity of some optimal data structures. *SIAM J. Comput.*, 10:1–10, 1981.

[83] Michael Fredman and Dan E. Willard. Surpassing the information-theoretic bound with fusion trees. *J. Comput. Syst. Sci.*, 47(3):424–436, 1993. The extended abstract (STOC 1990) had a much more colorful title.

[84] Z. Füredi and I. Palásti. Arrangements of lines with a large number of triangles. *Proc. Amer. Math. Soc.*, 92(4):561–566, 1984.

[85] A. Gajentaan and M. H. Overmars. n^2-hard problems in computational geometry. Report RUU-CS-93-15, Dept. Comput. Sci., Utrecht Univ., Utrecht, Netherlands, April 1993.

[86] A. Gajentaan and M. H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom. Theory Appl.*, 5:165–185, 1995.

[87] David Gale. Neighborly and cyclic polytopes. In V. Klee, editor, *Convexity*, volume VII of *Proc. Symposia in Pure Mathematics*, pages 225–232. Amer. Math. Soc., 1963.

[88] J. E. Goodman and R. Pollack. Multidimensional sorting. *SIAM J. Comput.*, 12:484–507, 1983.

[89] J. E. Goodman and R. Pollack. Allowable sequences and order types in discrete and computational geometry. In J. Pach, editor, *New Trends in Discrete and Computational Geometry*, volume 10 of *Algorithms and Combinatorics*, pages 103–134. Springer-Verlag, 1993.

[90] J. E. Goodman, R. Pollack, and B. Sturmfels. The intrinsic spread of a configuration in $\mathbf{R}^d$. *J. Amer. Math. Soc.*, 3:639–651, 1990.

[91] R. L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Inform. Process. Lett.*, 1:132–133, 1972.

[92] Ronald R. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Addison-Wesley, Reading, Massachusetts, 1989.

[93] Dima Grigoriev, Marek Karpinksi, Friedhelm Meyer auf der Heide, and Roman Smolensky. A lower bound for randomized algebraic decision trees. In *Proc. 28th Annu. ACM Sympos. Theory Comput.*, pages 612–619, 1996.

[94] Dima Grigoriev, Marek Karpinski, and Nicolai Vorobjov. Improved lower bound on testing membership to a polyhedron by algebraic decision trees. In *Proc. 36th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 258–265, 1995.

[95] B. Grünbaum. *Convex Polytopes.* Wiley, New York, NY, 1967. Revised edition, V. Klee and P. Kleinschmidt, editors, *Graduate Texts in Mathematics*, Springer-Verlag, in preparation.

[96] Branko Grünbaum. *Arrangements and Spreads.* Number 10 in Regional Conf. Ser. Math. Amer. Math. Soc., Providence, RI, 1972.

[97] H. Günzel. The universal partition theorem for oriented matroids. *Discrete Comput. Geom.* To appear.

[98] G. Hardy and E. Wright. *The Theory of Numbers.* Oxford University Press, London, England, 4th edition, 1965.

[99] Laurent Hyafil. Bounds for selection. *SIAM J. Comput.*, 5(1):109–114, 1976.

[100] Carl Gustav Jakob Jacobi. De functionibus alternantibus earumque divisione per productum e differentiis elementorum conflatum. *J. Reine Angew. Mathematik*, 22:360–371, 1841. Reprinted in *Gesammelten Werke* III, G. Reimer , Berlin, 1884.

[101] M. J. Katz and M. Sharir. An expander-based approach to geometric optimization. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 198–207, 1993.

[102] S. S. Kislicyn. On the selection of the kth element of an ordered set by pairwise comparisons. *Sibirskiĭ Matematičeskiĭ Žurnal*, 5:557–564, 1964. In Russian.

[103] D. E. Knuth. *Fundamental Algorithms*, volume 1 of *The Art of Computer Programming.* Addison-Wesley, Reading, MA, 2nd edition, 1973.

[104] D. E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming.* Addison-Wesley, Reading, MA, 1973.

[105] Donald E. Knuth. *Axioms and Hulls*, volume 606 of *Lecture Notes in Computer Science*. Springer-Verlag, Heidelberg, Germany, 1992.

[106] László Lovàsz. Communication complexity: A survey. In Bernhard Korte, László Lovász, Hans Jürgen Prömel, and Alexander Schrijver, editors, *Paths, Flows, and VLSI Layout*, volume 9 of *Algorithms and Combinatorics*, pages 235–265. Springer-Verlag, 1990.

[107] J. Matoušek. Reporting points in halfspaces. *Comput. Geom. Theory Appl.*, 2(3):169–186, 1992.

[108] J. Matoušek and O. Schwarzkopf. On ray shooting in convex polytopes. *Discrete Comput. Geom.*, 10(2):215–232, 1993.

[109] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 1–8, 1992.

[110] J. Matoušek. Linear optimization queries. *J. Algorithms*, 14:432–448, 1993.

[111] J. Matoušek. Range searching with efficient hierarchical cuttings. *Discrete Comput. Geom.*, 10(2):157–182, 1993.

[112] J. Matoušek. Geometric range searching. *ACM Comput. Surv.*, 26:421–461, 1994.

[113] N. Megiddo. Linear programming in linear time when the dimension is fixed. *J. ACM*, 31:114–127, 1984.

[114] F. Meyer auf der Heide. A polynomial time linear search algorithm for the n-dimensional knapsack problem. *J. ACM*, 31:668–676, 1984.

[115] J. Milnor. On the betti numbers of real algebraic varieties. *Proc. Amer. Math. Soc.*, 15:275–280, 1964.

[116] Nicoali E. Mnëv. The universailty theorems on the classification problem of configuration varieties and convex polytopes varieties. In O. Y. Viro, editor, *Topology and Geometry—Rohlin Seminar*, volume 1346 of *Lecture Notes in Mathematics*, pages 527–544. Springer-Verlag, 1988.

[117] Jaroslav Morávek. On the complexity of discrete programming problems. *Apl. Mat.*, 14:442–474, 1969.

[118] Jaroslav Morávek. A localization problem in geometry and complexity of discrete programming. *Kybernetika*, 8:498–516, 1972.

[119] János Pach. Personal communication.

[120] I. G. Petrovskiĭ and O. A. Oleĭnik. On the topology of real algebraic surfaces. *Isvestia Akad. Nauk SSSR. Ser. Mat.*, 13:389–402, 1949. In Russian.

[121] I. G. Petrovskiĭ and O. A. Oleĭnik. *On the topology of real algebraic surfaces*, volume 70 of *Amer. Math. Soc. Translation*. 1952. 20 pages.

[122] Vaughan R. Pratt and Foong Frances Yao. On lower bounds for computing the i-th largest element. In *Proc. 14th Annu. IEEE Sympos. Switching and Automata Theory*, pages 70–81, 1973.

[123] F. P. Preparata and S. J. Hong. Convex hulls of finite sets of points in two and three dimensions. *Commun. ACM*, 20:87–93, 1977.

[124] A. Prestel. *Lectures on Formally Real Fields*, volume 1093 of *Lecture Notes in Mathematics*. Springer-Verlag, 1984.

[125] M. O. Rabin. Proving simultaneous positivity of linear forms. *J. Comput. Syst. Sci.*, 6:639–650, 1972.

[126] E. M. Reingold. On the optimality of some set algorithms. *J. ACM*, 19:649–659, 1972.

[127] J. Richter-Gebert and G. M. Ziegler. Realization spaces of 4-polytopes are universal. *Bull. Amer. Math. Soc.*, 32(4), October 1995.

[128] Jürgen Richter-Gebert. Mnëv's universailty theorem revisited. Séminaire Lotaringien de Combinatorie, 1995. 15 pages. Available electronically at `http://winnie.math.tu-berlin.de/~richter/partition.ps.Z`.

[129] Jürgen Richter-Gebert. *Realization Spaces of 4-Polytopes are Universal*. Habilitationsschrift, Techniche Universität Berlin, May 1995. Lecture Notes in Mathematics, Springer-Verlag, in preparation. Available electronically at `http://winnie.math.tu-berlin.de/~richter/universality.ps.Z`.

[130] G. Rote. Degenerate convex hulls in high dimensions without extra storage. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 26–32, 1992.

[131] Issai Schur. *Über eine Klasse von Matrizen, die sich einer gegebenen Matrix zuordnen lassen.* Thesis, Berlin, 1901. Reprinted in *Gesammelte Abhandlungen*, Springer, 1973.

[132] R. Seidel. A convex hull algorithm optimal for point sets in even dimensions. M.Sc. thesis, Dept. Comput. Sci., Univ. British Columbia, Vancouver, BC, 1981. Report 81/14.

[133] R. Seidel. A method for proving lower bounds for certain geometric problems. In G. T. Toussaint, editor, *Computational Geometry*, pages 319–334. North-Holland, Amsterdam, Netherlands, 1985.

[134] R. Seidel. Constructing higher-dimensional convex hulls at logarithmic cost per face. In *Proc. 18th Annu. ACM Sympos. Theory Comput.*, pages 404–413, 1986.

[135] R. Seidel. A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons. *Comput. Geom. Theory Appl.*, 1:51–64, 1991.

[136] R. Seidel. Small-dimensional linear programming and convex hulls made easy. *Discrete Comput. Geom.*, 6:423–434, 1991.

[137] P. W. Shor. Stretchability of pseudolines is NP-hard. In *Applied Geometry and Discrete Mathematics: The Victor Klee Festschrift*, volume 4 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 531–554. AMS Press, 1991.

[138] J. M. Steele and A. C. Yao. Lower bounds for algebraic decision trees. *J. Algorithms*, 3:1–8, 1982.

[139] William Steiger and Ileana Streinu. A pseudo-algorithmic separation of lines from pseudo-lines. *Inform. Process. Lett.*, 53(5):295–299, 1995.

[140] J. Stolfi. *Oriented Projective Geometry: A Framework for Geometric Computations.* Academic Press, New York, NY, 1991.

[141] Volker Strassen. Die Berechnungskomplexität von elementarsymmetrichen Funktionen und von Interpolationskoeffizienten. *Numer. Math.*, 20:238–251, 1973.

[142] G. F. Swart. Finding the convex hull facet by facet. *J. Algorithms*, 6:17–48, 1985.

[143] E. Szemerédi and W. T. Trotter, Jr. Extremal problems in discrete geometry. *Combinatorica*, 3:381–392, 1983.

[144] R. E. Tarjan. A class of algorithms which require nonlinear time to maintain disjoint sets. *J. Comput. Syst. Sci.*, 18:110–127, 1979.

[145] T. G. Tarján. Complexity of lattice-configurations. *Studia Sci. Math. Hungar.*, 10:203–211, 1975.

[146] A. Tarski. *A Decision Method for Elementary Algebra and Geometry.* University of California Press, Berkeley, CA, 1951. Prepared for publication by J. C. C. McKinsey.

[147] R. Thom. Sur l'homologie des variétés algébriques reélles. In S. S. Cairns, editor, *Differential and Combinatorial Topology*, pages 225–265. Princeton Univ. Press, 1965.

[148] Hing F. Ting and Andrew C. Yao. A randomized algorithm for finding maximum with $O((\log n)^2)$ polynomial tests. *Inform. Process. Lett.*, 49(1):39–43, 1994.

[149] Zsolt Tuza. Covering of graphs by complete bipartite subgraphs; complexity of 0-1 matrices. *Combinatorica*, 4:111–116, 1984.

[150] P. M. Vaidya. Space-time tradeoffs for orthogonal range queries. *SIAM J. Comput.*, 18:748–758, 1989.

[151] J. van Leeuwen. Problem P20. *Bull. EATCS*, 19:150, 1983.

[152] H. E. Warren. Lower bounds for the approximation of nonlinear manifolds. *Trans. Amer. Math. Soc.*, 133:167–178, 1968.

[153] Dan E. Willard. Lower bounds for the addition-subtraction operations in orthogonal range queries and related problems. *Inform. Comput.*, 82(1):45–64, 1989.

[154] A. C. Yao. A lower bound to finding convex hulls. *J. ACM*, 28:780–787, 1981.

[155] A. C. Yao. On the complexity of maintaining partial sums. *SIAM J. Comput.*, 14:277–288, 1985.

[156] A. C. Yao. Lower bounds for algebraic computation trees with integer inputs. *SIAM J. Comput.*, 20:655–668, 1991.

[157] A. C. Yao and R. L. Rivest. On the polyhedral decision problem. *SIAM J. Comput.*, 9:343–347, 1980.

[158] Andrew Chi-Chih Yao. Decision tree complexity and Betti numbers. In *Proc. 26th Annu. ACM Sympos. Theory Comput.*, pages 615–624, 1994.

[159] Andrew Chi-Chih Yao. Algebraic decision trees and Euler characteristics. *Theor. Comput. Sci.*, 141(1–2):133–150, 1995.

[160] C. K. Yap. A geometric consistency theorem for a symbolic perturbation scheme. *J. Comput. Syst. Sci.*, 40:2–18, 1990.

[161] G. M. Ziegler. *Lectures on Polytopes*, volume 152 of *Graduate Texts in Mathematics*. Springer-Verlag, 1994.

Electronic addresses are given for preliminary unpublished versions of some references. These addresses are current as of July 1996, but may change at any time. Once these works are formally published, electronic copies may no longer be available.

Index of Notation

$\mathcal{A}$ — an algorithm ... 56, 83

$A \sqsubseteq B$ — A is simpler than B ... 97

$\mathcal{C}$ — a cell in an arrangement of hyperplanes ... 57, 96

Δ — the maximum outdegree of a partition graph ... 81

ε — a generic infinitesimal ... 49
a sufficiently small positive real number ... 25, 72
an arbitrarily positive constant (in an asymptotic time bound) ... 62

f^* — the dual flat of f ... 94

$I(P, H)$ — the number of incidences between points P and hyperplanes H ... 69

$I_d(n, m)$ — maximum number of incidences between n points and m hyperplanes in $\mathbb{R}^d$ ... 69

$i \perp j$ — i and j are relatively prime ... 72

$I(P, H)$ — the number of incidences between points P and hyperplanes H ... 100

$K(\varepsilon)$ — an extension of the ordered field K ... 49

$\widetilde{K}$ — the real closure of the ordered field K ... 49

$\log^* n$ — iterated logarithm of n ... 11

$M(P, H)$ — the relative orientation matrix of points P and hyperplanes H ... 68

$\mu(P, H)$ — minimum monochromatic cover size of points P and hyperplanes H ... 69

$\mu_d(n, m)$ — worst case monochromatic cover size for n points and m hyperplanes in $\mathbb{R}^d$ ... 69

$\mu_d^\circ(n, m)$ — worst case monochromatic cover size for n points and m hyperplanes in $\mathbb{R}^d$ with no incidences ... 69

$[n]$ — the integers $\{1, 2, \ldots, n\}$ ... 72

$n^{\underline{a}}$ — falling factorial power: $n!/(n-a)!$ ... 58

$\Omega(\cdot)$ — asymptotic lower bound ... 2

$\omega_d(t)$	the d-dimensional weird moment curve: $(t, t^2, \ldots, t^{d-1}, t^{d+1})$	20
ϕ	a fixed linear expression in r variables	50
$\varphi(n)$	the Euler totient function	72
$\Phi \leq \Pi$	Φ is a face of Π	94
$\pi_r(P, H)$	minimum r-polyhedral cover size of points P and hyperplanes H	100
$\hat{\pi}_{d,r}(n, m)$	worst case r-polyhedral cover size for monochromatic configurations of n points and m hyperplanes in $\mathbb{R}^d$	101
$\pi^\circ_{d,r}(n, m)$	worst case r-polyhedral cover size of n points and m hyperplanes in $\mathbb{RP}^d$ with no incidences	100
Π^*	the dual of the projective polyhedron Π	95
$\mathrm{proj}_f(X)$	the projection of a set X by a flat f	95
$\mathcal{Q}_{\mathcal{A}}$	the set of query polynomials used by algorithm $\mathcal{A}$	56
$\mathbb{R}(\varepsilon_1, \varepsilon_2, \varepsilon_3)$	a tower of field extensions of the reals	49
$\mathbb{RP}^d$	d-dimensional real projective space	94
$\mathcal{R}_v$	the set of query regions associated with a node v in a partition graph	81
σ	the inner product doubling map from $\mathbb{R}^3$ to $\mathbb{R}^6$	80
σ_d	the inner product doubling map from $\mathbb{R}^{d+1}$ to $\mathbb{R}^{\binom{d+2}{2}}$	101
$\mathrm{span}(X)$	the projective span of X	94
$\mathrm{susp}_f(X)$	the suspension of a set X by a flat f	95
$T_{\mathcal{A}}(P, H)$	the running time of algorithm $\mathcal{A}$ given P and H as input	83
$\mathcal{U}$	an open set in $(\mathbb{RP}^d)^n$	98
$\#W$	number of connected components of a set W	5
$\zeta(P, H)$	minimum zero cover size of points P and hyperplanes H	69
$\zeta_d(n, m)$	worst case zero cover size for n points and m hyperplanes in $\mathbb{R}^d$	69

Index

3SUM-hard, 27–28, 59–60

Abbott, Edwin, 46
adversary, 12
adversary argument, 4–5, 16–18, 21, 35, 51, 58, 65–67
Agarwal, Pankaj, 38, 62, 107
algebraic computation tree, *see* models of computation
algebraic decision tree, *see* models of computation
Amato, Nancy, 32, 38
Amenta, Nina, 39
Anderson, Laurie, iii
apex, 94, 96
Avis, David, 32, 38

Ben-Or, Michael, 6, 31, 32, 35, 48
de Berg, Mark, 91
Bern, Marshall, viii
Bremner, David, 32
Brönnimann, Hervé, 7, 106
bugs in original proofs, 13, 30, 45, 60

Chan, Timothy, 31, 38, 39
Chand, Donald, 31, 37
Chazelle, Bernard, 7–9, 11, 13, 31, 37, 62, 64, 71, 80, 88, 89, 106
Chung, Fan, 68
Chvátal, Vašek, 107
Clarkson, Ken, 31, 37, 38, 71, 74, 92
close pair, 72, 76
Cole, Richard, 62
collapsible simplex, 18, 21, 25, 27, 35, 37–39
collapsible triangle, 16, 23, 28
collapsible tuple, 41, 42, 44, 50, 52–55
comparison tree, *see* models of computation
complete bipartite subgraph, 8, 68, 106
configuration, 50, 57, 60
configuration space, 24, 50, 57
conjecture, 45, 102
convex hull, 32

decision tree, *see* models of computation
degenerate facet, 33
degenerate simplex, 14, 33
Dietzfelbinger, Martin, 48
Dieudonné, Jean, 12
Dobkin, David, viii, 5, 48
Dryden, John, 61
Drysdale, Scot, viii
duality, 13, 16, 22, 25, 26, 38, 63, 67, 75, 81, 87, 88, 92–95, 103, 104

Edelsbrunner, Herbert, 13, 62, 71, 74
edge
  of a partition graph, 81
  of a polytope, 33
Einstein, Albert, 39
elementary formula, 49, 56, 97–98
Eppstein, David, viii
Erdős, Pál, 64, 68, 71
exercises for the reader, 25, 67, 71

face, 94
  dimension of, 94
face lattice, 94, 95
facet, 33
flat, 94
  dual, 94, 98
Fredman, Michael, 7, 47, 48, 58, 71
Fredman/Yao semigroup arithmetic model, *see* models of computation

Fukuda, Komei, 38
Füredi, Zoltán, 22

Gaiman, Neil, vii
Gajentaan, Anka, 27
Gale, David, 33
Goodrich, Mike, viii
Grünbaum, Branko, 23, 32
Graham, Ron, 31
green, 57
Guibas, Leo, 13, 71, 74

Heisel, Carl Theodore, 30
Hershberger, John, 62
homogeneous coordinates, 24, 42, 65, 80, 97, 101
Hong, Se June, 31
Hopcroft, John, 62
hyperplane arrangement, 14, 57, 58, 94, 96

incidence graph, 8, 68, 75, 106
index, 124–127
infinitesimal, 49
information theory, 3, 6

Jacobi, Carl Gustav Jakob, 21

Kapur, Sham, 31, 37

Lee, Der-Tsai, 13
van Leeuwen, Jan, 13
Lem, Stanislaw, 60
linear decision tree, *see* models of computation
Lipton, Richard, 5, 48
lonely, 75
loosely monochromatic, 76

Maass, Wolfgang, 48
Matoušek, Jiří, 8, 38, 62, 80, 88, 92, 106
Mehlhorn, Kurt, vii
Meyer auf der Heide, Friedhelm, 48
Milnor, John, 6
Mnëv, Nikolai, 29
models of computation, 2
  algebraic computation tree, 6, 14, 31
  algebraic computation trees, 107
  algebraic decision tree, 5–7, 14, 24, 31, 91
    limitations, 6
  algebraic decision trees, 107
  comparison tree, 4
  decision tree, 3–7, 14
  direct query decision tree, 47
  group arithmetic, 8
  integer RAM, 28
  linear decision tree, 5, 47, 50
  partitioning algorithms, 63, 81–89
    containment shortcut, 86–87, 105
    nondeterminism of, 82
    polyhedral, 102–106
    trivial, 85, 104, 105
    unbounded, 86
    vs. real algorithms, 88–89
  pointer machine, 9
  r-linear decision tree, 47, 50–59
  semigroup arithmetic, 7–8, 64, 80, 106
    limitations, 8
moment curve, 20
monochromatic, 68, 101
monochromatic cover, 68–80, 84, 85
  size, 68

n^2-hard, *see* 3SUM-hard
nondegenerate with respect to a query, 24, 37, 42
nonuniform algorithm, 29, 58–59
notation, 3, 11, 20, 62, 69, 72, 91, 94–96, 99–100, 122–123
NP-hard, 6, 29, 32, 39, 60

O'Rourke, Joseph, vii, 13
obvious, 4, 5, 8, 21, 28, 59, 65, 106
Oleĭnik, Olga Arsenievna, 6
open problems, 5, 28, 38–39, 45–46, 59–60, 67, 79, 89–91, 106–107
order type, 14
ordered field, 49
orientation of a simplex, 14
orientation test, *see* primitives, sidedness query

output size, 2, 9, 31–32
Overmars, Mark, 27, 91

Pach, János, 7, 106
Palásti, Ilona, 22
partition graph, 81
  as a range searching data structure, 107
  level of node in, 82
  polyhedral, 102
  primal and dual edges, 84
  primal and dual nodes, 81, 88, 102
partitioning algorithms, *see* models of computation
partner, 82, 103
Petrovskiĭ, Ivan Goergievič, 6
Plato, 61
polyhedral cover, 99–102, 105
  size, 99
polyhedral minor, 99
polyhedral partitioning algorithm, *see* models of computation
polyhedron, *see* projective polyhedron
polytope, 32
  face of, 32
  projective, 94
Preparata, Franco, 31
primitives
  algebraic query, 24, 66
  allowable query, 23–26, 37, 59
    examples, 25–26
  circularly allowable query, 42
  direct query, 47
  incircle query, 40
  insphere query, 26, 40
  line query, 65
  point query, 65
  relative orientation query, 64
  sidedness query, 13, 21, 23
  tuple comparison, 19
  used by real convex hull algorithms, 37–38
problems
  20 questions, 3
  3SUM, 27, 53
  affine degeneracy, 13–30, 47
    arbitrary, 20–26
    input in convex position, 21
    nonvertical, 15–18
  circular degeneracy
    arbitrary, 41–42
    proper, 41
  convex hull, 31–39
  cyclic overlap, 90
  element uniqueness, 5, 28, 47
  extreme points, 39, 107
  halfspace emptiness, 67
  Hopcroft's problem, 47, 62–67, 82–86, 105–106
    counting version, 80, 85–86
  linear programming, 107
  linear satisfiability, 19, 47–60
  MAX, 4
  median selection, 4
  minimum measure simplex, 26
  range emptiness, 8
    offline halfspace, 92–93, 102–105
  range reporting, 9
    orthogonal, 9
    simplex, 9
  range searching, 7–8
    dynamic halfplane, 7
    dynamic orthogonal, 7, 8
    halfspace, 7, 106
    offline halfplane, 8
    offline orthogonal, 8
    offline simplex, 8, 64, 106
    orthogonal, 7
    simplex, 7, 64
  SEPARATOR2, 28
  set intersection, 63
  set membership, 5–7
  sorting, 3
  sorting $X + Y$, 48, 59
  spherical degeneracy, 47
    arbitrary, 45
    proper, 42–45
projection, 95, 98
projective polyhedron, 94
  dual, 95

projective polytope, *see* polytope, projective
projective transformation, 24, 37, 92, 94
proper face, 94
proper spherical degeneracy, 40
pseudoline arrangement, 29

quasi-simplicial polytope, 33
query, 3
query polynomial, 5, 24
query region, 81, 102

r-linear decision tree, *see* models of computation
r-separable, 96
Ramos, Edgar, 32, 38
randomization, viii
real closed field, 49
real closure, 49
red, 57
relative orientation matrix, 68
relatively collapsible tuple, 56–57
ridge, 33, 38
Rosenberg, Burton, 8, 9, 106
Rucker, Rudy, 91

Schur, Issai, 21
Schwarzkopf, Otfried, vii, 38, 91
Seidel, Miriam, vii
Seidel, Raimund, vii, 13, 25, 31, 32, 37, 38, 42, 62, 89
semialgebraic, 6, 50
semigroup arithmetic model, *see* models of computation
separates, 96
separation of lines and pseudolines, 29
Sharir, Micha, 13, 62, 71, 74
Shor, Peter, 31, 37
simple minor, 69
simpler, 39, 97
simplicial polytope, 33
Snoeyink, Jack, viii, 31, 38, 62
Spencer, Joel, 68
Steele, J. Michael, 6, 48
Steiger, Bill, 29, 59
Stolfi, Jorge, 37
Streinu, Ileana, 29, 59
supporting hyperplane, 32, 94
suspension, 95, 99
Swart, Garret, 37
Sylvester, James, 22
Székeley, László, 71
Szemerédi, Endre, 71

Tarján, T. G., 68
Tarjan, Robert, 9
Tarski, Alfred, 48–49
Thom, René, 6
Transfer Principle, 49, 56
Trotter, William T., 71
Tuza, Zsolt, 68

vague "conjecture", 28–30, 39
Vandermonde matrix, 52, 77
  almost, 20
vertex
  of a polytope, 33

weird moment curve, 20–22, 33–36
Welzl, Emo, vii, viii, 62, 71, 74
Whittlesey, Kim, viii
Willard, Fred, 8

Yao, Andrew Chi-Chih, 5, 7, 31, 32, 48
Yap, Chee, 31, 38, 62
yellow pig, 17

zero cover, 68
Zhuangzi (Chuang-tsu), 12
Ziegler, Günter, 32, 39, 94, 107